AlignDistil: Token-Level Language Model Alignment as Adaptive Policy Distillation

Anonymous ACL submission

Abstract

In modern large language models (LLMs), LLM alignment is of crucial importance and is typically achieved through methods such as reinforcement learning from human feedback (RLHF) and direct preference optimization (DPO). However, in most existing methods for LLM alignment, all tokens in the response are optimized using a sparse, response-level reward or preference annotation. The ignorance of token-level rewards may erroneously punish high-quality tokens or encourage low-quality tokens, resulting in suboptimal performance and slow convergence speed. To address this issue, we propose ***AlignDistil***, a RLHF-equivalent distillation method for token-level reward optimization. Specifically, we introduce the reward learned by DPO into the RLHF objective and theoretically prove the equivalence between this objective and a token-level distillation process, where the teacher distribution linearly combines the logits from the DPO model and a reference model. On this basis, we further bridge the accuracy gap between the reward from the DPO model and the pure reward model, by building a contrastive DPO reward with a normal and a reverse DPO model. Moreover, to avoid under- and over-optimization on different tokens, we design a token adaptive logit extrapolation mechanism to construct an appropriate teacher distribution for each token. Experimental results demonstrate the superiority of our AlignDistil over existing methods and showcase fast convergence due to its token-level distributional reward optimization.

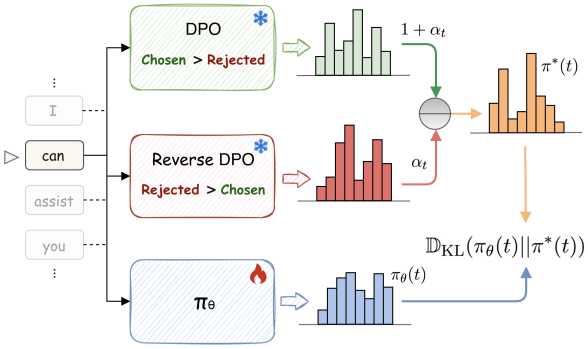

Figure 1: An overview of our AlignDistil. At token position t, the distribution from the current policy $\pi_\theta(t)$ is guided by a teacher distribution $\pi^*(t)$, which is constructed from an adaptive extrapolation between logit distributions from a DPO model and a reverse DPO model with a weight α_t.

1 Introduction

Current large language models (LLMs) have demonstrated remarkable capabilities in producing human-desired outputs under different circumstances (Bai et al., 2022; Ouyang et al., 2022; Llama Team, 2024). This is largely achieved by a key procedure in the post-training of LLMs, *i.e.*, LLM alignment with human preference. Existing solutions for LLM alignment mainly includes reinforcement learning from human feedback (RLHF) (Christiano et al., 2017; Stiennon et al., 2020; Bai et al., 2022; Ouyang et al., 2022) and direct preference learning algorithms (Rafailov et al., 2024b; Azar et al., 2024; Ethayarajh et al., 2024). Therein, RLHF is a two-stage method that first 1) trains a response-level reward model based on human preference labels, and then 2) optimizes the policy model with RL algorithms under this reward model while preventing deviation from the initial model. Alternatively, direct preference learning algorithms, *e.g.*, direct preference optimization (DPO, Rafailov et al. 2024b), simplify RLHF via parameterizing the reward with the policy model and directly training it on the preference data.

Despite their prevalence, most existing methods for LLM alignment optimize tokens with a sparse, response-level reward or preference annotation. However, this response-level feedback is coarse-grained and lacks reflection on the individual contribution of each token in the response

(Yoon et al., 2024; Li et al., 2024c; Xia et al., 2024; Yang et al., 2024c), which may erroneously punish tokens with high quality or encourage tokens with low quality. Consequently, those methods based on response-level feedback have been revealed with limitations on both performance and convergence speed (Chan et al., 2024; Zhong et al., 2024; Liu et al., 2024a).

To address this issue, in this paper, we propose ***AlignDistil*** (as shown in Figure 1), a simple distillation method derived from the RLHF objective for token-level reward optimization. Specifically, our method starts from introducing the DPO reward (Rafailov et al., 2024b) into the original objective of RLHF. Based on the property of token-level decomposition of the DPO reward (Rafailov et al., 2024a), we prove a theoretical equivalence between the original sequence-level objective of RLHF and a token-level distillation objective. In this distillation objective, the current policy is guided by a teacher distribution that linearly combines the logit distribution output from the two LLMs in the DPO reward. Built on this theoretical finding, our AlignDistil further involves two targeted designs for token-level optimization. Firstly, given that rewards from DPO generally perform worse than those from pure reward models (Lin et al., 2024), we use a contrastive DPO reward for AlignDistil with a DPO model and a reverse DPO model (Liu et al., 2024a), which yields better generalization performance than the vanilla DPO reward. Furthermore, to mitigate imbalanced under- and over-optimization across different tokens, we design a token adaptive logit extrapolation mechanism to construct an appropriate teacher distribution for each token position. Overall, our AlignDistil uses a simple distillation objective to achieve token-level reward optimization. Additionally, its training can flexibly switch between on-policy and off-policy, trading off between effectiveness and efficiency.

We evaluate the effectiveness of our method on three common benchmarks for LLM alignment, *i.e.*, AlpacaEval 2.0 (Dubois et al., 2024), MT-Bench (Zheng et al., 2023) and Arena-Hard (Li et al., 2024b). Experimental results demonstrate the superiority of our AlignDistil over existing methods and showcase the effectiveness of the targeted designs in the method. Moreover, AlignDistil exhibits a faster convergence speed compared to the variants with response-level and token-level scalar-type rewards, highlighting the advantage of token-level distributional reward optimization.

In a nutshell, the contributions of this paper are as follows:

- We build a theoretical equivalence between RLHF with DPO reward and a distillation process, which offers a new perspective for performing token-level reward optimization.
- On this basis, we design AlignDistil, a simple distillation method with a contrastive DPO reward and a token adaptive logit extrapolation.
- Experimental results showcase that AlignDistil significantly outperforms existing methods and achieves faster convergence due to the token-level distributional reward optimization.

2 Preliminary

2.1 Reinforcement Learning from Human Feedback

Generally, RLHF contains two stages, *i.e.*, reward modeling and policy optimization.

Reward Modeling. Reward modeling generally needs a human-labeled preference dataset with N samples $D = \{(x, y_w, y_l)_i\}_N$, where x is the prompt from the user, and y_w/y_l represents the human-annotated preferred/dispreferred response. Then, the human preference within the data is modeled by a reward model using the Bradley-Terry model (Bradley and Terry, 1952), which optimizes the reward r_ϕ with the following loss function:

$$\mathcal{L}_{\mathrm{RM}}(\phi) = -\mathop{\mathbb{E}}_{(x,y_w,y_l)\sim D}\Big[\log\sigma(r_\phi(x,y_w) - r_\phi(x,y_l))\Big]. \quad (1)$$

Policy Optimization. Afterward, the policy model π_θ (*i.e.*, the LLM) is optimized with RL algorithms like PPO to maximize its expected reward while preventing π_θ from being too far from the reference model π_{ref}:

$$\mathcal{J}_{\mathrm{RLHF}}(\theta) = \max_\theta \mathop{\mathbb{E}}_{\substack{x\sim D\\ y\sim\pi_\theta(\cdot|x)}}\Big[r_\phi(x,y) - \beta\log\frac{\pi_\theta(y|x)}{\pi_{\mathrm{ref}}(y|x)}\Big], \quad (2)$$

where β is a hyper-parameter to control the Kullback-Leibler (KL) divergence (Kullback and Leibler, 1951) from the reference model.

2.2 Direct Preference Optimization and its Implicit Reward

Although RLHF is proposed as the initial solution for LLM alignment, the process is somewhat complicated and expensive. To address this, Rafailov et al. (2024b) propose direct preference optimization (DPO) to directly train the LLM in the reward modeling stage. Specifically, they leverage the closed-form solution of the RLHF objective and parameterize the reward with a log ratio:

$$r_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)} + \beta \log Z(x), \quad (3)$$

where $Z(x)$ is the partition function and independent to y. Then the training objective of DPO is derived by substituting Eq. (3) into Eq. (1):

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim D}\Big[\log \sigma\Big(\beta \log \frac{\pi_\theta(y_w|x)}{\pi_{\text{ref}}(y_w|x)} - \beta \log \frac{\pi_\theta(y_l|x)}{\pi_{\text{ref}}(y_l|x)}\Big)\Big]. \quad (4)$$

Besides, Rafailov et al. (2024b) point out that $Z(x)$ in Eq. (3) can be omitted without loss of generality:

$$r_{\text{dpo}}(x, y) = \beta \log \frac{\pi_{\text{dpo}}(y|x)}{\pi_{\text{ref}}(y|x)}, \quad (5)$$

and token-level reward (Rafailov et al., 2024a) can be further represented by

$$r_{\text{dpo}}(x, y_{<t}, y_t) = \beta \log \frac{\pi_{\text{dpo}}(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)}. \quad (6)$$

These concise forms of reward further facilitate researches on self-rewarding (Chen et al., 2024a) and fine-grained optimization (Xia et al., 2024; Zhong et al., 2024; Yang et al., 2024c). Likewise, in this work, we also leverage the DPO reward and derive a RLHF-equivalent distillation objective for token-level reward optimization.

3 Theoretical Analysis: From RLHF to Policy Distillation

In this section, we provide a theoretical analysis for RLHF with DPO reward, building a connection between the objectives of RLHF and distillation. As presented in Sec. 2.2, DPO parameterizes the reward with the log ratio between two language models and trains it with the same objective of reward modeling. Thus, the first intuition of this work is to substitute the reward in Eq. (5) trained by DPO into the original RLHF objective:

$$\widetilde{\mathcal{J}}_{\text{RLHF}}(\theta) = \max_\theta \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \Big[r_{\text{dpo}}(x, y) - \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\Big] \quad (7)$$

$$= \max_\theta \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \Big[\underbrace{\beta_0 \log \frac{\pi_{\text{dpo}}(y|x)}{\pi_{\text{ref}}(y|x)}}_{\text{DPO reward}} - \underbrace{\beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}}_{\text{KL divergence}}\Big], \quad (8)$$

where β_0 denotes the original coefficient in DPO training and is a constant in this objective.

It can be found that both the DPO reward and the KL divergence in Eq. (8) can be decomposed into the sum of token-level results, which offers the potential to reformulate this objective into a token-level form. Through solving this, we build a connection between RLHF with DPO reward and a policy distillation process, as described in the following theorem:

Theorem 1. *Under the DPO reward, the RLHF objective is equivalent to the following token-level policy distillation objective:*

$$\max_\theta \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \Big[r_{\text{dpo}}(x, y) - \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\Big] \quad (9)$$

$$= \min_\theta \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \mathbb{D}_{\text{KL}}(\pi_\theta(\cdot|y_{<t}, x)||\pi^*(\cdot|y_{<t}, x)), \quad (10)$$

where $\mathbb{D}_{\text{KL}}(\cdot||\cdot)$ is token-level KL divergence and $\pi^*(\cdot|x, y_{<t})$ is the probability distribution output by the softmax function on a synthetic logit distribution z_t^*:

$$z_t^* = \frac{\beta_0}{\beta} z_t^{\text{dpo}} + (1 - \frac{\beta_0}{\beta}) z_t^{\text{ref}}, \quad (11)$$

where z_t^{dpo} and z_t^{ref} denote logit distributions of the DPO model and the reference model at t-th token position.

The proof is provided in Appendix A. Theorem 1 indicates that with DPO reward, we can equivalently convert the original sequence-level RLHF objective into a token-level distillation objective, thereby naturally achieving token-level reward optimization.

4 AlignDistil

In this section, we will introduce our AlignDistil motivated by the above theoretical analysis. Built on the theory, AlignDistil additionally introduces two intuitive designs, *i.e.*, contrastive DPO reward (§4.1) and token adaptive logit extrapolation (§4.2). Lastly, we will conclude the objectives of AlignDistil for both on-policy and off-policy training (§4.3).

4.1 Contrastive DPO Reward

Although the DPO reward can theoretically represent any reward under the Bradley-Terry model (Rafailov et al., 2024b), it has been pointed out to be less accurate than a pure reward model in practice (Lin et al., 2024). We also observe this phenomenon in our experiments (see Table 2) and conjecture that this imperfect reward estimation will impact the final alignment performance. Thus, in AlignDistil, we parameterize the DPO reward by a pair of contrastive DPO models (Liu et al., 2024a), *i.e.*, a normal DPO model and a reverse DPO model (trained by switch chosen-rejected pairs in training data). Intuitively, a reverse DPO model is more appropriate for the DPO reward as it captures negative features in low-quality data and makes the reward more discriminative. Formally, this contrastive DPO reward can be represented as:

$$r_{\text{ctr}}(x, y) = \beta_0 \log \frac{\pi_{\text{dpo}}(y|x)}{\pi^{-}_{\text{dpo}}(y|x)}, \tag{12}$$

where π^{-}_{dpo} represents the reverse DPO model. Note that the contrastive DPO reward introduces a new model π^{-}_{dpo} to the objective and increases the training cost. To solve this, we switch the reference model in the RLHF objective from the initial model to the DPO model π_{dpo}. This not only saves the required models in training, but also moves the reference model forward for better alignment. Afterward, the objective of RLHF in Eq. (9) becomes:

$$\max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \left[\beta_0 \log \frac{\pi_{\text{dpo}}(y|x)}{\pi^{-}_{\text{dpo}}(y|x)} - \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{dpo}}(y|x)}\right]. \tag{13}$$

Correspondingly, the synthetic logit distribution in Eq. (22) also changes to

$$z_t^* = (1 + \frac{\beta_0}{\beta}) z_t^{\text{dpo}} - \frac{\beta_0}{\beta} z_t^{\text{dpo}^-} \tag{14}$$

$$= \underbrace{z_t^{\text{dpo}}}_{\text{DPO distribution}} + \underbrace{\frac{\beta_0}{\beta}(z_t^{\text{dpo}} - z_t^{\text{dpo}^-})}_{\text{reward distribution}}. \tag{15}$$

The detailed derivation can be referred to in Appendix B. Given that $\beta_0 > 0$ and $\beta > 0$, this equation strictly describes an extrapolation between logit distributions of the DPO model and the reverse DPO model. The extrapolation is crucial for pushing the current policy to surpass the DPO model, since it constructs a stronger aligned distribution by removing some "*negative*" information from the reverse DPO model and has been proven effective in (Liu et al., 2024b).

4.2 Token Adaptive Logit Extrapolation

Although logit extrapolation theoretically provides a stronger distribution, we find that it is tricky to select an appropriate β in practice. Specifically, a large β yields a small $\frac{\beta_0}{\beta}$ and may result in under-optimization, while a small β produces a drastic distribution and tends to over-optimize the current policy. Considering that tokens in the sequence have different tendencies, we design a token-level adaptive weight to adjust $\frac{\beta_0}{\beta}$ for each token position. Specifically, we use the total variation distance (TVD)[1] (Levin and Peres, 2017) between the DPO distribution and the reverse DPO distribution to calculate a coefficient α_t for position t:

$$\alpha_t = \mathbb{D}_{\text{TVD}}(t) * r + \epsilon \in [\epsilon, r + \epsilon], \tag{16}$$

where $\mathbb{D}_{\text{TVD}}(t) := \frac{1}{2}\sum_{y_t\in\mathcal{V}} |\pi_{\text{dpo}}(y_t|y_{<t}, x) - \pi^{-}_{\text{dpo}}(y_t|y_{<t}, x)|$, $\mathcal{V}$ is the vocabulary, r is a hyperparameter to control the upper-bound of the coefficient, and $\epsilon = 0.001$ is a small value to avoid $\alpha_t = 0$. The intuition is that when DPO distribution is far from the reverse one, this position may have a key impact on the final reward and thus should learn from a stronger teacher distribution. Accordingly, we modify Eq. (15) as follows:

$$z_t^* = z_t^{\text{dpo}} + \alpha_t(z_t^{\text{dpo}} - z_t^{\text{dpo}^-}). \tag{17}$$

Note that we replace the constant $\frac{\beta_0}{\beta}$ with an adaptive weight α_t, and thus the static β in Eq. (10) also becomes adaptive as $\beta_t = \frac{\beta_0}{\alpha_t}$.

4.3 Overall Objectives

The theoretical objective of AlignDistil follows Eq. (10) with a synthetic teacher distribution π^* calculated from Eq. (17). It defines AlignDistil as an on-policy algorithm. Practically, the loss for on-policy training of AlignDistil relies on Monte-Carlo sampling to estimate the expectation in Eq.

[1] We choose TVD since it is symmetric and computationally efficient with a limited range in $[0, 1]$.

(10) and calculate the token-level distillation loss:

$$\mathcal{L}_{\text{AD}}^{\text{on}} = \tag{18}$$

$$\frac{1}{|\mathcal{B}|}\sum_{x\in\mathcal{B}}\frac{\beta_t}{|\hat{y}|}\sum_{t=1}^{|\hat{y}|}\mathbb{D}_{\text{KL}}(\pi_\theta(\cdot|\hat{y}_{<t},x)||\pi^*(\cdot|\hat{y}_{<t},x)),$$

where $\mathcal{B}$ represents the mini-batch of prompts sampled from the prompt dataset $\{(x)_i\}_N$ and $\hat{y}$ is sampled from $\pi_\theta(\cdot|x)$. As this loss function is actually a supervised distillation loss, we can also construct an off-policy version using a prompt-response dataset $\{(x,y)_i\}_N$:

$$\mathcal{L}_{\text{AD}}^{\text{off}} = \tag{19}$$

$$\frac{1}{|\mathcal{B}|}\sum_{(x,y)\in\mathcal{B}}\frac{\beta_t}{|y|}\sum_{t=1}^{|y|}\mathbb{D}_{\text{KL}}(\pi_\theta(\cdot|y_{<t},x)||\pi^*(\cdot|y_{<t},x)).$$

5 Experiments

In this section, we present the experimental setups and showcase the evaluation results of our method.

5.1 Experimental Setup

Models. In our experiments, we use two instruct models, *i.e.*, `Qwen2-1.5B-Instruct` (Yang et al., 2024a) and `Qwen2.5-1.5B-Instruct` (Yang et al., 2024b) as the initial models for further alignment.

Datasets and Training. Following most previous work (Meng et al., 2024), we use UltraFeedback (Cui et al., 2023) as the training dataset, which contains about 63K prompts and corresponding response pairs with preference annotation. Specifically, for DPO and reward modeling, we use both the prompts and the response pairs for training, while for other on-policy methods including ours, we only use the prompts for training. For off-policy AlignDistil, we use the prompts and the preferred response in UltraFeedback. For all experiments, we train the initial model for 1 epoch, with a batch size of 128, a learning rate of 1e-6, and a warmup ratio of 0.1. All our experiments are conducted on 8 $\times$ A100-40G GPUs. More training details are provided in Appendix C.

Evaluation. Following the common practice (Meng et al., 2024; Kim et al., 2024), we choose the following three benchmarks to evaluate the alignment performance of all the models:

- **AlpacaEval 2.0** (Dubois et al., 2024) consists of 805 instructions with the responses of GPT-4 as the baseline. The evaluated responses are compared with the baseline by an LLM evaluator. We report the win rate (**WR**) and the length-controlled win rate (**LC WR**) for each model, where the LC WR is designed to eliminate the length bias in LLM-as-Judge.
- **MT-Bench** (Zheng et al., 2023) contains 80 multi-turn questions and assesses the quality of responses with scores between $[1, 10]$ by an LLM evaluator. We report the scores in the 1st turn (**1st Turn**) and the 2nd turn (**2nd Turn**) and the final averaged scores (**Avg.**).
- **Arena-Hard** (Li et al., 2024b) incorporates 500 technical problem-solving queries with the responses of GPT-4 as the baseline. We report the win rate (**WR**) and the style-controlled win rate (**SC WR**) to mitigate the style bias in LLM evaluation.

We choose `Qwen2.5-72b-Instruct` as the automatic evaluator since we find that it achieves comparable judgment performance with GPT-4 with a much lower price (see Table 4).

5.2 Baseline Methods

We compare our method to the following methods:

DPO. DPO (Rafailov et al., 2024b) is the most common direct preference learning method. The model trained by DPO is used both as a baseline and to calculate rewards for our method.

KTO. KTO (Ethayarajh et al., 2024) is a direct preference learning method like DPO but optimizes on the non-paired preference data.

TDPO. Zeng et al. (2024) propose token-level DPO (TDPO) by equipping DPO reward with token-level forward KL constraint. This method contains two versions, *i.e.*, TDPO_1 and TDPO_2.

SimPO. SimPO (Meng et al., 2024) is also a direct preference learning method and further simplifies DPO by removing the reference model.

PPO. PPO (Schulman et al., 2017) is selected as the default RL algorithm for RLHF, which optimizes the advantages of the policy estimated by generalized advantage estimator (GAE).

RTO. Zhong et al. (2024) propose reinforced token optimization (RTO) by substituting token-level DPO reward from Eq. (6) into PPO.

The implementation details for these baselines are provided in Appendix C.

Methods	AlpacaEval 2.0		MT-Bench			Arena-Hard	
	LC WR (%)	WR (%)	1st Turn	2nd Turn	Avg.	WR (%)	SC WR (%)
Qwen2-1.5B-Instruct							
Initial Model	3.10	1.99	6.11	5.15	5.63	1.8	2.8
DPO (Rafailov et al., 2024b)	6.42	5.03	6.19	5.59	5.89	3.0	3.6
$\text{DPO}_{\beta=0.01}$ (Rafailov et al., 2024b)	10.72	11.61	6.70	6.06	6.38	7.0	**6.8**
KTO (Ethayarajh et al., 2024)	7.16	6.34	6.54	5.55	6.05	3.4	4.2
SimPO (Meng et al., 2024)	8.19	9.63	5.94	5.71	5.83	6.9	4.2
TDPO_1 (Zeng et al., 2024)	6.58	4.60	6.53	5.64	6.08	3.2	3.9
TDPO_2 (Zeng et al., 2024)	3.59	2.42	6.25	5.06	5.66	1.3	1.9
PPO (Schulman et al., 2017)	4.86	4.41	6.76	5.51	6.13	2.7	3.0
RTO (Zhong et al., 2024)	8.92	9.32	6.46	6.06	6.26	6.7	5.9
Off-Policy AlignDistil (ours)	11.79	14.29	6.83	5.68	6.25	10.5	6.0
On-Policy AlignDistil (ours)	**12.93**	**15.65**	**6.89**	**6.13**	**6.45**	**11.0**	6.7
Qwen2.5-1.5B-Instruct							
Initial Model	12.57	8.94	7.15	6.05	6.60	16.8	12.7
DPO (Rafailov et al., 2024b)	14.35	10.74	7.39	6.58	6.98	17.1	14.8
$\text{DPO}_{\beta=0.01}$ (Rafailov et al., 2024b)	14.09	14.29	7.36	6.54	6.95	16.2	15.5
KTO (Ethayarajh et al., 2024)	14.07	10.00	7.41	6.59	7.00	15.0	12.3
SimPO (Meng et al., 2024)	11.61	9.81	7.43	6.96	7.20	4.0	4.0
TDPO_1 (Zeng et al., 2024)	13.19	9.94	7.45	6.66	7.06	16.5	14.1
TDPO_2 (Zeng et al., 2024)	13.64	9.07	7.57	6.48	7.02	15.8	13.0
PPO (Schulman et al., 2017)	18.06	12.67	7.60	6.81	7.21	15.9	13.7
RTO (Zhong et al., 2024)	16.54	15.53	7.37	6.51	6.94	18.2	16.6
Off-Policy AlignDistil (ours)	**21.16**	**24.29**	7.62	6.53	7.07	**24.1**	21.8
On-Policy AlignDistil (ours)	19.45	22.11	**7.65**	**6.98**	**7.31**	24.0	**23.0**

Table 1: Evaluation results of baselines and our AlignDistil on three benchmarks. The best results are **bolded**. "$\text{DPO}_{\beta=0.01}$" represents DPO training with $\beta = 0.01$.

5.3 Main Results

The evaluation results on three benchmarks are listed in Table 1. We can draw several conclusions from the results: **1) Overall, both on-policy and off-policy AlignDistil significantly outperform baseline methods.** Although the teacher distributions in our AlignDistil are constructed from DPO models, the performance of AlignDistil surpasses DPO by a notable margin (*e.g.*, over 6 % improvement for length-controlled win rates on AlpacaEval 2.0). since Liu et al. (2024b) reveal that logit extrapolation in inference is similar to rescale β in DPO training, we also implement another DPO with $\beta = 0.01$ (noted as $\text{DPO}_{\beta=0.01}$). We observe that rescaling β does not always lead to improvement (*e.g.*, the results on Qwen2.5-1.5B-Instruct), which indicates that a simple logit extrapolation may not stably improve the performance and the design of the contrastive DPO reward and token adaptive logit extrapolation are necessary. **2) AlignDistil yields better token-level LLM alignment.** $\text{TDPO}_{1/2}$ introduces token-level KL constraint for DPO, while this constraint may limit the performance on small models. Besides, RTO introduces token-level DPO rewards into PPO and exhibits strong performance, especially surpassing PPO significantly on Qwen2-1.5B-Instruct. This superiority highlights the benefits of token-level rewards in LLM alignment. Nevertheless, our AlignDistil performs even better than RTO on both models since we further leverage the whole reward distribution instead of the scalar reward on the predicted token for optimization. **3) Off-policy AlignDistil performs competitively to the on-policy version.** Different from most methods for LLM alignment, our AlignDistil can work under both on-policy training and off-policy training. On Qwen2-1.5B-Instruct, on-policy AlignDistil performs better, while off-policy AlignDistil performs comparably with the on-policy one on Qwen2.5-1.5B-Instruct. We conjecture that for the DPO reward, the data for off-policy AlignDistil is in-distribution, while the data for on-policy AlignDistil is generated by the current policy model and is

Reward	Train Acc. (%)	Test Acc. (%)	AE2 LC. (%)
Reward Model	70.41	71.19	-
DPO Reward	72.85	69.53	16.51
Contrastive DPO Reward	**74.71**	**71.29**	**19.45**

Table 2: Reward accuracy of different types of rewards on 1000 samples from the training set and test set of UltraFeedback and the corresponding length-controlled win rate on AlpacaEval 2.0. All models are based on Qwen2.5-1.5B-Instruct.

$\frac{\beta_0}{\beta}$	Type	$\mathbb{D}_{\text{KL}}(\pi_\theta \| \pi_{\text{dpo}}) \downarrow$	Avg. Length $\downarrow$	AE2 LC. (%) $\uparrow$
1.0	constant	10.16	2332	18.40
1.2	constant	13.95	2481	18.33
1.5	constant	20.71	2973	19.49
1.8	constant	28.55	3644	21.87
2.0	constant	34.52	4434	25.17
α_t	token adaptive	22.95	2424	21.16

Table 3: Comparisons between constant extrapolation weight and token adaptive extrapolation weight. The training is off-policy to mitigate the impact of data.

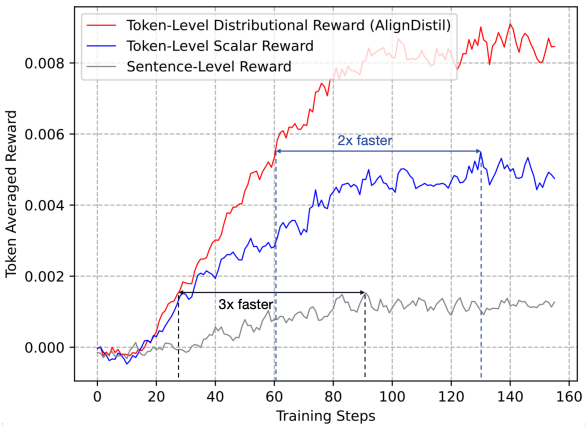

Figure 2: Convergence curves of token averaged reward from optimization on the sentence-level, token-level scalar-type, and token-level distributional reward.

out-of-distribution. These promising results suggest the off-policy AlignDistil as an efficient and effective method for LLM alignment.

6 Analysis

In this section, we first conduct the ablation study by separately analyzing the two designs in AlignDistil, *i.e.*, the contrastive DPO reward (§6.1) and the token adaptive logit extrapolation (§6.2). Then we showcase the advantage of our AlignDistil on the convergence speed (§6.3).

6.1 DPO Reward: Contrastive vs. Vanilla

Lin et al. (2024) reveal that DPO reward often shows inferior generalization performance than a pure reward model. We also verify this phenomenon in Table 2. Specifically, we calculate the response-level accuracy of different types of rewards on the training set and the test set of UltraFeedback. Table 2 shows a performance gap between the DPO reward and the reward model. By contrast, the contrastive DPO reward not only shows better accuracy than the vanilla DPO reward on training data, but also generalizes better on test data, even surpassing the reward model. Correspondingly, our on-policy AlignDistil with the contrastive DPO reward outperforms the one with the vanilla DPO reward on AlpacaEval 2.0. This performance gain can be attributed to the reverse DPO model, which captures subtle features in low-quality responses. Therefore, the contrastive DPO reward plays a key role in our AlignDistil.

6.2 Effect of Adaptive Logit Extrapolation

In our AlignDistil, we design a token adaptive logit extrapolation before constructing the teacher distribution. The motivation is that a constant extrapolation weight $\frac{\beta_0}{\beta}$ for all tokens tends to over-optimize or under-optimize on some tokens. Therefore, we explore whether this motivation holds. Specifically, we set the extrapolation weight $\frac{\beta_0}{\beta}$ to a constant and test the performance and the KL divergence from the DPO model π_{dpo} as well as the average response length under different constants. As shown in Table 3, when the constant is small (*e.g.*, 1.0 and 1.2), the teacher distribution is similar to the distribution of the DPO model π_{dpo}, reflecting by a small KL divergence. However, the mild extrapolation also limits the strength of the teacher distribution and leads to under-optimization of the current policy. By contrast, although a large extrapolation weight (*e.g.*, 1.8 and 2.0) indeed yields better performance on AlpacaEval, the current policy is over-optimized, showcasing too much deviation from the DPO model and extremely increasing the response length. Compared to these constant values, our token adaptive extrapolation weight considers the individual characteristics of different tokens and assigns an appropriate weight for each position, thus achieving a balance between performance and deviation.

6.3 Convergence Speed

As noted in previous literature (Chan et al., 2024; Zhong et al., 2024), token-level reward optimiza-

tion generally yields a faster convergence speed than sentence-level reward optimization. We also test the convergence speed of our AlignDistil. Specifically, we train the on-policy AlignDistil with a static β on a subset (10k prompts) of UltraFeedback. For comparison, we implement a sentence-level optimization method that optimizes the overall reward on the whole response as a bandit problem[2] and a REINFORCE-based method that optimizes on token-level scalar-type DPO rewards. The coefficient β is set to 0.08 for all methods[3]. The convergence curves on token averaged reward[4] corresponding to the training steps are plotted in Figure 2. It is shown that sentence-level reward optimization yields the poorest convergence, significantly lagging behind token-level rewards, which is consistent with the conclusion in (Chan et al., 2024; Zhong et al., 2024) Moreover, although token-level scalar reward boosts the convergence speed, our AlignDistil still has a more than $2\times$ faster convergence speed. The reason is that the optimization of AlignDistil leverages the whole reward distribution instead of a single scalar reward, allowing exact calculation of the reward expectation at each token position. This comparison sufficiently demonstrates the benefits of our AlignDistil on token-level reward optimization.

7 Related Work

Fine-Grained LLM Alignment. Existing methods for LLM alignment are criticized for optimizing sparse, coarse-grained rewards. To address this, (Lightman et al., 2023) propose the process reward model (PRM) trained with step-level human annotations for complicated LLM reasoning. On this basis, several methods are proposed to automatically collect step-level rewards without human annotation (Wang et al., 2024; Luo et al., 2024; Yuan et al., 2024). Besides, Cao et al. (2024) extract span-level rewards from LLM critiques to enhance the PPO algorithm. Furthermore, there are solutions for token-level reward signals via edit distance (Guo et al.; Chen et al., 2024b), attention scores in the reward model (Chan et al., 2024), and reward model outputs on intermediate tokens (Li et al., 2024a). Additionally, Rafailov et al. (2024a) reveal that DPO also automatically learns token-level reward. Afterward, this token-level DPO reward is applied to existing alignment methods like DPO (Liu et al., 2024a; Yang et al., 2024c) and PPO (Zhong et al., 2024) or new algorithms (Xia et al., 2024). Following this line, we also leverage the DPO reward in our method, while the difference is that we further exploit the distributional information in this reward for more sufficient optimization.

Knowledge Distillation for LLMs. Knowledge distillation (KD, Hinton, 2015) is proposed as an essential technique for compressing neural networks. With the emergence and development of LLMs, KD has attracted more attention to reduce the numerous parameters in LLMs. In this context, KD methods are divided into white-box KD (Hinton, 2015) and black-box KD (Kim and Rush, 2016), depending on whether the weight of the teacher model can be obtained. For white-box KD, approaches typically bridge probability distributions (Agarwal et al., 2024; Gu et al., 2024; Ko et al., 2024; Zhang et al., 2024) or intermediate features (Wang et al., 2020) between the teacher model and the student model. Intuitively, this process transfers sufficient information from the teacher model, thus often used for pre-training small yet powerful LLMs (Team et al., 2024; Meta, 2024). In contrast, black-box KD is actually more widely used for LLMs as it only requires collecting outputs from the teacher model and supervised fine-tuning the student model (Taori et al., 2023; Chiang et al., 2023; Tunstall et al., 2023). Different from these methods, our AlignDistil is derived from RLHF and aims for token-level reward optimization.

8 Conclusion

In this paper, we aim at the fine-grained LLM alignment problem and propose AlignDistil as the solution. Specifically, we introduce the DPO reward into the objective of RLHF and theoretically build an equivalence between RLHF and token-level policy distillation. On this basis, we design two components in our AlignDistil, *i.e.*, the contrastive DPO reward and token adaptive logit extrapolation, for better performance and stable optimization. Experimental results on prevalent alignment benchmarks sufficiently demonstrate the superiority of our AlignDistil compared to existing methods for LLM alignment. Moreover, we showcase that the token-level distributional reward optimization in AlignDistil offers a faster convergence speed than sentence-level and token-level scalar-type rewards.

[2]The final reward involves the same contrastive DPO reward and KL constraint as AlignDistil.

[3]Implementation details are provided in Appendix E.

[4]The token averaged reward is used to mitigate the length bias in DPO reward.

Limitations

Due to the resource limitation, the evaluation of our AlignDistil is limited within small language models (~1.5B). The evaluation on larger models is still under-explored and we leave this in future work.

References

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. 2024. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*.

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, et al. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*.

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

Meng Cao, Lei Shu, Lei Yu, Yun Zhu, Nevan Wichers, Yinxiao Liu, and Lei Meng. 2024. Enhancing reinforcement learning with dense rewards from language model critic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 9119–9138, Miami, Florida, USA. Association for Computational Linguistics.

Alex James Chan, Hao Sun, Samuel Holt, and Mihaela Van Der Schaar. 2024. Dense reward for free in reinforcement learning from human feedback. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 6136–6154. PMLR.

Changyu Chen, Zichen Liu, Chao Du, Tianyu Pang, Qian Liu, Arunesh Sinha, Pradeep Varakantham, and Min Lin. 2024a. Bootstrapping language models with dpo implicit rewards. *arXiv preprint arXiv:2406.09760*.

Zhipeng Chen, Kun Zhou, Wayne Xin Zhao, Junchen Wan, Fuzheng Zhang, Di Zhang, and Ji-Rong Wen. 2024b. Improving large language models via fine-grained reinforcement learning with minimum editing constraint. *arXiv preprint arXiv:2401.06081*.

Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. 2023. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

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

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.

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

Yuxian Gu, Li Dong, Furu Wei, and Minlie Huang. 2024. Minillm: Knowledge distillation of large language models. In *The Twelfth International Conference on Learning Representations*.

Geyang Guo, Ranchi Zhao, Tianyi Tang, Xin Zhao, and Ji-Rong Wen. Beyond imitation: Leveraging fine-grained quality signals for alignment. In *The Twelfth International Conference on Learning Representations*.

Geoffrey Hinton. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Dongyoung Kim, Kimin Lee, Jinwoo Shin, and Jaehyung Kim. 2024. Aligning large language models with self-generated preference data. *arXiv preprint arXiv:2406.04412*.

Yoon Kim and Alexander M Rush. 2016. Sequence-level knowledge distillation. *arXiv preprint arXiv:1606.07947*.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. 2024. Distillm: Towards streamlined distillation for large language models. *arXiv preprint arXiv:2402.03898*.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

David A Levin and Yuval Peres. 2017. *Markov chains and mixing times*, volume 107. American Mathematical Soc.

Jiahui Li, Tai-wei Chang, Fengda Zhang, Kun Kuang, and Long Chen. 2024a. R3hf: Reward redistribution for enhancing reinforcement learning from human feedback. *arXiv preprint arXiv:2411.08302*.

Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. 2024b. From live data to high-quality benchmarks: The arena-hard pipeline.

Wendi Li, Wei Wei, Kaihe Xu, Wenfeng Xie, Dangyang Chen, and Yu Cheng. 2024c. Reinforcement learning with token-level feedback for controllable text generation. *arXiv preprint arXiv:2403.11558*.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. *arXiv preprint arXiv:2305.20050*.

Yong Lin, Skyler Seto, Maartje Ter Hoeve, Katherine Metcalf, Barry-John Theobald, Xuan Wang, Yizhe Zhang, Chen Huang, and Tong Zhang. 2024. On the limited generalization capability of the implicit reward model induced by direct preference optimization. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 16015–16026, Miami, Florida, USA. Association for Computational Linguistics.

Aiwei Liu, Haoping Bai, Zhiyun Lu, Yanchao Sun, Xiang Kong, Simon Wang, Jiulong Shan, Albin Madappally Jose, Xiaojiang Liu, Lijie Wen, et al. 2024a. Tis-dpo: Token-level importance sampling for direct preference optimization with estimated weights. *arXiv preprint arXiv:2410.04350*.

Tianlin Liu, Shangmin Guo, Leonardo Bianco, Daniele Calandriello, Quentin Berthet, Felipe Llinares, Jessica Hoffmann, Lucas Dixon, Michal Valko, and Mathieu Blondel. 2024b. Decoding-time realignment of language models. *arXiv preprint arXiv:2402.02992*.

Meta AI Llama Team. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Liangchen Luo, Yinxiao Liu, Rosanne Liu, Samrat Phatale, Harsh Lara, Yunxuan Li, Lei Shu, Yun Zhu, Lei Meng, Jiao Sun, et al. 2024. Improve mathematical reasoning in language models by automated process supervision. *arXiv preprint arXiv:2406.06592*.

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

Team Meta. 2024. Llama3.2 model card.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

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

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024b. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

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

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. 2023. Stanford alpaca: An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

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

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

Peiyi Wang, Lei Li, Zhihong Shao, Runxin Xu, Damai Dai, Yifei Li, Deli Chen, Yu Wu, and Zhifang Sui. 2024. Math-shepherd: Verify and reinforce llms step-by-step without human annotations. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9426–9439.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. *Advances in Neural Information Processing Systems*, 33:5776–5788.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 2023. f-divergence minimization for sequence-level knowledge distillation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10817–10834, Toronto, Canada. Association for Computational Linguistics.

Han Xia, Songyang Gao, Qiming Ge, Zhiheng Xi, Qi Zhang, and Xuanjing Huang. 2024. Inverse-Q*: Token level reinforcement learning for aligning large language models without preference data. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 8178–8188, Miami, Florida, USA. Association for Computational Linguistics.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. 2024a. Qwen2 technical report. *Preprint*, arXiv:2407.10671.

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

Kailai Yang, Zhiwei Liu, Qianqian Xie, Jimin Huang, Erxue Min, and Sophia Ananiadou. 2024c. Selective preference optimization via token-level reward function estimation. *arXiv preprint arXiv:2408.13518*.

Eunseop Yoon, Hee Suk Yoon, SooHwan Eom, Gunsoo Han, Daniel Wontae Nam, Daejin Jo, Kyoung-Woon On, Mark A Hasegawa-Johnson, Sungwoong Kim, and Chang D Yoo. 2024. Tlcr: Token-level continuous reward for fine-grained reinforcement learning from human feedback. *arXiv preprint arXiv:2407.16574*.

Lifan Yuan, Wendi Li, Huayu Chen, Ganqu Cui, Ning Ding, Kaiyan Zhang, Bowen Zhou, Zhiyuan Liu, and Hao Peng. 2024. Free process rewards without process labels. *arXiv preprint arXiv:2412.01981*.

Yongcheng Zeng, Guoqing Liu, Weiyu Ma, Ning Yang, Haifeng Zhang, and Jun Wang. 2024. Token-level direct preference optimization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 58348–58365. PMLR.

Songming Zhang, Xue Zhang, Zengkui Sun, Yufeng Chen, and Jinan Xu. 2024. Dual-space knowledge distillation for large language models. *arXiv preprint arXiv:2406.17328*.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric Xing, et al. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36:46595–46623.

Han Zhong, Guhao Feng, Wei Xiong, Xinle Cheng, Li Zhao, Di He, Jiang Bian, and Liwei Wang. 2024. Dpo meets ppo: Reinforced token optimization for rlhf. *Preprint*, arXiv:2404.18922.

A Proof of Theorem 1

Here we recall Theorem 1:

Theorem. *Under the DPO reward, the RLHF objective is equivalent to the following token-level policy distillation objective:*

$$\max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \Big[r_{\text{dpo}}(x,y) - \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\Big] \tag{20}$$

$$= \min_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \mathbb{D}_{\text{KL}}(\pi_\theta(\cdot|x, y_{<t})||\pi^*(\cdot|x, y_{<t})), \tag{21}$$

where $\mathbb{D}_{\text{KL}}(\cdot||\cdot)$ is token-level KL divergence and $\pi^*(\cdot|x, y_{<t})$ is the probability distribution output by the softmax function on a synthetic logit distribution z_t^*:

$$z_t^* = \frac{\beta_0}{\beta} z_t^{\text{dpo}} + (1 - \frac{\beta_0}{\beta}) z_t^{\text{ref}}, \tag{22}$$

where z_t^{dpo} and z_t^{ref} denote logit distributions of the DPO model and the reference model at t-th token position.

Proof. First, we need to decompose the objective of RLHF into token level. Inspired by Wen et al. (2023), we derive the decomposition process from the objective in Eq. (8):

$$\widetilde{\mathcal{J}}_{\text{RLHF}}(\theta) = \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \Big[\beta_0 \log \frac{\pi_{\text{dpo}}(y|x)}{\pi_{\text{ref}}(y|x)} - \beta \log \frac{\pi_\theta(y|x)}{\pi_{\text{ref}}(y|x)}\Big]$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y_{1:T}\sim\pi_\theta(\cdot|x)}} \Big[\beta_0 \sum_{t=1}^{T} \log \frac{\pi_{\text{dpo}}(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)} - \beta \sum_{t=1}^{T} \log \frac{\pi_\theta(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)}\Big] \tag{23}$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y_{1:T}\sim\pi_\theta(\cdot|x)}} \sum_{t=1}^{T} \Big[\beta_0 \log \frac{\pi_{\text{dpo}}(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)} - \beta \log \frac{\pi_\theta(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)}\Big] \tag{24}$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y_{1:T}\sim\pi_\theta(\cdot|x)}} \sum_{t=1}^{T-1} \Big[\beta_0 \log \frac{\pi_{\text{dpo}}(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)} - \beta \log \frac{\pi_\theta(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)}\Big] \tag{25}$$

$$+ \mathop{\mathbb{E}}_{\substack{x\sim D \\ y_{1:T}\sim\pi_\theta(\cdot|x)}} \Big[\beta_0 \log \frac{\pi_{\text{dpo}}(y_T|y_{<T}, x)}{\pi_{\text{ref}}(y_T|y_{<T}, x)} - \beta \log \frac{\pi_\theta(y_T|y_{<T}, x)}{\pi_{\text{ref}}(y_T|y_{<T}, x)}\Big] \tag{26}$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y_{1:T-1}\sim\pi_\theta(\cdot|x)}} \sum_{t=1}^{T-1} \Big[\beta_0 \log \frac{\pi_{\text{dpo}}(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)} - \beta \log \frac{\pi_\theta(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)}\Big] \tag{27}$$

$$+ \mathop{\mathbb{E}}_{\substack{x\sim D \\ y_{1:T-1}\sim\pi_\theta(\cdot|x)}} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<T}, x) \Big[\beta_0 \log \frac{\pi_{\text{dpo}}(y_t|y_{<T}, x)}{\pi_{\text{ref}}(y_t|y_{<T}, x)} - \beta \log \frac{\pi_\theta(y_t|y_{<T}, x)}{\pi_{\text{ref}}(y_t|y_{<T}, x)}\Big] \tag{28}$$

Eq. (28) is derived by decomposing the expectation in Eq. (26) at the last step and exactly calculating it. Likewise, we can recursively decompose the expectation from step $T-1$ to step 1 and obtain the final token-level representation:

$$\widetilde{\mathcal{J}}_{\text{RLHF}}(\theta) = \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t}, x) \Big[\beta_0 \log \frac{\pi_{\text{dpo}}(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)} - \beta \log \frac{\pi_\theta(y_t|y_{<t}, x)}{\pi_{\text{ref}}(y_t|y_{<t}, x)}\Big] \tag{29}$$

Then, we reorganize the log ratio in Eq. (40):

$$\widetilde{\mathcal{J}}_{\mathrm{RLHF}}(\theta) = \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x)\Big[\beta_0 \log \frac{\pi_{\mathrm{dpo}}(y_t|y_{<t},x)}{\pi_{\mathrm{ref}}(y_t|y_{<t},x)} - \beta \log \frac{\pi_\theta(y_t|y_{<t},x)}{\pi_{\mathrm{ref}}(y_t|y_{<t},x)}\Big]$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x)\Big[\frac{\beta_0}{\beta} \log \pi_{\mathrm{dpo}}(y_t|y_{<t},x)$$

$$+ (1-\frac{\beta_0}{\beta}) \log \pi_{\mathrm{ref}}(y_t|y_{<t},x) - \log \pi_\theta(y_t|y_{<t},x)\Big] \tag{30}$$

Here we introduce an equivalence between log probabilities and logits, *i.e.*, when

$$p(i) = \frac{\mathrm{e}^{z_i}}{\sum_{j=1}^{|\mathcal{V}|} \mathrm{e}^{z_j}}, \tag{31}$$

we have

$$\log p(i) = z_i - \log \sum_{j=1}^{|\mathcal{V}|} \mathrm{e}^{z_j}. \tag{32}$$

Then, we substitute Eq. (32) into Eq. (30):

$$\widetilde{\mathcal{J}}_{\mathrm{RLHF}}(\theta) = \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x)\Big[\frac{\beta_0}{\beta} \log \pi_{\mathrm{dpo}}(y_t|y_{<t},x)$$

$$+ (1-\frac{\beta_0}{\beta}) \log \pi_{\mathrm{ref}}(y_t|y_{<t},x) - \log \pi_\theta(y_t|y_{<t},x)\Big] \tag{33}$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x)\Big[\frac{\beta_0}{\beta} z_t^{\mathrm{dpo}} + (1-\frac{\beta_0}{\beta}) z_t^{\mathrm{ref}} + Z - \log \pi_\theta(y_t|y_{<t},x)\Big] \tag{34}$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x)\Big[z_t^* - \log \pi_\theta(y_t|y_{<t},x)\Big], \tag{35}$$

where $z^* = \frac{\beta_0}{\beta} z_t^{\mathrm{dpo}} + (1-\frac{\beta_0}{\beta}) z_t^{\mathrm{ref}}$, and Z is a constant representing the logsumexp term in Eq. (32). Thus, it has no influence on the expectation and can be omitted in the later calculation. Then we leverage the equivalence again and convert the logits back to log probabilities:

$$\widetilde{\mathcal{J}}_{\mathrm{RLHF}}(\theta) = \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x)\Big[z_t^* - \log \pi_\theta(y_t|y_{<t},x)\Big]$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x)\Big[\log \pi^*(y_t|y_{<t},x) - \log \pi_\theta(y_t|y_{<t},x)\Big] \tag{36}$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x) \log \frac{\pi^*(y_t|y_{<t},x)}{\pi_\theta(y_t|y_{<t},x)} \tag{37}$$

$$= \min_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \mathbb{D}_{\mathrm{KL}}(\pi_\theta(y_t|x,y_{<t})||\pi^*(y_t|x,y_{<t})). \tag{38}$$

Thus, the theorem is proved. □

B Derivations for Changed Logit Distribution

As shown in Eq. (13), we rewrite the objective of RLHF under the contrastive DPO reward as follows:

$$\widetilde{\mathcal{J}}_{\mathrm{RLHF}}(\theta) = \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \Big[\beta_0 \log \frac{\pi_{\mathrm{dpo}}(y|x)}{\pi^{-}_{\mathrm{dpo}}(y|x)} - \beta \log \frac{\pi_\theta(y|x)}{\pi_{\mathrm{dpo}}(y|x)}\Big]. \tag{39}$$

Correspondingly, the token-level objective becomes

$$\widetilde{\mathcal{J}}_{\mathrm{RLHF}}(\theta) = \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x) \Big[\beta_0 \log \frac{\pi_{\mathrm{dpo}}(y_t|y_{<t},x)}{\pi^{-}_{\mathrm{dpo}}(y_t|y_{<t},x)} - \beta \log \frac{\pi_\theta(y_t|y_{<t},x)}{\pi_{\mathrm{dpo}}(y_t|y_{<t},x)}\Big] \tag{40}$$

$$= \max_{\theta} \mathop{\mathbb{E}}_{\substack{x\sim D \\ y\sim\pi_\theta(\cdot|x)}} \beta \sum_{t=1}^{|y|} \sum_{y_t\in\mathcal{V}} \pi_\theta(y_t|y_{<t},x) \Big[(1+\frac{\beta_0}{\beta}) \log \pi_{\mathrm{dpo}}(y_t|y_{<t},x) - \frac{\beta_0}{\beta} \log \pi^{-}_{\mathrm{dpo}}(y_t|y_{<t},x) - \log \pi_\theta(y_t|y_{<t},x)\Big]. \tag{41}$$

The following derivation is similar to the one after Eq. (30) and thus omitted.

C Implementation Details

In this section, we provide the implementation details of baseline methods and our AlignDistil. All our implementation is based on the open-source toolkit OpenRLHF[5]. Below we list the individual settings for each method in our experiments:

- **DPO** (default setting): we set $\beta = 0.1$ and optimize the model on UltraFeedback;
- **DPO**$_{\beta=0.01}$: the only difference compared to DPO (default setting) is $\beta = 0.01$;
- **KTO**: we set $\beta = 0.1$ and use the unpaired version of UltraFeedback for training;
- **TDPO**$_1$: similar to DPO in the default setting, we set $\beta = 0.1$;
- **TDPO**$_2$: TDPO$_2$ introduces a new hyper-parameter α to control the intensity of KL term and we find that $\alpha = 0.1$ works best in our experiments;
- **SimPO**: SimPO involves two hyper-parameters, *i.e.*, β and a ratio $\frac{\gamma}{\beta}$, which are needed fine-grained tuning to achieve ideal performance. Specifically, for both models, we set $\beta = 10$ and $\frac{\gamma}{\beta} = 0.5$ after extensive tuning.
- **PPO**: Before PPO, we first train a reward model on UltraFeedback based on Qwen2.5-1.5B-Instruct. We use the same reward model for both initial models since we find the reward model based on Qwen2-1.5B-Instruct leads to unstable PPO optimization. Afterward, we mainly follow the suggested settings in OpenRLHF for PPO training, *e.g.*, setting the critic learning rate to 9e-6, rollout batch size to 1024, and the KL coefficient to 0.01.
- **RTO**: The procedure of RTO is similar to PPO, except for the token-level DPO reward $\beta \log \frac{\pi_{\mathrm{dpo}}(y_t|y_{<t},x)}{\pi_{\mathrm{ref}}(y_t|y_{<t},x)}$. We use the DPO model in the default setting with $\beta = 0.1$ to calculate the DPO reward. Besides, RTO set β_2 as the KL coefficient in PPO. In our experiment, we find RTO is sensitive to β_2 and tends to produce overly long responses. Thus, we set $\beta_2 = 0.05$ as an appropriate value for stable training. After our experiments, the authors of RTO update their methods to fix this issue in the latest (v3) version of the paper (Zhong et al., 2024). Despite better performance, this update is a concurrent work with ours and our implementation of RTO is still based on the v2 version of the paper.

[5] https://github.com/OpenRLHF/OpenRLHF

- **On-Policy AlignDistil**: The on-policy AlignDistil uses the DPO model in the default setting as well as a reverse DPO model trained by switching the chosen/rejected responses in DPO training. For on-policy AlignDistil, we only use the prompts in Ultrafeedback and sample responses from the current policy. The hyper-parameter r for token adaptive logit extrapolation is set to 20 for Qwen2-1.5B-Instruct and 15 for Qwen2.5-1.5B-Instruct.
- **Off-Policy AlignDistil**: For off-policy AlignDistil, we use both the prompts and the chosen responses in Ultrafeedback for training. The hyper-parameter r is set to 10 for Qwen2-1.5B-Instruct and 15 for Qwen2.5-1.5B-Instruct.

D Performance for Qwen2.5-72B-Instruct as Judge

Qwen2.5-72B-Instruct has been demonstrated as a strong open-source model with comparable performance against the state-of-the-art LLMs. Following the tools for evaluating LLM-as-Judge provided in the repository[6] of (Dubois et al., 2024), we test the evaluation performance for Qwen2.5-72B-Instruct and list the performance in Table 4.

Evaluators	Human Agreement	Price [$/1000 examples]	Spearman corr.	Pearson corr.
alpaca_eval_gpt4	69.17	13.60	0.97	0.93
alpaca_eval_gpt4_turbo_fn	68.09	5.53	0.93	0.82
Qwen2.5-72B-Instruct	67.63	0	0.92	0.86
weighted_alpaca_eval_gpt4_turbo	65.73	4.32	0.78	0.77
humans	65.66	300	1.00	1.00

Table 4: Comparisons of Qwen2.5-72B-Instruct and some top evaluators on the AlpacaEval leaderboard in terms of performance and cost. We select several key columns from the leaderboard.

As shown in Table 4, Qwen2.5-72B-Instruct achieves comparable human agreement with `alpaca_eval_gpt4_turbo_fn` and `alpaca_eval_gpt4` with a much lower price since we can deploy the model with vLLM locally. Moreover, compared to the official recommended evaluator `weighted_alpaca_eval_gpt4_turbo`, Qwen2.5-72B-Instruct performs significantly better on both performance and cost. Thus, we choose Qwen2.5-72B-Instruct as the evaluator for the three benchmarks.

E Implementation Details for Convergence Speed Comparison

To evaluate the convergence speed of our AlignDistil, we use two methods that optimize sentence-level (response-level) rewards and token-level scalar-type rewards, respectively. For sentence-level optimization, we use the contrastive DPO reward on the whole sequence and calculate the gradient of the policy model as follows:

$$\nabla_\theta \mathcal{J}(\theta) = \frac{1}{|y|}\Big[\sum_{t=1}^{|y|} r_{\mathrm{ctr}}(x,y)\nabla_\theta \log \pi_\theta(y_t|y_{<t},x) - \beta \nabla_\theta \log \frac{\pi_\theta(y|x)}{\pi_{\mathrm{dpo}}(y|x)}\Big]. \tag{42}$$

For token-level optimization with scalar-type rewards, we optimize token-level contrastive reward with a REINFORCE algorithm:

$$\nabla_\theta \mathcal{J}(\theta) = \frac{1}{|y|}\sum_{t=1}^{|y|}\Big[G_t \nabla_\theta \log \pi_\theta(y_t|y_{<t},x) - \beta \nabla_\theta \mathbb{D}_{\mathrm{KL}}(\pi_\theta(\cdot|y_{<t},x)||\pi_{\mathrm{dpo}}(\cdot|y_{<t},x))\Big], \tag{43}$$

where $G_t = \sum_{i=t}^{|y|} r_{\mathrm{ctr}}(x, y_{<i}, y_i)$ is the return at position t.

[6] https://github.com/tatsu-lab/alpaca_eval