ADVANTAGE-GUIDED DISTILLATION FOR PREFERENCE ALIGNMENT IN SMALL LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Alignment techniques such as RLHF enable LLMs to generate outputs that align with human preferences and play an essential role in their effectiveness. However, their impact often diminishes when applied to smaller language models, likely due to the limited capacity of these models. Instead of directly applying existing alignment techniques to smaller models, we propose to utilize a well-aligned teacher LLM to guide the alignment process for these models, thereby facilitating the transfer of the teacher's knowledge of human preferences to the student model. To achieve this, we first explore a straightforward approach, Dual-Constrained Knowledge Distillation (DCKD), that employs knowledge distillation with two KL-divergence constraints from the aligned teacher to the unaligned student. To further enhance the contrastive effect, we then propose Advantage-Guided Distillation for Preference Alignment (ADPA), which leverages an advantage function from the aligned teacher to deliver more nuanced, distribution-level reward signals for the student's alignment. Our experimental results demonstrate that these two approaches appreciably improve the alignment of smaller language models and narrow the performance gap with their larger counterparts.

1 INTRODUCTION

Large Language Models (LLMs) have been effectively aligned with human preferences to generate helpful, truthful, and harmless responses through techniques like Reinforcement Learning from Human Feedback (RLHF) (Kaplan et al., 2020; Ouyang et al., 2022; Askell et al., 2021). However, deploying such large models in resource-constrained environments can be challenging due to their heavy computational and memory demands. While smaller language models are more suited for these scenarios, they often struggle to achieve the same level of alignment as larger LLMs. These small models may experience an "alignment tax", where their overall performance across various tasks declines after RLHF training (Bai et al., 2022). This decline is likely due to their limited capacity to capture the complexities of diverse tasks and nuanced human feedback, which can result in overfitting and poor generalization (Kirk et al., 2024; Zhao et al., 2023a). Moreover, traditional RLHF methods rely on sequence-level rewards that are sparse and coarse-grained (Sun, 2023; Chan et al., 2024), making optimization more challenging for smaller LLMs.

To enhance the alignment of smaller models with human preferences and achieve an "alignment bonus", a promising strategy is to leverage preference-aligned larger models to guide smaller models through knowledge distillation (KD) (Hinton, 2015). KD enables the student model to learn from the teacher's output distributions, which contain nuanced learning signals (Gu et al., 2024), to effectively transfer knowledge from teacher to student. However, existing KD methods primarily focus on the pre-training and instruction-tuning stages (Song et al., 2020; Khanuja et al., 2021) and often overlook the critical phase of preference alignment. This oversight prevents student models from capturing the teacher's alignment knowledge with human preferences. Moreover, most KD techniques emphasize positive signals from the teacher's outputs on ground-truth responses while neglecting negative signals from suboptimal outputs, which limits the overall alignment effect. Fortunately, these issues have recently garnered attention from the community. For instance, DPKD (Li et al., 2024) and PLaD (Zhang et al., 2024) treat the teacher's outputs as preferred responses and the student's outputs as dispreferred and carry out preference learning to train the student model.

In this work, we first explore a straightforward approach, Dual-Constrained Knowledge Distillation (DCKD) for preference alignment, which facilitates knowledge distillation from the aligned teacher to the unaligned student using preference training data. To integrate both positive and negative signals, we introduce an additional KL-divergence constraint term for dispreferred responses into the traditional knowledge distillation objective. This enables the student model to learn the teacher's predictive behaviors for both preferred and dispreferred responses. While this method enables direct transfer of preference knowledge from teacher to student, its effect could be limited by the lack of a contrastive mechanism to differentiate between preferred and dispreferred responses.

To overcome this limitation, we propose another approach that introduces stronger contrastive signals by incorporating a fine-grained preference alignment mechanism into the distillation process, allowing the teacher model to guide the student model during RLHF training. Specifically, we introduce Advantage-Guided Distillation for Preference Alignment (ADPA) that utilizes an advantage function derived from a teacher model trained with Direct Preference Optimization (DPO) (Rafailov et al., 2024b) and a pre-DPO reference model. The advantage function delivers distribution-level reward signals and allows the student model to optimize its policy based on fine-grained preferences and expected future rewards, which tackles the issue of sparse reward signals present in traditional RLHF. As illustrated in Figure 1, integrating preference alignment into knowledge distillation allows smaller models to better capture human preferences than directly applying DPO, ultimately reducing the performance gap between small and large language models.

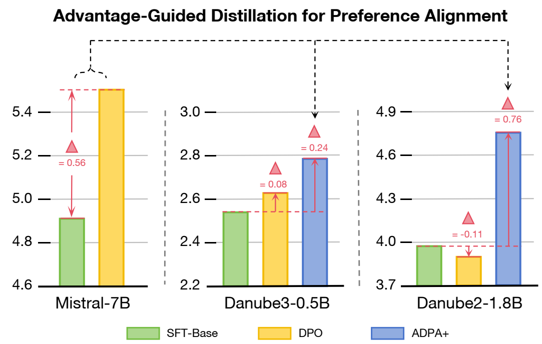

Figure 1: Results illustrating the "alignment tax" of small models and the effect of our ADPA method in relieving this issue. With DPO training, the larger model (Mistral-7B) shows a notable improvement on MT-Bench (+0.56), while smaller models (Danube3-0.5B and Danube2-1.8B) exhibit only a modest gain (+0.08) or even a decline. In contrast, ADPA enables the smaller models to achieve a larger increase on MT-Bench compared to DPO (+0.24 vs. +0.08 for Danube3-0.5B, and +0.76 vs. -0.11 for Danube2-1.8B).

The major contributions of this work can be summarized as follows:

- We investigate the alignment challenge for small language models through knowledge distillation from a preference-aligned teacher model to a smaller student model. We present Dual-Constrained Knowledge Distillation (DCKD) as a straightforward baseline, highlighting its benefits and limitations for the preference alignment of smaller models.
- We propose Advantage-Guided Distillation for Preference Alignment (ADPA), which uses an advantage function from a preference-aligned teacher model to provide distribution-level reward signals and expected future rewards for optimizing the student model.
- We conduct extensive experiments to demonstrate the effectiveness of our proposed approaches and provide valuable insights for future research in the preference alignment of small language models. Specifically, leveraging preference-aligned larger models to guide the alignment training of smaller language models holds promise for overcoming their limited capacity.

2 Related Work

Knowledge Distillation Knowledge Distillation (KD) (Hinton, 2015) is a widely used model compression technique in which a smaller student model learns to replicate the behavior of one or more larger teacher models. In the context of LLMs, KD typically involves reducing the Kullback-Leibler Divergence (KLD) between the output distributions of the student and the teacher models at each time step. Recent research has introduced several optimizations aimed at enhancing this process. For instance, MiniLLM (Gu et al., 2024) employs sequence-level reverse KLD to encourage the student model to focus on the most significant modes of the teacher's output distribution. DistiLLM (Ko et al., 2024), on the other hand, increases the efficiency of the distillation process by using asymmetric KLD (Skew-KLD) combined with adaptive off-policy methods. Likewise, f-DISTILL

(Wen et al., 2023) minimizes a symmetric f-divergence to mitigate challenges such as mode collapse, while Adaptive KL (AKL) (Wu et al., 2024) balances forward and reverse KLD to ensure the student model effectively learns across different parts of the distribution. Other approaches, including Vicuna (Chiang et al., 2023) and MCC-KD (Chen et al., 2023), take advantage of sequences generated by the teacher model to train the student, thereby enhancing its ability to follow instructions or perform more complex reasoning tasks, such as Chain-of-Thought (CoT) reasoning.

Preference Alignment Preference alignment aims to align the outputs of LLMs with human preferences and values. This objective is traditionally achieved by Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022), which relies on a reward model (RM) trained on preference data to guide the optimization of the policy model through methods like proximal policy optimization (PPO) (Schulman et al., 2017). Recent research has increasingly focused on using contrastive learning methods to eliminate the need to construct a reward model (RM) and to simplify the complexities of reinforcement learning. Notable approaches in this area include Direct Preference Optimization (DPO) (Rafailov et al., 2024b) and SLic-HF (Zhao et al., 2023b). In addition to these methods that utilize sequence-level rewards, other studies explore fine-grained rewards to provide more detailed guidance to the policy model. For example, Yang et al. (2024) define trajectory-wise rewards as aggregations of individual token-wise rewards learned through standard preference-based RM training. Similarly, Token-Level Continuous Reward (TLCR) (Yoon et al., 2024) utilizes GPT-4 as a reviser on preference pairs to generate token-level preference labels, which are then used to train a discriminator capable of assigning rewards at the token level.

Given the high cost of obtaining quality preference labels for training reward models, recent research has shifted towards leveraging larger and more powerful LLMs to provide feedback on the preferences of candidate responses. For instance, RLAIF (Lee et al., 2023) utilizes an off-the-shelf LLM to provide feedback for candidate responses, which are then used to train a reward model for reinforcement learning. Zephyr (Tunstall et al., 2023) and Starling (Zhu et al., 2024) collect responses from multiple LLMs and rank them using GPT-4 to obtain preference data. While the former employs this data to train the policy using DPO, the latter uses it to train a reward model for reinforcement learning. Other approaches, such as DPKD (Li et al., 2024) and PLaD (Zhang et al., 2024) treat the teacher's outputs as preferred responses and the student's outputs as dispreferred responses and conduct preference learning. RLCD (Yang et al., 2023) constructs positive and negative prompts to elicit corresponding responses, designating these as preferred and dispreferred, respectively, and then uses this preference data to train a reward model for reinforcement learning. Reward Model Distillation (RMD) (Fisch et al., 2024) aligns the reward margin predicted by the policy with that of a reward model trained on preference data to enhance the robustness of DPO training.

3 Methodology

In this section, we introduce the proposed Dual-Constrained Knowledge Distillation (DCKD) and Advantage-Guided Distillation for Preference Alignment (ADPA) approaches in detail. We start with an overview of the preliminaries of knowledge distillation and preference alignment in LLMs, followed by a detailed explanation of the DCKD and ADPA methods.

3.1 Preliminaries

Knowledge Distillation Given a dataset of prompt-response pairs (x, y), a teacher LLM π_t, and a smaller student model π_s, the goal of knowledge distillation (KD) is to enable the student model to mimic the predictions of the teacher as effectively as possible. Specifically, there are typical two loss terms to minimize in the objective function. First, the supervised fine-tuning (SFT) term computes a negative log-likelihood (NLL) loss for the student model to predict the next token y_t in the response conditioned on the prompt x and the previous response tokens $y_{<t}$. Second, the Kullback-Leibler Divergence (KLD) between the output distributions of the teacher and the student is calculated. These two terms are combined using a weighted sum:

$$\mathcal{L}_{\text{KD}} = -\sum_{t=1}^{|y|} \left(\log \pi_s(y_t \mid x, y_{<t}) + \alpha D_{\text{KL}} \left(\pi_t(\cdot \mid x, y_{<t}) \mid\mid \pi_s(\cdot \mid x, y_{<t})\right)\right). \tag{1}$$

Preference Alignment for LLMs Preference alignment methods such as RLHF (Ouyang et al., 2022) optimize LLMs to produce outputs that align with human preferences. Given a preference dataset $\mathcal{D}$ containing a set of tripples, each consisting of a prompt x, a preferred response y_w, and a dispreferred response y_l, a sequence-level reward model (RM) can be trained as follows:

$$\mathcal{L}_{\text{RM}} = \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log \sigma \left(\text{RM}_\theta(x, y_w) - \text{RM}_\theta(x, y_l)\right)\right], \tag{2}$$

where σ is the sigmoid function. After training the RM, classical RLHF methods typically optimize the SFT-trained LLMs using policy gradient techniques, such as PPO (Schulman et al., 2017). Formally, the objective is to maximize the sequence-level reward assigned by the RM while penalizing deviations from the reference policy using a KLD term, weighted by a coefficient β:

$$\max_\theta \mathbb{E}_{y\sim\pi_\theta(\cdot|x)} \left[\text{RM}(x, y) - \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)}\right], \tag{3}$$

where π_{ref} denotes the reference policy. Offline RLHF methods like DPO (Rafailov et al., 2024b) optimize the policy model directly using the Bradley-Terry preference model (Bradley & Terry, 1952) without requiring an external reward model and online Reinforcement Learning (RL) training:

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

3.2 Dual-Constrained Knowledge Distillation

A straightforward approach for transferring preference knowledge from large models to smaller ones is to perform knowledge distillation with preference data. Specifically, Dual-Constrained Knowledge Distillation (DCKD) begins by fine-tuning the teacher model on preference data through Direct Preference Optimization (DPO). Then, the distillation process minimizes the divergence between the output distributions of the teacher and student models for both preferred and dispreferred responses.

Formally, we define a pair of responses as (y_w, y_l), where y_w denotes the preferred response and y_l indicates the dispreferred response. Let π_{dpo} represent the teacher policy trained with DPO. We then formulate two KL-divergence constraints using y_w and y_l as:

$$\mathcal{L}_{\text{KLD-}w}(\pi_{\text{dpo}}, \pi_\theta) = \mathbb{E}_{(x,y_w)\sim\mathcal{D}} \left[\sum_{t=1}^{|y_w|} D_{\text{KL}} \left(\pi_{\text{dpo}}(\cdot \mid x, y_{w,1:t-1}) \,\|\, \pi_\theta(\cdot \mid x, y_{w,1:t-1})\right)\right] \tag{5}$$

$$\mathcal{L}_{\text{KLD-}l}(\pi_{\text{dpo}}, \pi_\theta) = \mathbb{E}_{(x,y_l)\sim\mathcal{D}} \left[\sum_{t=1}^{|y_l|} D_{\text{KL}} \left(\pi_{\text{dpo}}(\cdot \mid x, y_{l,1:t-1}) \,\|\, \pi_\theta(\cdot \mid x, y_{l,1:t-1})\right)\right]. \tag{6}$$

With the supervised fine-tuning (SFT) term, the overall objective of DCKD is formulated as:

$$\mathcal{L}_{\text{DCKD}} = \mathcal{L}_{\text{SFT}} + \alpha \left(\mathcal{L}_{\text{KLD-}w} + \mathcal{L}_{\text{KLD-}l}\right) \tag{7}$$

There are two key differences between DCKD and traditional knowledge distillation approaches. First, DCKD distills knowledge from a teacher model fine-tuned with DPO, which encodes richer preference information compared to traditional teacher models. Second, instead of minimizing the KL-divergence solely on preferred responses, DCKD minimizes it for both preferred and dispreferred responses, thus enabling the student model to align more effectively with human preferences.

3.3 Advantage-Guided Distillation for Preference Alignment

While DCKD enables direct transfer of preference knowledge from the teacher to the student, it may not effectively emphasize the differences between preferred and dispreferred responses. Therefore, we propose Advantage-Guided Distillation for Preference Alignment (ADPA), which utilizes an advantage function derived from a teacher model trained with Direct Preference Optimization (DPO) (Rafailov et al., 2024b) and a pre-DPO reference model. The sign of the advantage function explicitly distinguishes positive and negative actions at the distribution level, providing stronger guidance for the student model to distinguish between positive and negative actions and learn fine-grained preferences. Our experimental results demonstrate that this approach appreciably improves the alignment of smaller models and reduces the performance gap with larger LLMs.

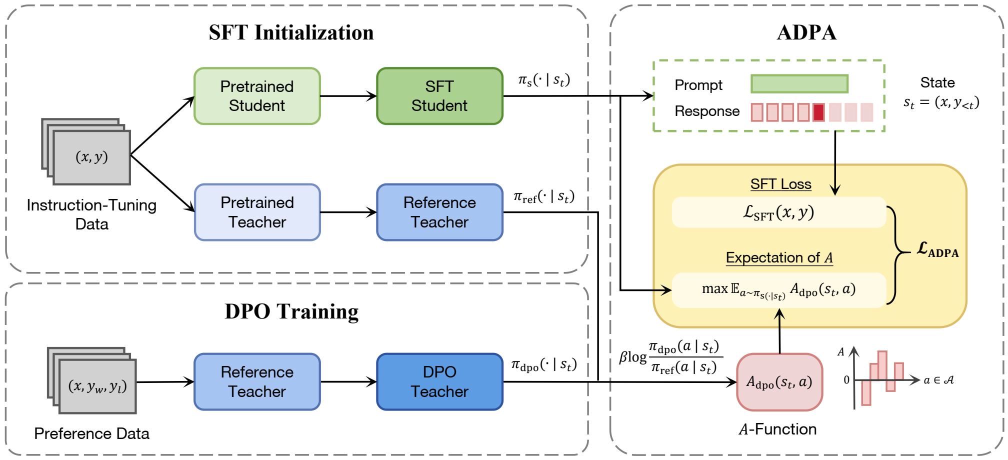

Figure 2: Overview of the ADPA approach. The training process involves two teacher models: a reference teacher π_{ref}, fine-tuned on instruction-tuning data, and a DPO teacher π_{dpo}, fine-tuned using DPO on preference data. The student model is trained by fine-tuning on the instruction-tuning data and performing advantage-guided distillation using on-policy sampled data.

Deriving the Advantage Function Consider an SFT-trained teacher model π_{ref} and a DPO-trained teacher model π_{dpo} that is initialized from π_{ref}. The DPO process aims to maximize the expected reward of the teacher model based on human preferences. We first define the Q-function that quantifies the benefit of π_{dpo} through preference alignment at each time step. The token generation of LLMs is modeled as a Markov Decision Process (MDP), where each generation corresponds to an action (token) a_t selected from the action set (vocabulary) $\mathcal{A}$, and the current state s_t comprises the prompt x along with all previously generated tokens $y_{<t}$. The trajectory $\tau = \{(s_t, a_t)\}_{t=1}^{|\tau|}$ denotes the generation sequence, with $|\tau|$ indicating its length and $a_{|\tau|} =$ EOS. Traditionally, a sequence-level reward is produced by DPO at the final time step.

Formally, we define the Q-function as:

$$Q_{\text{dpo}}(s_t, a) = \beta \left[\sum_{i=1}^{t-1} \log \frac{\pi_{\text{dpo}}(a_i \mid s_i)}{\pi_{\text{ref}}(a_i \mid s_i)} + \log \frac{\pi_{\text{dpo}}(a \mid s_t)}{\pi_{\text{ref}}(a \mid s_t)} \right], \tag{8}$$

where β is a scaling parameter. $Q_{\text{dpo}}(s_t, a)$ captures the expected cumulative reward starting from state s_t and taking action a, using the relative probabilities between π_{dpo} and π_{ref}.

The advantage function (A-function) aims to measure the relative preference of each possible action in $\mathcal{A}$ at a given state s_t and is derived from the Q-function as follows:

$$A_{\text{dpo}}(s_t, a) = \beta \log \frac{\pi_{\text{dpo}}(a \mid s_t)}{\pi_{\text{ref}}(a \mid s_t)}. \tag{9}$$

Refer to Appendix A for a detailed derivation process of the Q-function and A-function.

A-function provides fine-grained, distribution-level reward signals that capture the relative preference of an action a at state s_t, as determined by the DPO-trained teacher model in comparison to the reference model. It guides the student model during training by quantifying the preference for each possible action at every state. This granular feedback enables the student model to align more closely with human preferences, helping capture subtle nuances and improve overall performance.

ADPA Training Objective In ADPA, we integrate the A-Function into the training objective. Specifically, we maximize the expectation of A-Function for the student policy model:

$$\max \mathbb{E}_{a\sim\pi_s(\cdot\mid s_t)} A_{\text{dpo}}(s_t, a) = \max \mathbb{E}_{a\sim\pi_s(\cdot\mid s_t)} \log \frac{\pi_{\text{dpo}}(a \mid s_t)}{\pi_{\text{ref}}(a \mid s_t)} \tag{10}$$

The overall ADPA loss function is defined as:

$$\mathcal{L}_{\text{ADPA}} = \mathbb{E}_{(x,y,\hat{y})} \left[\mathcal{L}_s(x,y) - \gamma \sum_{t=1}^{|\hat{y}|} \sum_{a \in \mathcal{A}} \pi_s(a \mid x, \hat{y}_{<t}) \log \frac{\pi_{\text{dpo}}(a \mid x, \hat{y}_{<t})}{\pi_{\text{ref}}(a \mid x, \hat{y}_{<t})} \right] \tag{11}$$

where $\mathcal{L}_{\text{SFT}}(x,y)$ is the supervised fine-tuning (SFT) loss, which ensures the student model retains its ability in the current domain (Hong et al., 2024) and prevent over-optimization (Liu et al., 2024b). γ is a scaling factor to balance the SFT loss and the advantage-guided distillation loss. $\hat{y}$ is the sequence generated by the initial student model for prompt x before the training process. π_s is the student policy model. The overall ADPA pipeline is shown in the Appendix Algorithm 1.

4 Experiment

4.1 Experiment Setup

Training Details In our experiments, we evaluate preference alignment using three small language models: H2O-Danube3-500M (Pfeiffer et al., 2024), H2O-Danube2-1.8B-Base (Singer et al., 2024), and LLaMA-2-7B. For H2O-Danube3-500M and H2O-Danube2-1.8B-Base, we use Mistral-7B-V0.1 (Jiang et al., 2023) as the teacher model. For LLaMA-2-7B, we use Llama-2-13B (Touvron et al., 2023) to serve as the teacher model. We begin by performing Supervised Fine-Tuning (SFT) on both student and teacher models using an instruction-tuning dataset, training for 3 epochs with a learning rate of 2e-5 and a batch size of 128. Next, we apply DPO to the fine-tuned teacher to create the DPO teacher model, using $\beta = 0.05$, a reduced learning rate of 5e-7 and the same batch size of 128. During the KD phase, we follow the context distillation method from Bai et al. (2022), pre-computing the teacher's logits on the preference dataset and saving the top 50 tokens by probability, along with the summed probability for the remaining tokens. In the DCKD phase, we experiment with α in $[0.1, 0.2, 0.5, 1, 2, 5]$ and γ in $[0.5, 1, 1.5, 2, 3, 5]$. For the ADPA phase, we pre-compute $\log P_{\text{DPO}} - \log P_{\text{Ref}}$ for the stored probabilities. Tokens in the DPO teacher's top 50 but absent from the reference teacher's have their log probabilities adjusted by subtracting the lowest probability from the reference's top 50. Tokens in the reference's top 50 but not in the teachers' are omitted.

Datasets For SFT, we use the Deita-10K-V0 (Liu et al., 2024a) dataset, which contains 10k high-quality instruction-response pairs. This dataset is utilized to train both the teacher and student models. For preference alignment, we draw upon two distinct datasets. The first is DPO-MIX-7K [1], a meticulously curated collection of high-quality pairwise comparison data sourced from existing datasets. The second dataset is HelpSteer2 (Wang et al., 2024), which is developed to align models for enhanced helpfulness. In our application of HelpSteer2, we differentiate between positive and negative samples based on the helpfulness metric and exclude samples with identical scores.

Validation We employ FsfairX-LLaMA3-RM-V0.1 (Dong et al., 2024; Xiong et al., 2024), a high-performing reward model from Reward Bench (Lambert et al., 2024), to evaluate and determine the optimal checkpoints during the training process. This reward model generates an average score for responses produced based on prompts derived from the validation subset of our preference dataset.

Evaluation We assess the models' performance using two benchmarks: MT-Bench (Zheng et al., 2023) and AlpacaEval (Li et al., 2023). For MT-Bench, we utilize GPT-4-0125-Preview as the evaluator, in accordance with recent recommendations [2], to rectify any inaccuracies in the reference answers originally provided by GPT-4. For AlpacaEval, while the standard protocol involves comparing responses against GPT-4, this approach can be overly demanding for smaller models, potentially leading to low win rates and high variability when comparing knowledge distillation (KD) methods. Given that our primary objective is to ascertain whether alternative methods can achieve or exceed the performance of ADPA, we employ student models trained with ADPA as reference models, thereby enabling direct performance comparisons. To calculate win rates on the test questions from AlpacaEval, we adhere to the default setup, utilizing GPT-4-1106-Preview as the evaluator.

4.2 Main Results

We compare DCKD, ADPA and ADPA+ with two basic methods, SFT and DPO (Rafailov et al., 2024b), and sevreal state-of-the-art knowledge distillation and preference alignment baselines, in-

[1] https://huggingface.co/datasets/argilla/dpo-mix-7k

[2] https://github.com/lm-sys/FastChat/pull/3158

Table 1: Overall results of our methods using Daunbe3-0.5B, Daunbe2-1.8B, and LLaMA-2-7B as the student models. We show the Win Rate (WR) and Length Control Win Rate (LC WR) against ADPA-trained student models on AlpacaEval (AE), and the average score on MT-Bench.

Model	**Method**	**DPO-MIX-7K**			**HelpSteer2**		
		AE WR(%)	**AE LC WR(%)**	**MT-Bench**	**AE WR(%)**	**AE LC WR(%)**	**MT-Bench**
Daunbe3 0.5B	Teacher	85.2	84.8	5.90	93.9	93.2	5.59
	Student	34.4	34.7	2.54	38.0	38.3	2.54
	SFT	37.1	38.4	2.51	32.4	34.0	2.29
	DPO	35.1	35.3	2.62	36.1	36.4	2.52
	VanillaKD	37.0	37.5	2.60	36.2	37.0	2.28
	SeqKD	39.4	39.3	2.53	41.7	41.6	2.46
	ATKD	38.0	38.5	2.64	35.5	36.3	2.50
	PLAD	35.1	35.3	2.64	38.0	38.4	2.58
	DDPO	37.3	37.4	2.67	37.0	37.3	2.58
	DPKD	34.3	34.6	2.66	36.3	36.9	2.51
	DCKD	38.9	39.2	2.77	34.2	35.3	2.50
	ADPA	**50.0**	**50.0**	2.67	50.0	50.0	2.70
	ADPA+	49.0	48.3	**2.78**	**53.2**	**53.0**	**2.76**
Daunbe2 1.8B	Teacher	61.1	68.8	5.90	82.5	83.7	5.59
	Student	28.6	29.1	3.98	39.5	39.9	3.98
	SFT	29.1	29.6	3.91	40.4	40.3	4.09
	DPO	31.4	30.6	3.87	40.3	40.7	3.87
	VanillaKD	28.3	28.6	4.01	46.3	46.9	4.03
	SeqKD	32.8	33.4	4.18	42.3	41.9	4.10
	ATKD	29.8	30.0	4.10	42.9	42.8	3.93
	PLAD	29.1	29.7	4.06	44.4	40.1	3.84
	DDPO	31.7	33.6	4.06	39.2	39.6	3.68
	DPKD	38.7	40.1	4.42	43.2	43.1	3.97
	DCKD	34.2	34.6	4.29	51.1	51.3	4.03
	ADPA	50.0	50.0	4.33	50.0	50.0	4.02
	ADPA+	**61.0**	**61.3**	**4.74**	**62.7**	**62.4**	**4.33**
LLaMA-2 7B	Teacher	42.6	50.2	5.74	71.3	74.6	5.43
	Student	21.5	22.6	4.34	24.0	24.9	4.34
	SFT	21.6	21.8	4.70	35.7	35.9	4.30
	DPO	28.7	33.5	4.49	38.6	39.6	4.51
	VanillaKD	29.5	28.0	4.75	35.3	35.6	4.60
	SeqKD	25.0	27.9	4.74	28.6	29.3	4.47
	ATKD	24.1	24.7	4.68	32.0	32.7	4.43
	PLaD	21.7	22.8	4.24	28.0	28.6	4.35
	DDPO	21.7	23.0	4.67	30.4	31.0	3.78
	DPKD	22.3	23.4	4.40	28.7	27.6	3.97
	DCKD	32.5	34.5	4.80	39.1	38.3	4.41
	ADPA	50.0	50.0	5.29	50.0	50.0	4.40
	ADPA+	**60.6**	**59.6**	**5.42**	**60.1**	**59.1**	**4.86**

cluding KD (Hinton, 2015), SeqKD (Kim & Rush, 2016), ATKD (Zhong et al., 2024b), PLAD (Zhang et al., 2024), DDPO (Fisch et al., 2024) and DPKD (Li et al., 2024). Here, ADPA+ leverages the DCKD model to initialize the learning process of ADPA, incorporating the $\hat{y}$ generated by the DCKD model into training, as shown in the Appendix Algorithm 2. Additionally, for DPKD and PLAD, we use actual preference data as positive and negative samples, rather than pseudo pairs, to ensure fairness.

In Table 4.1, we present the comparative results across both preference datasets. Several key observations emerge from these findings. **First**, our proposed methods, DCKD and ADPA, consistently outperform baseline approaches, demonstrating the effectiveness of our dual-constrained distillation and advantage-guided approaches. For example, on a smaller model like Danube2-1.8B, DCKD and ADPA achieve 10.8% and 11.9% improvements over DPO in MT-Bench on DPO-MIX-7K, indicating that the preference-aligned teacher model can more effectively guide the student in aligning its output with human preferences. **Second**, when ADPA is used as the reference model for AlpacaEval, existing distillation and preference alignment methods achieve a win rate below 50%, validating the strength of preference-based distillation and emphasizing the value of preference signal-based distillation. **Lastly**, initializing ADPA with a student model from DCKD, as in ADPA+, results in significantly superior performance compared to either method alone. This combination allows the student model to better capture the teacher's output distribution while effectively learning nu-

Table 2: Results of model ablation on DCKD and ADPA with different teacher-student setups on DPO-MIX-7K dataset.

Method	Mistral-7B → Danube3-0.5B		Mistral-7B → Danube2-1.8B		LLaMA-2-13B → LLaMA-2-7B	
	AlpacaEval	**MT-Bench**	**AlpacaEval**	**MT-Bench**	**AlpacaEval**	**MT-Bench**
DCKD	**50.0**	**2.77**	**50.0**	**4.29**	**50.0**	**4.80**
- w/o DPO Teacher	48.2	2.55	35.6	3.83	39.1	4.55
- w/o dispreferred response	40.3	2.57	39.9	4.13	37.9	4.71
ADPA	**50.0**	**2.67**	**50.0**	**4.33**	**50.0**	**5.29**
- w/o Ref teacher	31.6	2.36	36.6	4.05	46.2	4.54

anced preference reward signals. This highlights the synergistic benefits of using DCKD for student initialization, particularly in capturing more granular preference structures during ADPA training.

4.3 Model Ablation

To evaluate the impact of different components in our methods, we conduct ablation experiments by removing each component from DCKD and ADPA. Specifically, for DCKD, we replace the DPO teacher with an SFT teacher trained on the preferred responses from the preference dataset. Additionally, We evaluate the effect of removing the $\mathcal{L}_{\text{KLD}-l}$ loss by excluding the dispreferred responses. For ADPA, we remove the reference teacher and minimize the reverse cross-entropy between the student and the DPO teacher's output distributions. Table 2 presents the ablation results on the DPO-MIX-7K dataset.

The results show that removing the DPO teacher in DCKD leads to noticeable performance degradation, highlighting the importance of the DPO training process. This suggests that the DPO teacher, by being optimized on human preference data, aligns better with human-like decision-making, and thus transfers more effective guidance to the student model. The absence of DPO training diminishes the teacher's capacity to represent nuanced preferences, resulting in less impactful knowledge transfer. Additionally, excluding dispreferred responses from DCKD also causes performance drops. This occurs because dispreferred responses help the student model learn not only which behaviors are preferred but also what to avoid. This component enables a more comprehensive understanding of both preferred and dispreferred behaviors, which is crucial for achieving better alignment with human preferences.

In the case of ADPA, removing the reference teacher results in significant performance losses. For example, in Danube3-0.5B, the MT-Bench score drops from 2.67 to 2.36, and the AlpacaEval win rate falls from 50.0% to 31.6%. This demonstrates that the reference teacher provides critical comparative feedback, allowing the Advantage Function to capture relative improvements in preference alignment. Without it, the student model lacks a robust baseline, weakening the reward signal and leading to diminished performance.

4.4 Analysis and Discussion

Impact of Different Levels of Reward. In our ADPA approach, we leverage a distribution-level reward signal to facilitate fine-grained preference learning. To demonstrate that ADPA provides a more stable and efficient training process, we distill a Danube2-1.8B model from the Mistral-7B model on DPO-MIX-7K dataset, and conduct a comprehensive comparison with traditional PPO-based methods which rely on token-level and sequence-level rewards. The details of *sequence-level* reward and *token-level* reward are provided in the Appendix B.

Table 3: Comparison of ADPA (distribution-level reward) with other levels of reward methods optimized by PPO.

Method	Reference	WR	LC WR
DPPO (Seq-Level)	ADPA	27.7	28.5
DPPO (Token-Level)	ADPA	40.0	39.3
ADPA (Distribution-Level)	ADPA	**50.0**	**50.0**

Using FsFairX as the evaluator, we tested the outputs of the trained student model on the DPO-MIX-7K validation set. As shown in Figure 3, ADPA significantly improves the stability of the training process compared to both token-level and sequence-level reward PPO methods. ADPA offers a detailed, distribution-level preference reward signal rather than assigning re-

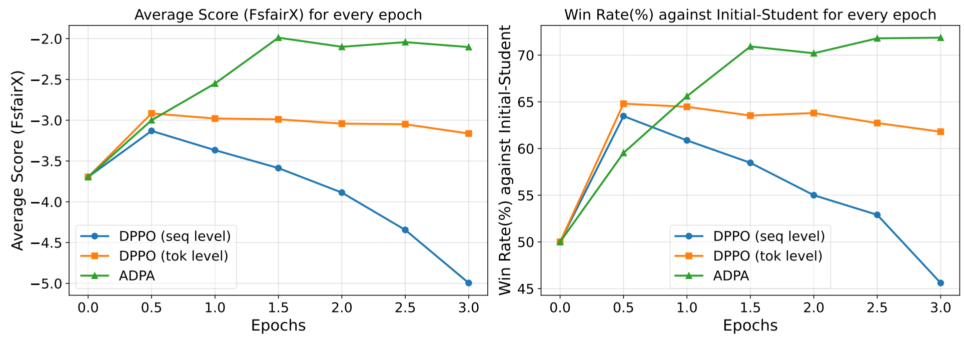

Figure 3: Comparison between ADPA and PPO based methods on test set over epochs. The x-axis represents the training epochs, and the y-axis represents the average score evaluated by the RM.

wards at the token or sequence level. Moreover, it employs an offline optimization process, which is more stable and efficient than the time-consuming and resource-intensive online RL training. Table 3 further presents the win rate of various approaches on AlpacaEval. ADPA clearly outperforms both PPO-based methods by significant margins. It achieves the highest win rate of 50.0% against itself, while the token-level and sequence-level reward PPO methods reach 40.0% and 27.7% win rate respectively. These results demonstrate that ADPA provides more stable training compared to PPO-based approaches, which rely on token-level or sequence-level rewards. By using the Advantage Function as a distribution-level reward, ADPA enables the student model to align more effectively with human preferences, resulting in better performance and faster convergence.

Sample Complexity Analysis. To evaluate the efficiency of ADPA, we analyze the sample complexity of identifying the optimal action a_t^* for a given state s_t under Advantage Function (distribution-level), token-level, and sequence-level rewards, as illustrated in Figure 4.4. For *Advantage Function*, finding the optimal action a_t^* at state s_t requires only evaluating the current sample, leading to a sample complexity of $O(1)$. In contrast, for *token-level reward*, the student model must explore each action $a_t' \in A$, transition to the next state $f(s_t, a_t')$, and obtain the corresponding reward. This results in a sample complexity of $O(|A|)$, as it requires evaluating all actions in the vocabulary. For *sequence-level reward*, the model must compute rewards over all possible future sequences, requiring $|A|^{T-t}$ samples, where T is the total length of the responses. This results in an exponential sample complexity of $O(|A|^{T-t})$. The lower sample complexity of ADPA contributes to more stable training by minimizing variance and reducing computational demands. This enhanced stability allows the student model to learn more efficiently from the teacher's preference signals, leading to superior overall performance compared to methods that depend on PPO optimization using token-level or sequence-level rewards. As a result, ADPA not only accelerates convergence but also achieves better alignment with human preferences.

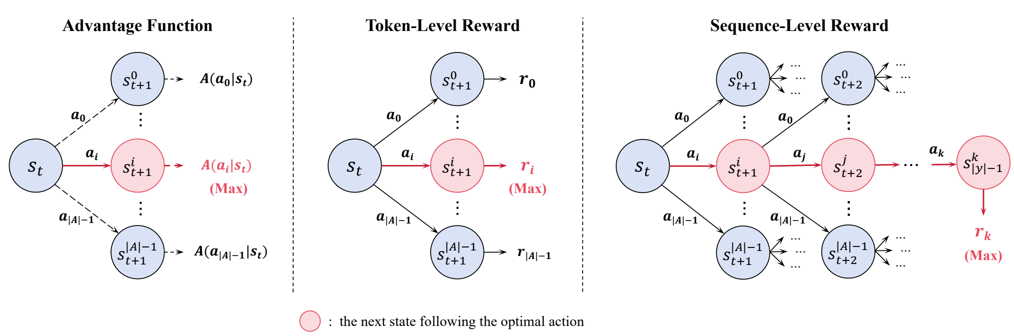

Figure 4: An illustration of our efficient ADPA with distribution-level reward signal, compares with token-level, and sequence-level rewards. **Left**: With a distribution-level reward, the student model directly selects the optimal action a_i based on $A(\cdot \mid s_t)$, as indicated by the dotted line, meaning no need for exploring other actions or states. **Middle**: For token-level rewards, the model takes actions (i.e., $a_0, \ldots, a_{|A|-1}$), transitions to subsequent states (i.e., $s_{t+1}^0, \ldots, s_{t+1}^{|A|-1}$), and receives reward signals (i.e., $r_0, \ldots, r_{|A|-1}$) before determining the optimal action. **Right**: With sequence-level rewards, the model must reach EOS before obtaining any reward signal, requiring exploration of all actions and states until then to identify the optimal action.

Impact of α and γ. We further investigate the effects of varying the hyperparameters α in DCKD, and γ in ADPA on the student model's preference alignment. We report the results of distilling Mistral-7B to Danube2-1.8B on DPO-MIX-7K dataset in Figure 5. The evaluation was conducted on the validation set using the Fsfairx reward model, which provided average scores for the responses generated by the student model. To further analyze the student model's ability to learn preference information, we employed the Reward Accuracy metric as defined by (Meng et al., 2024). This metric assesses the probability that the student model assigns a higher average log-probability to preferred responses compared to dispreferred ones in the preference dataset, effectively capturing the model's capability to distinguish between positive and negative samples.

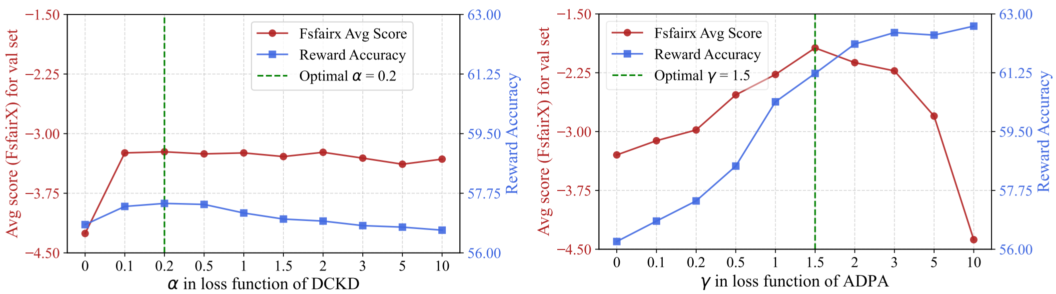

Figure 5: Variation of the average score by the RM on validation set and the reward accuracy on preference data with α in DCKD **(Left)** and γ in ADPA **(Right)**

It can be seen from the left figure that as the value of α increases, the reward accuracy initially rises and then declines, though the changes are not particularly significant when $\alpha \neq 0$. The highest average score on the FsFairX validation set is observed at $\alpha = 0.2$, indicating optimal performance at this value. However, the differences in average scores for other non-zero α values are relatively minor, suggesting that the model's preference alignment is fairly robust to variations in α.

From the right figure, we observe that as γ increases, both reward accuracy and the FsFairX average score consistently improve, suggesting that the student model becomes more adept at distinguishing between preferred and dispreferred responses. However, when γ exceeds a value of 3, the model becomes over-optimized with respect to the distillation objective, leading to a decline in both reward accuracy and the FsFairX score. This indicates that an excessively large γ causes the student model to overemphasize the Advantage Function signals, reducing its ability to generalize. Thus, while $\gamma = 1.5$ achieves the optimal balance, larger values (e.g., $\gamma > 3$) degrade performance by causing overfitting to advantage function.

Additional Analysis. We've conducted further experiments to understand the impact of different distillation objectives based on Q-function in Appendix C. The impact of source of state in ADPA is shown in Appendix D. Several case studies are provided in Appendix G.

5 Conclusion

In this paper, we address the challenge of aligning small language models with human preferences by leveraging knowledge distillation guided by larger, well-aligned teacher models. We first introduced DCKD, a straightforward method that employs KD with two KL-divergence constraints to transfer alignment knowledge from teacher to student. Acknowledging the limitations of DCKD in highlighting the differences between preferred and dispreferred responses, we proposed ADPA, which utilizes an *advantage function* derived from a teacher model trained with DPO, providing fine-grained, distribution-level reward signals that enhance the student's alignment with human preferences. Our extensive experiments demonstrate that both DCKD and ADPA improve the alignment of smaller language models. Additionally, ADPA+, which combines DCKD and ADPA, significantly improves the alignment of smaller language models, effectively narrowing the performance gap with larger models. This work highlights the potential of leveraging larger, preference-aligned models to guide the preference alignment of smaller models, offering a promising direction for developing effective preference-aligned language models in resource-constrained environments. Future work may explore further enhancements to the distillation process and investigate the applicability of the proposed methods to a broader range of tasks and model sizes.

REFERENCES

Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, et al. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021.

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

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

Alex James Chan, Hao Sun, Samuel Holt, and Mihaela van der Schaar. Dense reward for free in reinforcement learning from human feedback. In *Forty-first International Conference on Machine Learning*, 2024.

Hongzhan Chen, Siyue Wu, Xiaojun Quan, Rui Wang, Ming Yan, and Ji Zhang. Mcc-kd: Multi-cot consistent knowledge distillation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 6805–6820, 2023.

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. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, March 2023. URL `https://lmsys.org/blog/2023-03-30-vicuna/`.

Wojciech M Czarnecki, Razvan Pascanu, Simon Osindero, Siddhant Jayakumar, Grzegorz Swirszcz, and Max Jaderberg. Distilling policy distillation. In *The 22nd international conference on artificial intelligence and statistics*, pp. 1331–1340. PMLR, 2019.

Hanze Dong, Wei Xiong, Deepanshu Goyal, Yihan Zhang, Winnie Chow, Rui Pan, Shizhe Diao, Jipeng Zhang, SHUM KaShun, and Tong Zhang. Raft: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*, 2024.

Adam Fisch, Jacob Eisenstein, Vicky Zayats, Alekh Agarwal, Ahmad Beirami, Chirag Nagpal, Pete Shaw, and Jonathan Berant. Robust preference optimization through reward model distillation. *arXiv preprint arXiv:2405.19316*, 2024.

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

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

Jiwoo Hong, Noah Lee, and James Thorne. Reference-free monolithic preference optimization with odds ratio. *arXiv preprint arXiv:2403.07691*, 2024.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Simran Khanuja, Melvin Johnson, and Partha Talukdar. Mergedistill: Merging language models using pre-trained distillation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pp. 2874–2887, 2021.

Yoon Kim and Alexander M Rush. Sequence-level knowledge distillation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 1317–1327, 2016.

Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the effects of rlhf on llm generalisation and diversity. In *The Twelfth International Conference on Learning Representations*, 2024.

Jongwoo Ko, Sungnyun Kim, Tianyi Chen, and Se-Young Yun. Distillm: Towards streamlined distillation for large language models. In *Forty-first International Conference on Machine Learning*, 2024.

Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.

Harrison Lee, Samrat Phatale, Hassan Mansoor, Kellie Lu, Thomas Mesnard, Colton Bishop, Victor Carbune, and Abhinav Rastogi. Rlaif: Scaling reinforcement learning from human feedback with ai feedback. *arXiv preprint arXiv:2309.00267*, 2023.

Xuechen Li, Tianyi Zhang, Yann Dubois, Rohan Taori, Ishaan Gulrajani, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Alpacaeval: An automatic evaluator of instruction-following models, 5 2023. URL `https://github.com/tatsu-lab/alpaca_eval`.

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

Wei Liu, Weihao Zeng, Keqing He, Yong Jiang, and Junxian He. What makes good data for alignment? a comprehensive study of automatic data selection in instruction tuning. In *The Twelfth International Conference on Learning Representations*, 2024a.

Zhihan Liu, Miao Lu, Shenao Zhang, Boyi Liu, Hongyi Guo, Yingxiang Yang, Jose Blanchet, and Zhaoran Wang. Provably mitigating overoptimization in rlhf: Your sft loss is implicitly an adversarial regularizer. *arXiv preprint arXiv:2405.16436*, 2024b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

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

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

Pascal Pfeiffer, Philipp Singer, Yauhen Babakhin, Gabor Fodor, Nischay Dhankhar, and Sri Satish Ambati. H2o-danube3 technical report. *arXiv preprint arXiv:2407.09276*, 2024.

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

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

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

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

Philipp Singer, Pascal Pfeiffer, Yauhen Babakhin, Maximilian Jeblick, Nischay Dhankhar, Gabor Fodor, and Sri Satish Ambati. H2o-danube-1.8 b technical report. *arXiv preprint arXiv:2401.16818*, 2024.

Kaitao Song, Hao Sun, Xu Tan, Tao Qin, Jianfeng Lu, Hongzhi Liu, and Tie-Yan Liu. Lightpaff: A two-stage distillation framework for pre-training and fine-tuning. *arXiv preprint arXiv:2004.12817*, 2020.

Hao Sun. Reinforcement learning in the era of llms: What is essential? what is needed? an rl perspective on rlhf, prompting, and beyond. *arXiv preprint arXiv:2310.06147*, 2023.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

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

Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Shengyi Huang, Kashif Rasul, Alvaro Bartolome, Alexander M. Rush, and Thomas Wolf. The alignment handbook, 2024. URL `https://github.com/huggingface/alignment-handbook`.

Zhilin Wang, Yi Dong, Olivier Delalleau, Jiaqi Zeng, Gerald Shen, Daniel Egert, Jimmy J. Zhang, Makesh Narsimhan Sreedhar, and Oleksii Kuchaiev. Helpsteer2: Open-source dataset for training top-performing reward models, 2024.

Yuqiao Wen, Zichao Li, Wenyu Du, and Lili Mou. 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)*, pp. 10817–10834, 2023.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 38–45, 2020.

Taiqiang Wu, Chaofan Tao, Jiahao Wang, Zhe Zhao, and Ngai Wong. Rethinking kullback-leibler divergence in knowledge distillation for large language models. *arXiv preprint arXiv:2404.02657*, 2024.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. *arXiv preprint arXiv:2312.11456*, 2024.

Kevin Yang, Dan Klein, Asli Celikyilmaz, Nanyun Peng, and Yuandong Tian. Rlcd: Reinforcement learning from contrastive distillation for lm alignment. In *The Twelfth International Conference on Learning Representations*, 2023.

Shentao Yang, Shujian Zhang, Congying Xia, Yihao Feng, Caiming Xiong, and Mingyuan Zhou. Preference-grounded token-level guidance for language model fine-tuning. *Advances in Neural Information Processing Systems*, 36, 2024.

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

Rongzhi Zhang, Jiaming Shen, Tianqi Liu, Haorui Wang, Zhen Qin, Feng Han, Jialu Liu, Simon Baumgartner, Michael Bendersky, and Chao Zhang. Plad: Preference-based large language model distillation with pseudo-preference pairs. *arXiv preprint arXiv:2406.02886*, 2024.

Xingmeng Zhao, Tongnian Wang, Sheri Osborn, and Anthony Rios. Babystories: Can reinforcement learning teach baby language models to write better stories? In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pp. 186–197, 2023a.

Yao Zhao, Rishabh Joshi, Tianqi Liu, Misha Khalman, Mohammad Saleh, and Peter J Liu. Slic-hf: Sequence likelihood calibration with human feedback. *arXiv preprint arXiv:2305.10425*, 2023b.

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

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

Qihuang Zhong, Liang Ding, Li Shen, Juhua Liu, Bo Du, and Dacheng Tao. Revisiting knowledge distillation for autoregressive language models. *arXiv preprint arXiv:2402.11890*, 2024b.

Banghua Zhu, Evan Frick, Tianhao Wu, Hanlin Zhu, Karthik Ganesan, Wei-Lin Chiang, Jian Zhang, and Jiantao Jiao. Starling-7b: Improving helpfulness and harmlessness with RLAIF. In *First Conference on Language Modeling*, 2024.

SUMMARY OF THE APPENDIX

This appendix contains additional experimental results and discussions of *Advantage-Guided Distillation for Preference Alignment in Small Language Models*, organized as follows:

- Appendix **A** presents the **Derivation of the Q-function and Advantage Function**.
- Appendix **B** describes more **Details of the Sequence- and Token-Level Reward**.
- Appendix **C** presents **Variants of KD Objective based on Q-function**.
- Appendix **D** provides **Impact of the Source of State in ADPA**.
- Appendix **E** gives more **Details of Training Configurations**.
- Appendix **F** adds more discussions of **Limitations and Future Work**.
- Appendix **G** includes several **Case Studies**.

Algorithm 1 ADPA Training Pipeline

Require: Student model π_s, teacher model π_{dpo}, instruction-tuning dataset $\mathcal{D}_{\text{IT}}$, preference dataset $\mathcal{D}_{\text{pref}}$
Ensure: Trained student model π_s
1: Fine-tune π_{dpo} and π_s on $\mathcal{D}_{\text{IT}}$ to obtain supervised fine-tuned (SFT) models for both teacher and student, named Ref teacher π_{ref} and SFT student model $\pi_s^{'}$.
2: Fine-tune π_{ref} on $\mathcal{D}_{\text{pref}}$ using DPO to obtain π_{dpo} (DPO teacher model).
3: Create new dataset $\hat{\mathcal{D}} = \{\}$
4: **for** prompt x in $\mathcal{D}_{\text{pref}}$ **do**
5: Generate outputs from the SFT student model $\pi_s^{'}$ for the given prompt x to obtain $\hat{y}$.
6: $\hat{\mathcal{D}} \leftarrow \hat{\mathcal{D}} + \{(x, y_w, \hat{y})\}$
7: **end for**
8: Optimize the SFT student model $\pi_s^{'}$ on $(x, y_w, \hat{y})$ in $\hat{\mathcal{D}}$ using the ADPA loss.
9: Return the trained student model $\pi_s^{''}$.

Algorithm 2 ADPA+ Training Pipeline

Require: Student model π_s, Teacher model π_{dpo}, Instruction-Tuning dataset $\mathcal{D}_{\text{IT}}$, Preference dataset $\mathcal{D}_{\text{pref}}$
Ensure: Trained student model π_s
1: Fine-tune π_{dpo} and π_s on $\mathcal{D}_{\text{IT}}$ to obtain supervised fine-tuned (SFT) models for both teacher and student, named Ref teacher π_{ref} and SFT student model $\pi_s^{'}$.
2: Fine-tune π_{ref} on $\mathcal{D}_{\text{pref}}$ using DPO to obtain π_{dpo} (DPO teacher model).
3: Fine-tune $\pi_s^{'}$ on $\mathcal{D}_{\text{pref}}$ by DCKD algorithm, with the guidence of π_{dpo}, to obtain $\pi_{\text{stu-DCKD}}$.
4: Create new dataset $\hat{\mathcal{D}} = \{\}$
5: **for** prompt x in $\mathcal{D}_{\text{pref}}$ **do**
6: Generate outputs from DCKD student model $\pi_{\text{stu-DCKD}}$ for the given prompt x to obtain $\hat{y}$.
7: $\hat{\mathcal{D}} \leftarrow \hat{\mathcal{D}} + \{(x, y_w, \hat{y})\}$
8: **end for**
9: Optimize the DCKD student model $\pi_{\text{stu-DCKD}}$ on $(x, y_w, \hat{y})$ in $\hat{\mathcal{D}}$ using the ADPA loss, to obtain the $\pi_s^{''}$.
10: Return the trained student model $\pi_s^{''}$.

A Derivations of Q-function and Advantage Function

We model the generation of LLM as a Markov Decision Process (MDP), where each token generation corresponds to an action, and the current state consists of the prompt and all previously generated tokens$s_t = (x, y_{<t})$. Let the trajectory $\tau = \{(s_t, a_t)\}_{t=1}^{|\tau|}$ represent the generation sequence, where $|\tau|$ is the length of the trajectory and $a_{|\tau|} = \text{EOS}$. The sequence-level reward provided by the reward model (RM) is applied in the final step. Therefore, the reward function is defined as:

$$r(s_t) = \begin{cases} 0, & \text{if } a_t \neq \text{EOS} \\ \text{RM}(\tau), & \text{if } a_t = \text{EOS} \end{cases} \tag{12}$$

In this MDP framework, considering the DPO-trained model π_{dpo} and the reference model π_{ref} prior to DPO training, the Q-function for state s_t and action a can be expressed as following:

$$Q_{\text{dpo}}(s_t, a) = \beta \left[\sum_{i=1}^{t-1} \log \frac{\pi_{\text{dpo}}(a_i \mid s_i)}{\pi_{\text{ref}}(a_i \mid s_i)} + \log \frac{\pi_{\text{dpo}}(a \mid s_t)}{\pi_{\text{ref}}(a \mid s_t)} \right], \tag{13}$$

Proof. According to our MDP framework, the future returns of the reference policy π_{ref} at the current timestep are determined by the **Bellman equation**, a fundamental concept in reinforcement learning that relates the value of a state-action pair to the immediate reward and the expected value of the next state.

The Bellman equation for the Q-function $Q_{\text{ref}}(s_t, a_t)$ is given by:

$$Q_{\text{ref}}(s_t, a_t) = r(s_t, a_t) + \gamma V_{\text{ref}}(s_{t+1}), \tag{14}$$

where:

- $r(s_t, a_t)$ is the **immediate reward** received after taking action a_t in state s_t.
- γ is the **discount factor**, which balances immediate and future rewards. In the context of LLM RLHF, γ always typically set as 1, to consider the full future reward without discounting.
- $V_{\text{ref}}(s_{t+1})$ is the **value of the next state** s_{t+1}, representing the expected cumulative reward from that state onward under policy π_{ref}.
- $s_{t+1} = f(s_t, a_t)$ is the **next state**, resulting from taking action a_t in state s_t. In language generation, this corresponds to appending the token a_t to the sequence s_t.

In our specific setting:

- The immediate reward $r(s_t, a_t)$ is **zero** at every timestep except when the **end-of-sequence (EOS)** token is generated.
- When the EOS token is generated, the immediate reward is provided by the reward model $\text{RM}(\tau)$, which evaluates the entire generated sequence τ.

Therefore, we can express the Bellman equation for $Q_{\text{ref}}(s_t, a_t)$ as:

$$Q_{\text{ref}}(s_t, a_t) = \begin{cases} 0 + \gamma V_{\text{ref}}(s_{t+1}) = V_{\text{ref}}(s_{t+1}), & \text{if } a_t \neq \text{EOS}, \\ \text{RM}(\tau) + \gamma V_{\text{ref}}(s_{t+1}), & \text{if } a_t = \text{EOS}. \end{cases} \tag{15}$$

Since there are no future rewards after generating the EOS token (the sequence ends), $V_{\text{ref}}(s_{t+1}) = 0$ when $a_t = \text{EOS}$. Additionally, because $\gamma = 1$, the equation simplifies to:

$$Q_{\text{ref}}(s_t, a_t) = \begin{cases} V_{\text{ref}}(s_{t+1}), & \text{if } a_t \neq \text{EOS}, \\ \text{RM}(\tau), & \text{if } a_t = \text{EOS}. \end{cases} \tag{16}$$

To further our derivation, we first define the **value function** $V_{\text{ref}}(s_t)$ for the reference policy π_{ref}. This function represents the expected cumulative reward starting from state s_t when actions are selected according to π_{ref}:

$$V_{\text{ref}}(s_t) = \mathbb{E}_{a\sim\pi_{\text{ref}}(\cdot|s_t)}\left[Q_{\text{ref}}(s_t, a)\right]. \tag{17}$$

Next, we consider the **DPO-trained policy** π_{dpo}. This policy can be associated with a sequence-level reward model, which assigns rewards based on the divergence from the reference policy over an entire trajectory τ:

$$\text{RM}_{\text{dpo}}(\tau) = \beta \sum_{t=1}^{|\tau|} \log \frac{\pi_{\text{dpo}}(a_t \mid s_t)}{\pi_{\text{ref}}(a_t \mid s_t)}. \tag{18}$$

Here, β is a scaling parameter introduced in the DPO algorithm. The partition function $Z(s_1)$ that normalizes the reward model is omitted in optimization since it does not affect the gradients with respect to the policy parameters.[3]

By substituting the reward model $\text{RM}_{\text{dpo}}(\tau)$ into the Bellman equation, we can express the Q-function for the DPO-trained policy:

$$Q_{\text{dpo}}(s_t, a_t) = V_{\text{dpo}}(s_{t+1}) = \mathbb{E}_{\tau'}\left[\beta \sum_{i=1}^{|\tau'|} \log \frac{\pi_{\text{dpo}}(a_i \mid s_i)}{\pi_{\text{ref}}(a_i \mid s_i)}\right], \tag{19}$$

where $s_{t+1} = f(s_t, a_t)$ is the state resulting from taking action a_t in state s_t, typically corresponding to appending the token a_t in text generation tasks.

In this expression, the trajectory τ' includes:

1. **Initial Segment**: The sequence of states and actions before time t, denoted as (s_i, a_i) for $i = 1$ to $t - 1$.
2. **Current State and Action**: The pair (s_t, a_t).
3. **Future Segment**: The sequence of states and actions from s_{t+1} onward, generated by following the reference policy π_{ref} until the end of the sequence (EOS) is reached.

By considering these components, the expected cumulative reward accounts for the immediate divergence from the reference policy at time t and the expected future divergence when following π_{ref} afterward. This formulation helps in understanding how the DPO-trained policy evaluates the benefit of taking action a_t in state s_t in terms of preference alignment.

In language generation tasks using LLMs, we can reasonably assume that both the reference policy π_{ref} and the DPO-trained policy π_{dpo} assign a probability of 1 to generating the end-of-sequence (EOS) token at the final state $s_{|\tau|}$:

$$\pi_{\text{ref}}(a = \text{EOS} \mid s_{|\tau|}) = \pi_{\text{dpo}}(a = \text{EOS} \mid s_{|\tau|}) = 1.$$

This means that once the model reaches the end of the sequence, it will produce the EOS token with certainty.

Additionally, the EOS token generated at step $|\tau|$ does not contribute to the overall reward provided by the reward model (RM), as the reward depends on the sequence generated up to that point.

Base Case ($t = |\tau|$):

At the final time step $t = |\tau|$, the Q-function can be expressed as:

$$Q_{\text{dpo}}(s_{|\tau|}, a) = \text{RM}(\tau) = \beta \sum_{i=1}^{|\tau|} \log \frac{\pi_{\text{dpo}}(a_i \mid s_i)}{\pi_{\text{ref}}(a_i \mid s_i)}.$$

[3] The partition function $Z(s_1)$ normalizes the reward function provided by the DPO-trained model. In practice, it can be omitted during optimization without influencing the outcome (Zhong et al., 2024a; Rafailov et al., 2024a).

Since the EOS token does not affect the reward and both policies generate it with probability 1, the term involving the EOS token ($i = |\tau|$) contributes nothing to the sum (as $\log 1 = 0$). Therefore, we can simplify the expression:

$$Q_{\text{dpo}}(s_{|\tau|}, a) = \beta \sum_{i=1}^{|\tau|-1} \log \frac{\pi_{\text{dpo}}(a_i \mid s_i)}{\pi_{\text{ref}}(a_i \mid s_i)}.$$

This matches the proposed expression for the Q-function at $t = |\tau|$.

Inductive Step:

Assuming that Eq. (13) is established when $t = k$, we can prove that it is true when $t = k - 1$:

$$\begin{aligned}
&Q_{\text{dpo}}(s_{k-1}, a) \\
&= V_{\text{dpo}}(f(s_{k-1}, a)) \\
&= \mathbb{E}_{a' \sim \pi_{\text{ref}}(\cdot|f(s_{k-1},a))} \left[Q_{\text{dpo}}(f(s_{k-1}, a), a')\right] \\
&= \mathbb{E}_{a' \sim \pi_{\text{ref}}(\cdot|f(s_{k-1},a))} \left[\sum_{t=1}^{k-2} \beta \log \frac{\pi_{\text{dpo}}(a_t \mid s_t)}{\pi_{\text{ref}}(a_t \mid s_t)} + \beta \log \frac{\pi_{\text{dpo}}(a \mid s_{k-1})}{\pi_{\text{ref}}(a \mid s_{k-1})} + \beta \log \frac{\pi_{\text{dpo}}(a' \mid f(s_{k-1}, a))}{\pi_{\text{ref}}(a' \mid f(s_{k-1}, a))}\right] \\
&= \beta \sum_{t=1}^{k-2} \log \frac{\pi_{\text{dpo}}(a_t|s_t)}{\pi_{\text{ref}}(a_t|s_t)} + \log \frac{\pi_{\text{dpo}}(a|s_{k-1})}{\pi_{\text{ref}}(a|s_{k-1})} + \beta \log \mathbb{E}_{a' \sim \pi_{\text{ref}}(\cdot|f(s_{k-1},a))} \frac{\pi_{\text{dpo}}(a'|f(s_{k-1}, a))}{\pi_{\text{ref}}(a'|f(s_{k-1}, a))} \\
&= \beta \sum_{t=1}^{k-2} \log \frac{\pi_{\text{dpo}}(a_t|s_t)}{\pi_{\text{ref}}(a_t|s_t)} + \log \frac{\pi_{\text{dpo}}(a|s_{k-1})}{\pi_{\text{ref}}(a|s_{k-1})}
\end{aligned} \tag{20}$$

Therefore, Eq. (13) is established when $1 \leq t \leq |\tau|$. The value function $V(s)$ can be formulated as:

$$V_{\text{dpo}}(s_i) = Q_{\text{dpo}}(s_{i-2}, a_{i-1}) = \beta \sum_{t=1}^{i-1} \log \frac{\pi_{\text{dpo}}(a_t|s_t)}{\pi_{\text{ref}}(a_t|s_t)} \tag{21}$$

The advantage function can be formulated as:

$$A_{\text{dpo}}(s_i, a) = Q_{\text{dpo}}(s_i, a) - V_{\text{dpo}}(s_i) = \beta \log \frac{\pi_{\text{dpo}}(a|s_i)}{\pi_{\text{ref}}(a|s_i)} \tag{22}$$

□

B Details of the Sequence- and Token-Level Reward

We provide more details of the sequence-level and token-level rewards in this section. Specifically, the *sequence-level* reward given by the DPO teacher is defined as:

$$\text{RM}(x, y) = \beta_T \log \frac{\pi_{\text{dpo}}(y \mid x)}{\pi_{\text{ref}}(y \mid x)} \tag{23}$$

Here β_T is the beta parameter in the training process of the DPO teacher. The reward is assigned to the last position in the sequence, while all positions are regulated by a KLD penalty. The reward for each token at time step t is given as follows:

$$r_{\text{sequence-level}}(x, y_t) = \begin{cases} 0 - \beta \log \frac{\pi_s(y_t|x, y_{<t})}{\pi_{\text{ref}}(y_t|x, y_{<t})}, & \text{if } y_t \neq \text{EOS} \\ \text{RM}(x, y) - \beta \log \frac{\pi_s(y_t|x, y_{<t})}{\pi_{\text{ref}}(y_t|x, y_{<t})}, & \text{if } y_t = \text{EOS} \end{cases} \tag{24}$$

For the *token-level* reward, each token in the sequence receives an individual reward from the DPO teacher and Ref teacher:

$$\text{RM}(\{x, y_{<t}\}, y_t) = \beta_T \log \frac{\pi_{\text{dpo}}(y_t \mid x, y_{<t})}{\pi_{\text{ref}}(y_t \mid x, y_{<t})} \tag{25}$$

In this experiment, we use the token-level difference in the output log probability between the DPO teacher and the Ref teacher as the token-level reward (Zhong et al., 2024a). The reward for each token at time step t is:

$$r_{\text{token-level}}(x, y_t) = \text{RM}(\{x, y_{<t}\}, y_t) - \beta \log \frac{\pi_s(y_t \mid x, y_{<t})}{\pi_{\text{ref}}(y_t \mid x, y_{<t})} \tag{26}$$

We use the token-level and sequence-level rewards, defined above, to optimize the student model using PPO, named Distilled PPO (DPPO). In order to be fair when comparing ADPA and DPPO with different levels of reward, and to enhance the stability of the online RL process, we add $\mathcal{L}_{\text{SFT}}$ with a weight of 1 into the overall loss function.

C Variants of KD Objective based on Q-function

In this section, we explore alternative approaches to utilizing the Q-function derived from the DPO-trained teacher and reference models in the KD process. Our goal is to investigate whether different formulations of the KD objective based on the Q-function can enhance the student's preference alignment performance compared to our proposed ADPA method.

Specifically, we use argmax or softmax operation on $Q_{\text{dpo}}(\cdot \mid s)$ to obtain policies[4], and then distill it to student by minimizing the KLD or Cross Entropy (CE) loss between the student policy and the policies deduced by Advantage Function (Rusu et al., 2015; Czarnecki et al., 2019).

Using argmax on $Q_{\text{dpo}}(\cdot \mid s)$ and then distilling allows the student model to focus on mimicking the Advantage Function's most confident decisions.

$$\mathcal{L}_{\text{A-argmax}} = \mathbb{E}_{(x,y,\hat{y})} \left[\mathcal{L}_{\text{SFT}}(x, y) + \frac{\gamma}{|\hat{y}|} \sum_{i=1}^{|\hat{y}|} \text{CE} \left(\mathbf{1}_{\{\arg\max_a A_{\text{dpo}}(s_t, \cdot)\}}, \pi_s(\cdot \mid s_t) \right) \right] \tag{27}$$

Using softmax on $Q_{\text{dpo}}(\cdot \mid s)$ and then distilling allows the student model to learn from the Advantage Function's full policy distribution, capturing nuances in decision-making that go beyond simply selecting the highest Q-value action.

$$\mathcal{L}_{\text{A-softmax}} = \mathbb{E}_{(x,y,\hat{y})} \left[\mathcal{L}_{\text{SFT}}(x, y) + \frac{\gamma}{|\hat{y}|} \sum_{i=1}^{|\hat{y}|} D_{\text{KL}} \left(\text{softmax}(A_{\text{dpo}}(s_t, \cdot)) \mid\mid \pi_s(\cdot \mid s_t) \right) \right] \tag{28}$$

We conducted experiments using these variants and compared them with the ADPA method. The results are presented in Table 4. The experimental results indicate that our proposed ADPA method outperforms both Q-argmax KD and Q-softmax KD.

Table 4: Comparison with Q-argmax KD and Q-softmax KD. We show the Win Rate (WR) and Length Control Win Rate (LC WR) against ADPA on AlpacaEval.

Method	Reference	WR (%)	LC WR (%)
Q-argmax KD	ADPA	41.8	42.1
Q-softmax KD	ADPA	28.2	28.7
ADPA	ADPA	**50.0**	**50.0**

D Impact of the Source of State s in ADPA

In the optimization objective of ADPA 10, the state s_t can be sourced not only from the student model's own sampling but also from other sources. We conducted additional experiments on DPO-MIX-7K dataset for Danube3-0.5B to compare the effects of different sources of s_t with the standard

[4]Adding constants to the inputs of softmax and argmax does not affect the results. For simplicity, we apply these operations to the Advantage Function in Eq. (9).

Table 5: Comparison of different sources of s_t in Eq. (10). We show the Win Rate (WR) and Length Control Win Rate (LC WR) against ADPA on AlpacaEval.

Method	**Reference**	**WR (%)**	**LC WR (%)**
s_t from preferred responses	ADPA	30.6	34.2
s_t from dispreferred responses	ADPA	49.1	48.8
s_t from teacher sampling text	ADPA	30.5	30.6
s_t from student sampling text (default in ADPA)	ADPA	**50.0**	**50.0**

ADPA approach: (1) using the preferred responses from the preference dataset as samples for s_t, (2) using dispreferred responses from the preference dataset, and (3) using text generated by the teacher model as the source for s_t.

Table 5 presents a comparison of ADPA when using different sources for state s_t. The default ADPA setting, which uses state s_t sampled from the student's own generated text, achieves the highest performance. This result underscores the importance of aligning the training process with the inference conditions. When the student model generates its own samples $\hat{y}$, it creates a training environment that closely mirrors the actual conditions encountered during inference, leading to more effective learning and better overall performance.

E Details of Training Configurations

In our experiments, we train the teacher LLMs (Mistral-7B and LLaMA-2-13B) and LLaMA-2-7B students on a single node with 8x80GB NVIDIA A800 GPUs. For student models with other sizes (0.5B and 1.8B), we train them on a single node with 4x24GB NVIDIA RTX 3090 GPUs. All experiments are optimized using the AdamW (Loshchilov & Hutter, 2019) optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use a weight decay of 0.0 and gradient clipping of 1.0. A cosine learning rate schedule is employed, with a maximum learning rate of The optimal learning rate obtained through search (often 1.0e-5) and a warmup ratio of 0.1. Our training framework is implemented based on the HuggingFace Transformers (Wolf et al., 2020) and alignment-handbook (Tunstall et al., 2024).

F Limitations and Future Work

Limitations While our proposed methods, DCKD and ADPA, demonstrate significant improvements in aligning smaller language models with human preferences, several limitations warrant consideration:

- **Dependence on Teacher Models**: The effectiveness of our approaches relies heavily on the availability of well-aligned teacher models. If such teacher models are not accessible or are misaligned, the performance gains of the student models may be limited.
- **Computational Overhead**: Computing the Advantage Function in ADPA requires access to the output probabilities of both the DPO-trained teacher and the reference model at each token generation step. This process introduces additional computational overhead, especially for models with large vocabularies or when processing long sequences.
- **Generalization Across Domains**: Our experiments are primarily conducted on specific datasets and domains. The generalizability of our methods to a broader range of tasks, languages, and more diverse datasets remains to be thoroughly investigated.

Future Work Building upon our findings, several avenues for future research are identified:

- **Enhancing Teacher Model Alignment**: Investigate strategies for improving the alignment of teacher models when well-aligned teachers are not readily available. This could involve developing more effective alignment techniques for large models or utilizing ensemble methods.
- **Reducing Computational Complexity**: Explore techniques to minimize the computational overhead associated with computing the Advantage Function. Potential solutions include approximation methods, efficient sampling strategies, or leveraging parallel computation.

- **Cross-Domain Generalization**: Evaluate the effectiveness of the proposed methods across a wider array of tasks, languages, and domains to assess their robustness and adaptability in different settings.

G CASE STUDIES

We provide some cases generated by the models distilled by different methods based on the Danube2-1.8B model family in Table 6. The prompts are sampled from the DPO-MIX-7K dataset. We find that ADPA+ generates more accurate responses compared with the baselines.

Table 6: Instruction-following cases from the validation set of DPO-MIX-7K. ADPA+ can better follow the instructions and generate more accurate responses. Without the guidance of the Advantage Function, the distilled models generate redundant sentences (Case #1).

Case #1	
Instruction	The task is to read a passage and convert numbers in their text forms into digit forms. The text numbers can be used to represent Quantity, Dates and Time, Addresses or Positions.\nOne example is below.\nQ: Jacob ate three burgers today.\nA: Jacob ate 3 burgers today.\nRationale: The number in text form three is converted into the number in digit form 3.\nQ: If Oswald left the Depository Building at twelve:thirty three p.m.,\nA:
SFT w/o KD	If Oswald left the Depository Building at 12:33 p.m.\nRationale: The number in text form 12:33 p.m. is converted into the number in digit form 1233.
DPO	If Oswald left the Depository Building at 12:33 p.m.,\n\nTo solve this task, I will follow these steps:\n\n1. Identify the text numbers in the given passage.\n2. Convert each text number into its corresponding digit form.\n3. Combine the digit forms to represent the numbers in the text.\n\nHere's the step-by-step process:\n\n1. Identify the text numbers in the given passage:\n\n* Jacob ate three burgers today.\n* If Oswald left the Depository Building at twelve:thirty three p.m.,\n\n2. Convert each text number into its corresponding digit form:\n\n* Jacob ate 3 burgers today. (3 in decimal)\n* If Oswald left the Depository Building at twelve:thirty three p.m., (12 in decimal, 30 in decimal, 33 in decimal)\n\n3. Combine the digit forms to represent the numbers in the text:\n\n* Jacob ate 3 burgers today. (3 in decimal)\n* If Oswald left the Depository Building at twelve:thirty three p.m., (12 in decimal, 30 in decimal, 33 in decimal)\n\nSo, the final answer is:\n\n* Jacob ate 3 burgers today.\n* If Oswald left the Depository Building at twelve:thirty three p.m.,
DCKD	If Oswald left the Depository Building at 12:33 p.m.\n\nIn this example, the text number Ï2:33ïs converted into the digit form Ï233̈.
ADPA	If Oswald left the Depository Building at 12:33 p.m.
ADPA+	If Oswald left the Depository Building at 12:33 p.m.
ADPA w/o Ref teacher	If Oswald left the Depository Building at 12:33 p.m.\nRationale: The number in text form 12:33 p.m. is converted into the number in digit form 1233.
Case #2	
Instruction	The ancient city of Carthage, founded by Phoenician settlers, was located in what present-day North African country?\n
PLAD	The ancient city of Carthage, founded by Phoenician settlers, was located in what present-day North African country?\n\nAnswer: Tunisia.
DCKD	The ancient city of Carthage, founded by Phoenician settlers, was located in what present-day North African country?\n\nThe ancient city of Carthage was located in present-day Tunisia.
ADPA+	The ancient city of Carthage, founded by Phoenician settlers, was located in present-day Tunisia.
ADPA w/o Ref teacher	The ancient city of Carthage, founded by Phoenician settlers, was located in what present-day North African country?