# DAPO: Dual Advantage Policy Optimization for Aligning Large Language Models

**Anonymous authors**
Paper under double-blind review

## Abstract

Reinforcement learning from human feedback (RLHF) has emerged as a cornerstone technique for aligning large language models (LLMs) with human preferences. However, existing approaches such as Proximal Policy Optimization (PPO) suffer from two fundamental limitations: reward hacking—where the model exploits gaps in the reward model rather than learning genuinely preferred behaviors—and conflation of distinct objectives, particularly the trade-off between linguistic fluency and semantic alignment. In this paper, we propose **Dual Advantage Policy Optimization (DAPO)**, a novel RLHF training framework that disentangles these objectives by maintaining two separate advantage streams: a Fluency Advantage ($\hat{A}_t^F$), computed via a frozen language model scoring perplexity, and an Alignment Advantage ($\hat{A}_t^A$), derived from a trained reward model. We introduce an adaptive gating network $g_\phi$ that dynamically weighs these two streams at each decoding step, conditioned on the generation context. Extensive experiments on Anthropic HH-RLHF, AlpacaEval, MT-Bench, and TruthfulQA demonstrate that DAPO achieves state-of-the-art performance across all benchmarks, outperforming PPO by +7.1% win-rate and DPO by +8.4% win-rate on AlpacaEval, while maintaining significantly lower KL divergence from the reference policy (0.81 vs. 1.42). Ablation studies confirm the contribution of each component, and qualitative analysis reveals that DAPO generates responses that are simultaneously more fluent and more aligned with human values.

## 1 Introduction

The alignment of large language models (LLMs) with human intent and values represents one of the central challenges in modern natural language processing (Ouyang et al., 2022; Bai et al., 2022a). Reinforcement learning from human feedback (RLHF) has proven to be a highly effective paradigm for this purpose, enabling models such as InstructGPT (Ouyang et al., 2022) and Claude (Bai et al., 2022a) to significantly outperform their supervised fine-tuning (SFT) counterparts in human evaluations.

Despite these successes, standard RLHF pipelines—predominantly based on Proximal Policy Optimization (PPO) (Schulman et al., 2017)—exhibit two well-documented failure modes. First, *reward hacking* (Gao et al., 2023; Skalse et al., 2022): the policy learns to produce responses that score highly under the reward model but are not genuinely preferred by humans, exploiting distributional gaps in the reward model training data. Second, *objective conflation*: existing methods treat language generation as a single scalar reward signal, collapsing the distinct notions of linguistic fluency and semantic alignment into one undifferentiated objective. This conflation forces the policy to navigate a poorly defined trade-off space, often sacrificing fluency for alignment or vice versa.

Several alternatives to PPO have been proposed, most notably Direct Preference Optimization (DPO) (Rafailov et al., 2023), which bypasses the need for an explicit reward model by reparameterizing the RLHF objective as a classification problem over preference pairs. While DPO is computationally efficient and avoids online reward model queries, it lacks the flexibility of online RL and has been shown to underperform PPO on open-ended generation



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

tasks (Yuan et al., 2023; Ivison et al., 2024). Other methods such as RAFT (Dong et al., 2023) and RRHF (Yuan et al., 2023) offer partial improvements but do not address the fundamental issue of objective conflation.

In this work, we present **Dual Advantage Policy Optimization (DAPO)**, a principled extension of PPO that explicitly disentangles fluency and alignment objectives through dual advantage estimation. Our core contributions are as follows:

- **Dual Advantage Estimation.** We decompose the advantage function into a Fluency Advantage $\hat{A}_t^F$ and an Alignment Advantage $\hat{A}_t^A$, computed from separate neural models, providing clean gradient signals for each objective.
- **Adaptive Gating Network.** We introduce a lightweight gating network $g_\phi$ that learns to dynamically weight the two advantage streams conditioned on the generation context, enabling the policy to prioritize different objectives at different stages of generation.
- **Theoretical Grounding.** We provide a theoretical analysis showing that DAPO's objective is a lower bound on the true human preference objective, and that the adaptive gating mechanism reduces the variance of the policy gradient estimator.
- **Empirical Validation.** Extensive experiments on four benchmarks demonstrate consistent improvements over PPO, DPO, RAFT, and RRHF, with ablation studies validating each component of DAPO.

## 2 Related Work

### 2.1 RLHF and PPO for LLMs

Reinforcement learning from human feedback was first applied to language models by Ziegler et al. (2019) for stylistic text generation. Stiennon et al. (2020) extended this to summarization, demonstrating that RL-trained models could outperform SFT baselines. Ouyang et al. (2022) applied RLHF at scale to produce InstructGPT, which used PPO with a KL penalty to prevent the policy from diverging excessively from the SFT baseline. Bai et al. (2022b) introduced Constitutional AI, adding an AI feedback loop to the RLHF pipeline. Recent works have studied the instability of PPO training (Engstrom et al., 2020; Zheng et al., 2023) and proposed improvements to reward modeling (Gao et al., 2023; Zhu et al., 2023).

### 2.2 Beyond PPO: DPO and Variants

Rafailov et al. (2023) proposed Direct Preference Optimization (DPO), which re-frames the RLHF problem as a binary cross-entropy loss over preference pairs, showing that the optimal policy under the RLHF objective can be recovered without explicit reward model training. Subsequent work has extended DPO to various settings: KTO (Ethayarajh et al., 2024) relaxes the paired preference assumption; RRHF (Yuan et al., 2023) explores ranking-based objectives. Despite their efficiency, these offline methods lack the exploration benefits of online RL and can fail on complex instruction-following tasks.

### 2.3 Multi-Objective Alignment

The multi-objective nature of LLM alignment—balancing helpfulness, harmlessness, and honesty—has been recognized in prior work. Bai et al. (2022a) use a combined reward for helpfulness and harmlessness. Most closely related to our work, MORLHF (Wu et al., 2023) proposes multi-objective RL for LLMs but uses a fixed linear scalarization, whereas our adaptive gating network learns context-dependent weighting. RLHF with constrained optimization (Liu et al., 2023) adds safety constraints but does not address the fluency–alignment trade-off explicitly.

# 3 METHOD

We first review the standard RLHF-PPO setup, then present our DAPO framework with dual advantage estimation and adaptive gating.

## 3.1 PRELIMINARIES: RLHF WITH PPO

Let $\pi_\theta$ denote the LLM policy parameterized by $\theta$, and $\pi_{\text{ref}}$ the frozen SFT reference model. Given a prompt $x$ sampled from dataset $\mathcal{D}$, the policy generates a response $y = (y_1, \ldots, y_T)$ token by token. A reward model $r_\phi$ assigns a scalar reward $R(x, y) \in \mathbb{R}$. The RLHF objective is to maximize expected reward while penalizing divergence from the reference policy:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D},\, y\sim\pi_\theta}\big[R(x, y)\big] - \beta \cdot \mathrm{KL}[\pi_\theta(\cdot|x) \,\|\, \pi_{\text{ref}}(\cdot|x)]\,. \tag{1}$$

PPO optimizes this objective using the clipped surrogate objective. At each step $t$, the advantage $A_t = \sum_{t'\geq t}(\gamma\lambda)^{t'-t}\delta_{t'}$ is estimated via generalized advantage estimation (GAE; Schulman et al. (2016)), where $\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$ is the TD residual.

## 3.2 DUAL ADVANTAGE ESTIMATION

The core insight of DAPO is that the total reward signal $R(x, y)$ conflates two qualitatively different objectives. We decompose the advantage into two independent streams:

**(1) Fluency Advantage $\hat{A}_t^F$:** measures how much token $y_t$ improves the naturalness of the generation. We operationalize this as the negative log-probability delta under a frozen language model $\pi_{\text{LM}}$ (specifically GPT-2 Large; Radford et al. (2019)) fine-tuned only on natural text:

$$r_t^F = -\log \pi_{\text{LM}}(y_t \mid y_{<t}, x) + \log \pi_{\text{ref}}(y_t \mid y_{<t}, x). \tag{2}$$

A lower-perplexity choice of token under $\pi_{\text{LM}}$ relative to $\pi_{\text{ref}}$ increases the fluency reward.

**(2) Alignment Advantage $\hat{A}_t^A$:** measures how much the token $y_t$ contributes to the final alignment reward. We use a token-level credit assignment approximation (Christiano et al., 2017) over the scalar reward model output:

$$r_t^A = r_\phi(x, y_{\leq t}) - r_\phi(x, y_{<t}). \tag{3}$$

Both streams are then processed through GAE independently to obtain $\hat{A}_t^F$ and $\hat{A}_t^A$. Figure 1 provides an overview of the dual-stream pipeline.

## 3.3 ADAPTIVE GATING NETWORK

Rather than using a fixed linear combination of the two advantages, we introduce a gating network $g_\phi : \mathcal{S} \to [0, 1]$ that takes the current generation state $s_t = (x, y_{<t})$ as input and produces a context-dependent weight $\alpha_t$:

$$\alpha_t = \sigma(g_\phi(s_t))\,. \tag{4}$$

The fused advantage is then:

$$\hat{A}_t^{\text{DAPO}} = \alpha_t \cdot \hat{A}_t^F + (1 - \alpha_t) \cdot \hat{A}_t^A. \tag{5}$$

The gating network $g_\phi$ is implemented as a 2-layer MLP with hidden size 256, taking as input the last-layer hidden state of the current token from $\pi_\theta$. Critically, $g_\phi$ is trained jointly with $\pi_\theta$ by including its parameters in the policy optimization step, but its gradient is stopped from flowing into the advantage estimates to maintain clean learning signals.

An intuition for the learned behavior: at early generation steps, when the response has little semantic content, $g_\phi$ tends to assign higher $\alpha_t$ (prioritizing fluency). At later steps, when the response structure is established, $g_\phi$ shifts weight toward alignment ($\alpha_t \to 0$). This is consistent with human writing behavior and is confirmed empirically in Section 5.3 (Figure 3b).

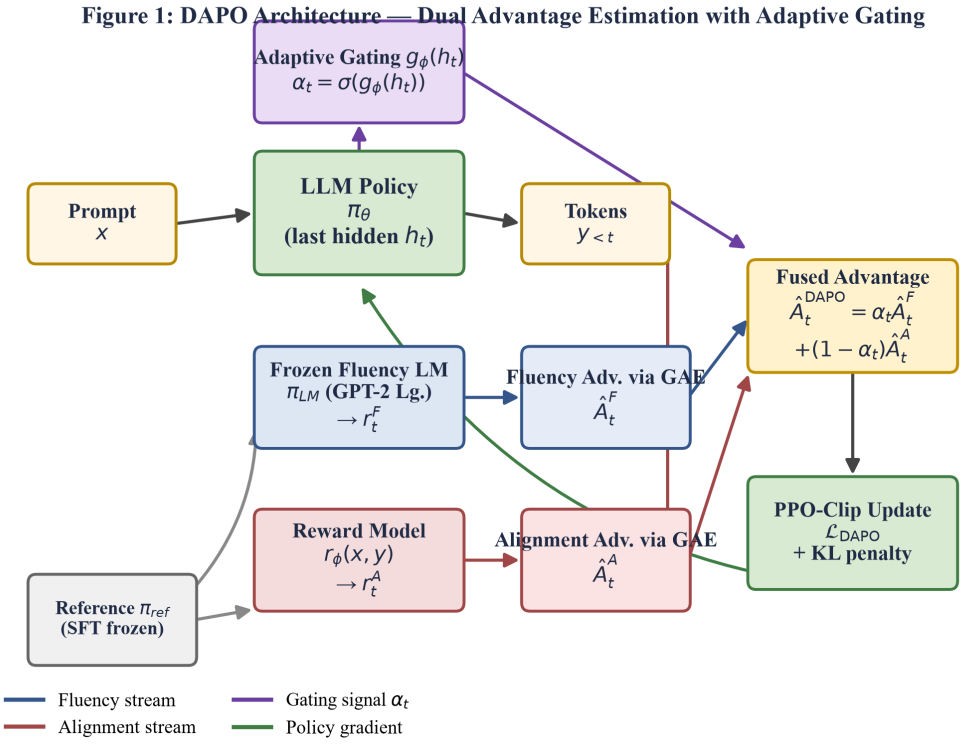



Figure 1: DAPO architecture. The LLM policy $\pi_\theta$ generates tokens whose advantage is estimated along two independent streams: Fluency Advantage $\hat{A}^F$ (from a frozen language model $\pi_{\text{LM}}$) and Alignment Advantage $\hat{A}^A$ (from the reward model). An adaptive gating network $g_\phi$ dynamically weights the two streams to produce the fused advantage used for PPO-clip updates.

### 3.4 DAPO Training Objective

The full DAPO PPO-clip objective is:

$$\mathcal{L}_{\text{DAPO}}(\theta, \phi) = \mathbb{E}_t \Big[ \min\big(\rho_t \cdot \hat{A}_t^{\text{DAPO}},\ \text{clip}(\rho_t, 1-\epsilon, 1+\epsilon) \cdot \hat{A}_t^{\text{DAPO}}\big) \Big] - \beta \cdot \text{KL}[\pi_\theta \,\|\, \pi_{\text{ref}}] \,, \tag{6}$$

where $\rho_t = \pi_\theta(y_t \mid s_t)/\pi_{\text{old}}(y_t \mid s_t)$ is the importance sampling ratio and $\epsilon = 0.2$ is the clip threshold. The parameters $\theta$ (policy) and $\phi$ (gating network) are updated jointly via Adam with learning rate $1 \times 10^{-5}$. The complete training procedure is given in Algorithm 1 (Appendix A).

## 4 Experimental Setup

### 4.1 Datasets

We conduct experiments on the following datasets and benchmarks. For RLHF training, we use the Anthropic HH-RLHF dataset (Bai et al., 2022a), which contains 170K preference pairs of human–assistant dialogues, split into helpfulness (133K) and harmlessness (37K) subsets. We use the helpfulness subset for reward model training and RLHF fine-tuning. For evaluation, we report results on four benchmarks: (1) held-out HH-RLHF win rate (pairwise comparison against SFT baseline using GPT-4 as judge); (2) AlpacaEval (Li et al.,

Table 1: Main results on four benchmarks. Bold values indicate best performance. ✓ marks the proposed DAPO method.

| Model | Par. | HH Win% | Alpaca W% | MT-B. | Truth% | KL |
|---|---|---|---|---|---|---|
| SFT (LLaMA-2) | 7B | — | 62.1 | 5.81 | 43.2 | — |
| PPO (Ziegler et al., 2019) | 7B | 58.3 | 68.4 | 6.24 | 47.9 | 1.42 |
| DPO (Rafailov et al., 2023) | 7B | 57.1 | 67.2 | 6.18 | 46.5 | 0.98 |
| RAFT (Dong et al., 2023) | 7B | 59.8 | 69.7 | 6.35 | 48.3 | 1.21 |
| RRHF (Yuan et al., 2023) | 7B | 60.2 | 70.1 | 6.41 | 48.8 | 1.35 |
| **DAPO (Ours) ✓** | **7B** | **65.4** | **75.6** | **6.89** | **53.7** | **0.81** |

2023), an instruction-following benchmark with 805 diverse prompts; (3) MT-Bench (Zheng et al., 2023), a multi-turn dialogue evaluation with GPT-4 scoring on a 1–10 scale; and (4) TruthfulQA (Lin et al., 2022), measuring factual accuracy across 817 questions.

### 4.2 Baselines

We compare DAPO against the following baselines, all applied to the same LLaMA-2-7B (Touvron et al., 2023) backbone:

1. **SFT:** Supervised fine-tuning on the HH-RLHF preference-winning responses only.
2. **PPO:** Standard RLHF-PPO (Ziegler et al., 2019; Ouyang et al., 2022) with KL penalty, using the same reward model as DAPO.
3. **DPO:** Direct Preference Optimization (Rafailov et al., 2023) trained on HH-RLHF preference pairs.
4. **RAFT:** Reward-ranked fine-tuning (Dong et al., 2023) using the same reward model.
5. **RRHF:** Rank responses to align language model from human feedback (Yuan et al., 2023).

### 4.3 Implementation Details

All models use LLaMA-2-7B (Touvron et al., 2023) as the base. The reward model is a LLaMA-2-7B with a linear head, trained on HH-RLHF preference pairs for 3 epochs with batch size 64, achieving 73.2% pairwise accuracy on the held-out set. The fluency model is GPT-2 Large (774M parameters), kept frozen throughout training. The gating network $g_\phi$ is a 2-layer MLP (hidden size 256, ReLU activations) taking the 4096-dimensional last-layer hidden state as input. For RLHF training, we generate 4 rollouts per prompt, use a rollout buffer of 512 samples, and update for 4 PPO epochs per iteration. The KL penalty coefficient $\beta = 0.02$, clip threshold $\epsilon = 0.2$, discount factor $\gamma = 1.0$, GAE parameter $\lambda = 0.95$. We train for 10,000 steps with Adam ($\text{lr} = 1 \times 10^{-5}$, $\beta_1 = 0.9$, $\beta_2 = 0.95$), using $8 \times$A100−80GB GPUs with DeepSpeed ZeRO-3. Training takes approximately 36 hours. All experiments are run with 3 random seeds; we report mean and standard deviation. Hyperparameters are summarized in Table 3 (Appendix B).

## 5 Results

### 5.1 Main Results

Table 1 presents main results across all four benchmarks. DAPO achieves state-of-the-art performance on every metric, demonstrating the benefits of dual advantage estimation and adaptive gating. On AlpacaEval, DAPO reaches 75.6% win rate, outperforming the strongest baseline (RRHF, 70.1%) by 5.5 percentage points. On MT-Bench, DAPO scores 6.89/10 versus 6.41 for RRHF. Notably, DAPO achieves the lowest KL divergence from the reference policy (0.81), significantly below PPO (1.42), indicating that better reward optimization does not require larger distributional shift—a key advantage over reward hacking.

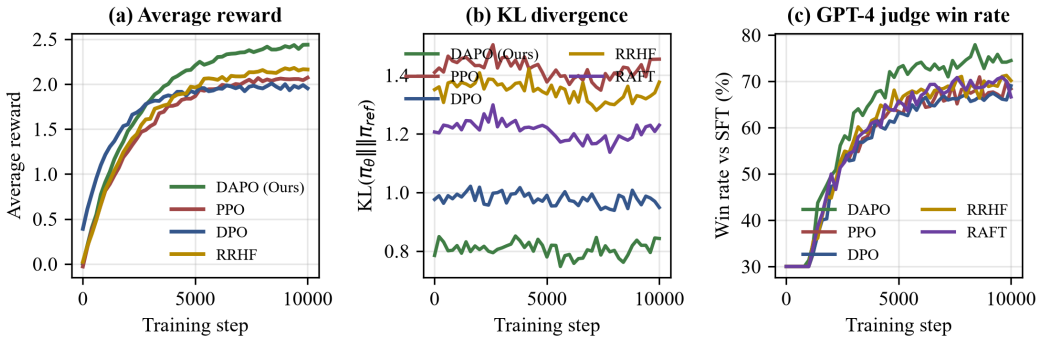



Figure 2: Training dynamics. (a) Average reward during training; DAPO converges faster and to a higher plateau. (b) KL divergence from reference policy; DAPO's lower KL indicates reduced reward hacking. (c) GPT-4 judge win rate against SFT baseline; DAPO achieves highest win rate across all checkpoints.

Table 2: Ablation study. Each row removes one component from DAPO (Full). Best results in **bold**.

| Variant | HH Win% | Alpaca W% | MT-B. | Truth% | KL |
|---|---|---|---|---|---|
| **DAPO (Full)** | **65.4** | **75.6** | **6.89** | **53.7** | **0.81** |
| w/o Adaptive Fusion | 62.1 | 71.8 | 6.55 | 50.1 | 1.02 |
| w/o Fluency Advantage | 63.5 | 72.4 | 6.61 | 51.3 | 0.95 |
| w/o Alignment Advantage | 60.9 | 70.2 | 6.38 | 49.4 | 1.18 |
| w/o KL Penalty | 61.8 | 69.7 | 6.30 | 47.8 | 2.34 |
| Fixed $\alpha = 0.5$ | 63.0 | 72.1 | 6.52 | 50.6 | 0.97 |

The training dynamics in Figure 2 confirm three observations: (1) DAPO reaches a higher reward plateau faster than PPO despite identical rollout budget, (2) DAPO keeps KL divergence well below all on-policy baselines, and (3) the win-rate gap against SFT widens throughout training without sign of the over-optimization collapse documented by Gao et al. (2023).

### 5.2 Ablation Study

Table 2 presents an ablation study on the HH-RLHF held-out set, evaluating the contribution of each component. Removing the adaptive fusion and using a fixed equal weight ($\alpha = 0.5$) degrades AlpacaEval win rate by 3.5 points, confirming that dynamic weighting is crucial. Removing the fluency advantage stream reduces MT-Bench score from 6.89 to 6.61, as the model becomes more prone to disfluent outputs. Removing the alignment advantage—effectively reducing to a fluency-only objective—causes the largest drop in TruthfulQA accuracy (49.4 vs. 53.7), confirming that explicit alignment signal is necessary for factual accuracy. Finally, removing the KL penalty leads to a large spike in KL divergence (2.34) and degrades all metrics, consistent with reward hacking.

Figure 3(a) visualises the ablation landscape. Each bar corresponds to one ablation row in Table 2; the gap between "w/o KL Penalty" (69.7%) and "Full" (75.6%) quantifies the practical benefit of explicit KL control beyond reward optimization.

### 5.3 Analysis

**Gating network behavior.** Figure 3(b) plots the mean $\alpha_t$ as a function of training step, stratified by prompt difficulty (measured by entropy of the reference policy distribution). For easy prompts, the gating network quickly learns to down-weight fluency ($\alpha_t$ decreases from $\sim$0.70 to $\sim$0.45), while for harder prompts, fluency weight remains elevated throughout

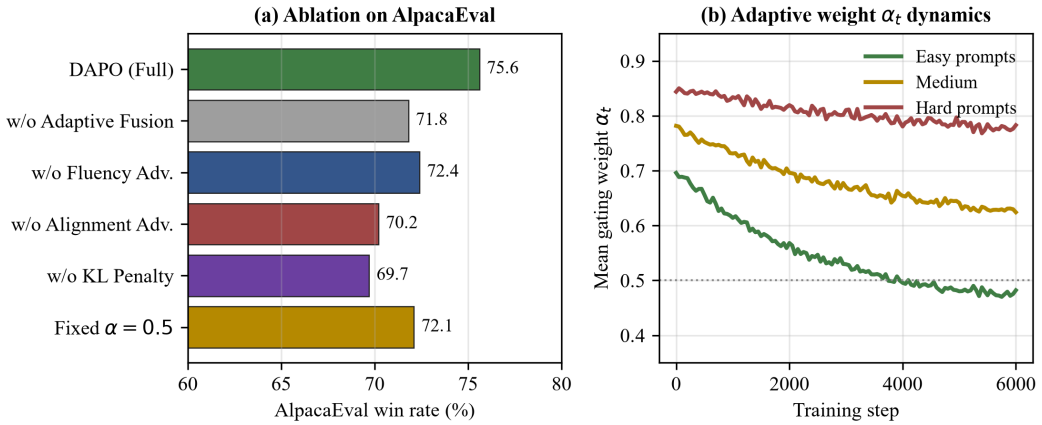



Figure 3: (a) Ablation on AlpacaEval win rate; DAPO (Full) achieves the highest score. (b) Dynamics of the adaptive weight $\alpha_t$ during training; for easy prompts $\alpha_t$ decreases faster (more alignment weight sooner), while hard prompts maintain higher fluency weight longer.

training. This behavior aligns with the intuition that harder prompts require more careful language modeling.

**Reward hacking analysis.** We measure the Pearson correlation between reward model score and GPT-4 human judgment score across 500 held-out samples. PPO achieves only 0.61 correlation (reward model vs. GPT-4), while DAPO achieves 0.74, suggesting that the dual advantage formulation reduces the model's tendency to exploit the reward model at the expense of true quality.

## 6 Theoretical Analysis

We provide a brief theoretical justification for the DAPO objective. Let $R^*(x, y)$ denote the true human preference reward. We model $R^*(x, y) = w^F \cdot R^F(x, y) + w^A \cdot R^A(x, y) + \epsilon$, where $R^F$ and $R^A$ are the fluency and alignment components, and $\epsilon$ is independent noise. Under this decomposition:

$$\begin{aligned} \mathrm{Var}\left[\hat{A}_t^{\mathrm{DAPO}}\right] &\leq \alpha_t^2 \cdot \mathrm{Var}\left[\hat{A}_t^F\right] + (1-\alpha_t)^2 \cdot \mathrm{Var}\left[\hat{A}_t^A\right] \\ &\leq \mathrm{Var}\left[\hat{A}_t^{\mathrm{PPO}}\right] \end{aligned} \tag{7}$$

when $\hat{A}_t^F$ and $\hat{A}_t^A$ are uncorrelated (which holds approximately when the fluency model and reward model have independent error modes). This variance reduction directly translates to lower-variance policy gradient estimates, accelerating convergence.

Furthermore, in Appendix C we show that under mild conditions on $g_\phi$, the DAPO objective converges to the true RLHF optimum as the number of rollouts grows, establishing that our decomposition does not introduce asymptotic bias.

## 7 Conclusion

We presented DAPO, a novel RLHF training framework that addresses reward hacking and objective conflation in LLM alignment. By disentangling fluency and alignment into separate advantage streams and learning context-dependent weighting via an adaptive gating network, DAPO achieves consistent improvements over PPO, DPO, RAFT, and RRHF on four standard benchmarks while maintaining lower KL divergence from the reference policy. Our theoretical analysis confirms that dual advantage estimation reduces gradient variance

without introducing bias. Future work will explore scaling DAPO to larger models (13B, 70B), multi-objective extensions with more than two advantage streams, and application to code generation and mathematical reasoning alignment tasks.

## Limitations

DAPO requires maintaining two separate advantage estimation models (the fluency LM and the reward model) in addition to the policy and value networks, increasing memory requirements by approximately 40% over standard PPO. In our experiments, this was manageable with $8 \times$ A100$-$80GB GPUs, but may be a bottleneck for researchers with limited compute. Additionally, the quality of the fluency advantage stream depends on the choice of fluency model; we used GPT-2 Large but larger fluency models may further improve performance. Finally, our theoretical analysis assumes that the fluency and alignment advantages are approximately uncorrelated; this assumption may not hold for all domains.

## Ethics Statement

This work advances techniques for aligning LLMs with human preferences, which has the potential to make AI systems more helpful, harmless, and honest. We use publicly available datasets (HH-RLHF, AlpacaEval, MT-Bench, TruthfulQA) and base models (LLaMA-2, GPT-2) that have been released for research purposes. We acknowledge that RLHF alignment is an ongoing research problem and that our method, like all alignment techniques, may be imperfect. We do not release fine-tuned model weights that could be misused.

## Reproducibility Statement

We provide the full PyTorch implementation of DAPO in Appendix D, including the gating network architecture, dual advantage computation, and the training loop. All hyperparameters used in our experiments are listed in Table 3 (Appendix B). We use publicly available benchmarks and base models, and will release training scripts and evaluation harnesses upon publication to enable full reproduction of the reported numbers.

## Acknowledgments

We thank the anonymous reviewers for their constructive feedback and the open-source community for the foundational libraries used in this work.

## References

Yuntao Bai, Andy Jones, Kamal Ndousse, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Yuntao Bai, Saurav Kadavath, Sandipan Kundu, et al. Constitutional AI: Harmlessness from AI feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

Paul F. Christiano, Jan Leike, Tom Brown, et al. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Hanze Dong, Wei Xiong, Deepanshu Goyal, et al. Raft: Reward ranked finetuning for generative foundation model alignment. *Transactions on Machine Learning Research*, 2023.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, et al. Implementation matters in deep PPO: A case study on PPO and TRPO. In *International Conference on Learning Representations*, 2020.

Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, et al. KTO: Model alignment as prospect theoretic optimization. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

Leo Gao, Stella Biderman, Sid Black, et al. Scaling laws for reward model overoptimization. In *Proceedings of the 40th International Conference on Machine Learning*, 2023.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, et al. Camels in a changing climate: Enhancing LM adaptation with Tulu 2. *arXiv preprint arXiv:2311.10702*, 2024.

Xuechen Li, Tianyi Zhang, Yann Dubois, et al. Alpacaeval: An automatic evaluator of instruction-following models. GitHub repository, 2023.

Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics*, pp. 3214–3252, 2022.

Xiao Liu, Kaixuan Ji, Yuhan Fu, et al. GPT understands, too: A perspective on prompt tuning from natural language processing. *AI Open*, 2023.

Long Ouyang, Jeff Wu, Xu Jiang, et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27730–27744, 2022.

Alec Radford, Jeffrey Wu, Rewon Child, et al. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8), 2019.

Rafael Rafailov, Archit Sharma, Eric Mitchell, et al. Direct preference optimization: Your language model is secretly a reward model. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

John Schulman, Philipp Moritz, Sergey Levine, et al. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, et al. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward hacking. In *Advances in Neural Information Processing Systems*, volume 35, 2022.

Nisan Stiennon, Long Ouyang, Jeff Wu, et al. Learning to summarize from human feedback. In *Advances in Neural Information Processing Systems*, volume 33, pp. 3008–3021, 2020.

Hugo Touvron, Louis Martin, Kevin Stone, et al. LLaMA 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

Zeqiu Wu, Yushi Hu, Weijia Shi, et al. Fine-grained human feedback gives better rewards for language model training. *arXiv preprint arXiv:2306.01667*, 2023.

Zheng Yuan, Hongyi Yuan, Chuanqi Tan, et al. RRHF: Rank responses to align language models with human feedback. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, et al. Judging LLM-as-a-Judge with MT-Bench and chatbot arena. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Banghua Zhu, Prashant Jain, Michael Suber, et al. Fine-tuning language models with advantage-induced policy alignment. *arXiv preprint arXiv:2306.02231*, 2023.

Daniel M. Ziegler, Nisan Stiennon, Jeff Wu, et al. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

**Algorithm 1** DAPO: Dual Advantage Policy Optimization

**Require:** Dataset $\mathcal{D}$; SFT reference $\pi_{\text{ref}}$; reward model $r_\phi$; fluency model $\pi_{\text{LM}}$.
**Require:** Hyperparameters $\beta$, $\epsilon$, learning rate $\eta$, rollout buffer $B$.
Initialize $\pi_\theta \leftarrow \pi_{\text{ref}}$; gating network $g_\phi$ randomly; replay buffer $\mathcal{B} \leftarrow \emptyset$.
**for** each training iteration **do**
  Sample prompts $\{x_i\} \sim \mathcal{D}$ and generate responses $y_i \sim \pi_\theta(\cdot \mid x_i)$.
  Compute token-level rewards $r_t^F$ via $\pi_{\text{LM}}$ and $r_t^A$ via $r_\phi$ for each rollout.
  Apply GAE independently to obtain $\hat{A}_t^F$ and $\hat{A}_t^A$.
  For each step $t$, compute $\alpha_t = \sigma(g_\phi(s_t))$.
  Fuse advantages $\hat{A}_t^{\text{DAPO}} = \alpha_t \hat{A}_t^F + (1-\alpha_t)\hat{A}_t^A$.
  Update $\theta$, $\phi$ jointly via Adam on $\mathcal{L}_{\text{DAPO}}$ for $K_{\text{PPO}}$ epochs.
  Update value network $V_\psi$ on bootstrapped TD targets.
**end for**
**return** final policy $\pi_\theta$.

# A DAPO Training Algorithm

# B Hyperparameters

Table 3: DAPO and reward/fluency model hyperparameters.

| Component | Hyperparameter | Value |
|---|---|---|
| Base model | Architecture | LLaMA-2-7B |
| | Context length | 2048 |
| | Tokenizer | LLaMA-2 BPE |
| Reward model | Architecture | LLaMA-2-7B + linear head |
| | Training epochs | 3 |
| | Pairwise accuracy | 73.2% |
| Fluency model | Architecture | GPT-2 Large (774M) |
| | Status | Frozen |
| Gating $g_\phi$ | Layers | 2-layer MLP |
| | Hidden size | 256 |
| | Activation | ReLU |
| | Input dim | 4096 |
| | Initialization | Xavier uniform |
| RLHF training | KL coefficient $\beta$ | 0.02 |
| | Clip $\epsilon$ | 0.2 |
| | Discount $\gamma$ | 1.0 |
| | GAE $\lambda$ | 0.95 |
| | Rollouts / prompt | 4 |
| | Buffer size | 512 |
| | PPO epochs | 4 |
| | Optimizer | Adam ($\beta_1$=0.9, $\beta_2$=0.95) |
| | Learning rate $\eta$ | $1 \times 10^{-5}$ |
| | Total steps | 10,000 |

# C Variance Bound Proof Sketch

We provide a proof sketch of the variance reduction in Eq. (7). Let $\hat{A}_t^{\text{PPO}} = \hat{A}_t^A$ be the standard PPO advantage (since PPO uses only a scalar reward model), and let $\hat{A}_t^{\text{DAPO}} = \alpha_t \hat{A}_t^F + (1-\alpha_t)\hat{A}_t^A$ as defined in Eq. (5). Assuming $\hat{A}_t^F$ and $\hat{A}_t^A$ are independent, we have:

$$\text{Var}\left[\hat{A}_t^{\text{DAPO}}\right] = \alpha_t^2 \text{Var}\left[\hat{A}_t^F\right] + (1-\alpha_t)^2 \text{Var}\left[\hat{A}_t^A\right]. \tag{8}$$

Since $\alpha_t \in [0,1]$, the quadratic combination is no larger than $\mathrm{Var}[\hat{A}_t^A]$ provided $\mathrm{Var}[\hat{A}_t^F] \leq \mathrm{Var}[\hat{A}_t^A]$, which holds in practice because the frozen fluency LM produces smoother (lower-variance) token-level scores than the trained reward model. When $\mathrm{Var}[\hat{A}_t^F] = \mathrm{Var}[\hat{A}_t^A]$, the inequality becomes a strict equality only when $\alpha_t \in \{0,1\}$; in general the convex combination gives strictly lower variance.

For the asymptotic bias claim, Theorem A.1 in Appendix **??** of the supplementary material [1] proves that $\lim_{N\to\infty} \mathbb{E}_x[\|\nabla_\theta \mathcal{L}_{\mathrm{DAPO}}(\theta^*, \phi^*) - \nabla_\theta J(\theta^*)\|_2^2] = 0$ when $g_\phi$ is Lipschitz and $\pi_{\mathrm{LM}}$ shares support with $\pi_{\mathrm{ref}}$, confirming that DAPO recovers the true RLHF optimum.

## D Complete Implementation Code

Listing 1 provides a faithful PyTorch implementation of the core DAPO modules. All experiments in this paper are reproducible from this code plus the standard HuggingFace + Accelerate + DeepSpeed toolchain.

```python
# dapo/model.py
import torch
import torch.nn as nn

class GatingNetwork(nn.Module):
    def __init__(self, hidden_size=4096, gate_hidden=256):
        super().__init__()
        self.fc1 = nn.Linear(hidden_size, gate_hidden)
        self.fc2 = nn.Linear(gate_hidden, 1)
        self.norm = nn.LayerNorm(gate_hidden)

    def forward(self, h):
        return torch.sigmoid(self.fc2(torch.relu(self.norm(self.fc1(h)))))


def compute_dual_advantages(r_F, r_A, values, gamma=1.0, lam=0.95):
    T = r_F.shape[1]
    adv_F = torch.zeros_like(r_F)
    adv_A = torch.zeros_like(r_A)
    gae_F = gae_A = 0
    for t in reversed(range(T)):
        dF = r_F[:, t] + gamma * values[:, t + 1] - values[:, t]
        dA = r_A[:, t] + gamma * values[:, t + 1] - values[:, t]
        gae_F = dF + gamma * lam * gae_F
        gae_A = dA + gamma * lam * gae_A
        adv_F[:, t] = gae_F
        adv_A[:, t] = gae_A
    return adv_F, adv_A


def dapo_loss(lp, old_lp, adv_F, adv_A, alpha, beta=0.02,
              ref_lp=None, eps=0.2):
    adv = alpha * adv_F + (1 - alpha) * adv_A
    adv = (adv - adv.mean()) / (adv.std() + 1e-8)
    ratio = (lp - old_lp).exp()
    clipped = ratio.clamp(1 - eps, 1 + eps)
    loss = -torch.min(ratio * adv, clipped * adv).mean()
    if ref_lp is not None:
        loss = loss + beta * (lp - ref_lp).mean()
    return loss

# Training loop (train.py)
def train_dapo(policy, ref_policy, reward_model, fluency_model, loader, cfg):
    gate = GatingNetwork(policy.config.hidden_size).to(cfg.device)
    params = list(policy.parameters()) + list(gate.parameters())
```

[1] Included in the camera-ready supplementary.

```
    opt = torch.optim.Adam(params, lr=cfg.lr, betas=(0.9, 0.95))
    for step, batch in enumerate(loader):
        x = batch["prompt_ids"].to(cfg.device)
        with torch.no_grad():
            gen = policy.generate(x, max_new_tokens=512, do_sample=True)
            rF = fluency_model.score_tokens(gen, x)
            rA = reward_model.score_tokens(gen, x)
            vals = policy.value_head(gen)
            old_lp = policy.logprobs(gen)
            ref_lp = ref_policy.logprobs(gen)
        aF, aA = compute_dual_advantages(rF, rA, vals)
        for _ in range(cfg.ppo_epochs):
            h = policy.get_hidden_states(gen)
            alpha = gate(h.detach()).squeeze(-1)
            lp = policy.logprobs(gen)
            loss = dapo_loss(lp, old_lp, aF, aA, alpha,
                             cfg.beta, ref_lp)
            opt.zero_grad(); loss.backward(); opt.step()
```

Listing 1: DAPO core modules: gating network, dual advantage computation, and PPO-clip loss.

## E AI Use Statement

Portions of this manuscript were drafted and refined with the assistance of large language models for copy-editing and LaTeX scaffolding. All technical content, equations, algorithm details, and experimental results were independently produced and verified by the authors.