Accelerated Preference Optimization for Large Language Model Alignments

Jiafan He and **Huizhuo Yuan** and **Quanquan Gu**

Department of Computer Science, University of California, Los Angeles

Momentum in Preference Optimization

Reinforcement Learning from Human Feedback via direct preference optimization has established itself as a crucial methodology for language model alignment. While momentum-based acceleration has demonstrated benefits in optimization theory, its theoretical foundations and practical applications in preference optimization remain unexplored.

Our Goal: Establish a momentum-based acceleration framework for preference optimization and validate its efficiency through large-scale language model experiments.

Setup

- **Setting for the Optimization Porblem:**
  - Context set $\mathcal{X}$ and response set $\mathcal{Y}$
  - Policy $\pi : \mathcal{X} \to \Delta(\mathcal{Y})$ maps prompts to response distributions
- **Preference Collection Process:**
  - Sample context x from distribution ρ
  - Generate responses (y_1, y_2) from reference policy μ
  - Collect preference feedback $(y^w \succ y^l)$
- **Latent Preference Model:**
  - Bradley-Terry model with latent reward $r^*(x, y)$

$$P(y_1 \succ y_2|x) = \frac{\exp(r^*(x, y_1))}{\exp(r^*(x, y_1)) + \exp(r^*(x, y_2))}$$

Iterative Preference Optimization Framework

- **Policy Update Process:** For each iteration $t \in [T]$:
  1. Update reward model with current policy π_t:
  $$r_t(\cdot,\cdot) \leftarrow \arg\max_r \mathbb{E}_{\mathcal{D}_t}[\ell(r, x, y^w, y^l, \pi_t)]$$
  where the loss function ℓ can be one of:
     - Direct Preference Optimization (DPO):
     $$\ell_{\mathsf{DPO}}(r_\pi, x, y^w, y^l, \pi_t) = -\log\sigma(r_\pi(x, y^w) - r_\pi(x, y^l))$$
     - Self-Play Preference Optimization (SPPO):
     $$\ell_{\mathsf{SPPO}}(r_\pi, x, y^w, y^l, \pi_t) = \frac{1}{2}(r_\pi(x, y^w) - 1 + \log Z_{\pi_t}(x))^2 + \frac{1}{2}(r_\pi(x, y^l) + \log Z_{\pi_t}(x))^2$$
     - Identity Preference Optimization (IPO):
     $$\ell_{\mathsf{IPO}}(r_\pi, x, y^w, y^l, \pi_t) = (r_\pi(x, y^w) - r_\pi(x, y^l) - \tau^{-1})^2$$
  2. Optimize policy with KL regularization:
  $$\widehat{\pi}_{t+1} \leftarrow \arg\max_\pi \mathbb{E}_{\rho,\pi}[r_t] - \beta\mathbb{E}_\rho[\mathsf{KL}(\pi\|\pi_t)]$$
- **Direct Preference Optimization:**
  - **Reparameterize Reward** instead of reward model:
  $$r_\pi(x, y) = \beta\log\frac{\pi(y|x)}{\pi_t(y|x)}$$
  - One-step optimization:
  $$\widehat{\pi}_{t+1} \leftarrow \arg\min_{r_\pi} \mathbb{E}_{\mathcal{D}_t}[\ell(r_\pi, x, y^w, y^l, \pi_t)]$$
- **Proximal Point Method:** The iterative optimization resembles **Bregman Proximal Point Method**:
$$\pi_{t+1} \leftarrow \arg\min_\pi\{L_t(\pi) + \beta D(\pi, \pi_t)\}$$
where $L_t(\pi)$ corresponds to expected reward and $D(\pi, \pi_t)$ is KL divergence

Accelerated Preference Optimization(APO)

- **Motivation:** Nesterov's momentum method and Catalyst framework which accelerates proximal point method
- **Catalyst Framework: Extrapolation** after update:
$$x_{t+1} = \arg\min_x\{f(x) + \kappa D(x, y_t)\}$$
$$y_{t+1} = x_{t+1} + \alpha_t(x_{t+1} - x_t)$$
- **Extrapolation on Preference Optimization:** After policy update, apply **Momentum**:
$$\log\pi_{t+1}(y|x) = \log\widehat{\pi}_{t+1} + \alpha(\log\widehat{\pi}_{t+1} - \log\widehat{\pi}_t)$$
$$\pi_{t+1}(y|x) \propto \widehat{\pi}_{t+1}(y|x)\cdot(\widehat{\pi}_{t+1}(y|x)/\widehat{\pi}_t(y|x))^\alpha$$
- **Implementation:** Update reduces to parameter momentum when policy is softmax parameterized

Theoretical Results

- **Main Results:** Under mild assumptions of realizability and boundedness, APO achieves sub-optimality gap:
$$\mathbb{E}_{x\sim\rho,y\sim\pi^*,y'\sim\widehat{\pi}_{T+1}}\big[r^*(x, y) - r^*(x, y')\big] \le \widetilde{O}\big((1-\alpha)\beta/T\big)$$
- **Enhanced Convergence:** With additional minimal sub-optimality gap, both DPO and SPPO loss converge:
$$\mathbb{E}_{x\sim\rho}[\mathsf{D_{TV}}(\widehat{\pi}_{T+1}, \pi^*)] \le \exp(-O(T/(1-\alpha)))$$
- Acceleration factor $(1-\alpha)$ improves upon vanilla methods in both case after introducing the momentum

Experimental Results

- **Evaluation Metrics:**
  - LC Win Rate: Length-controlled win rate in head-to-head comparisons with Claude-2
  - MT-Bench: Average scores on 8 multi-turn conversation tasks
  - Five Tasks: Performance on training-relevant dimensions (Writing, Roleplay, Extraction, STEM, Humanities)
- **Results Summary:**

Method	LC Win Rate	MT-Bench	Five Tasks
Base	17.11	7.64	9.14
DPO (3 iter)	27.32	7.43	9.14
APO (3 iter)	**31.73**	7.53	**9.57**

- **Key Findings:**
  - APO achieves 31.73 % win rate, improving DPO by 4.41%
  - Strong performance on training-specific domains (9.57/10)

Key Takeaways

- APO introduces theoretically-grounded acceleration to preference optimization
- Maintains strong performance across diverse tasks
- Framework generalizes to multiple loss functions

Get the Paper