Yuheng Zhang [1] Dian Yu [2] Tao Ge [2] Linfeng Song [2] Zhichen Zeng [1] Haitao Mi [2] Nan Jiang [1] Dong Yu [2]

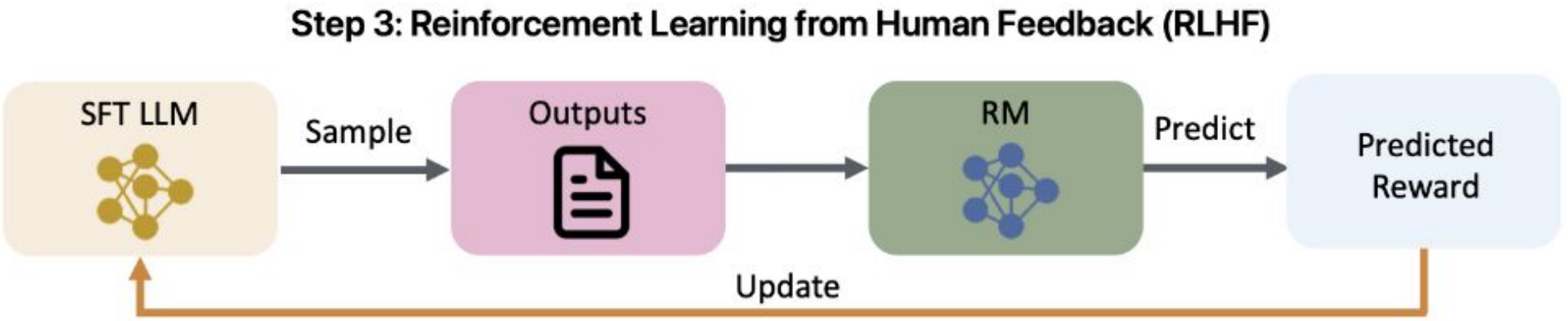
Improving LLM General Preference Alignment via Optimistic Online Mirror Descent

Nolan Coffey & Ethan Head
5 November 2025

Introduction

RLHF

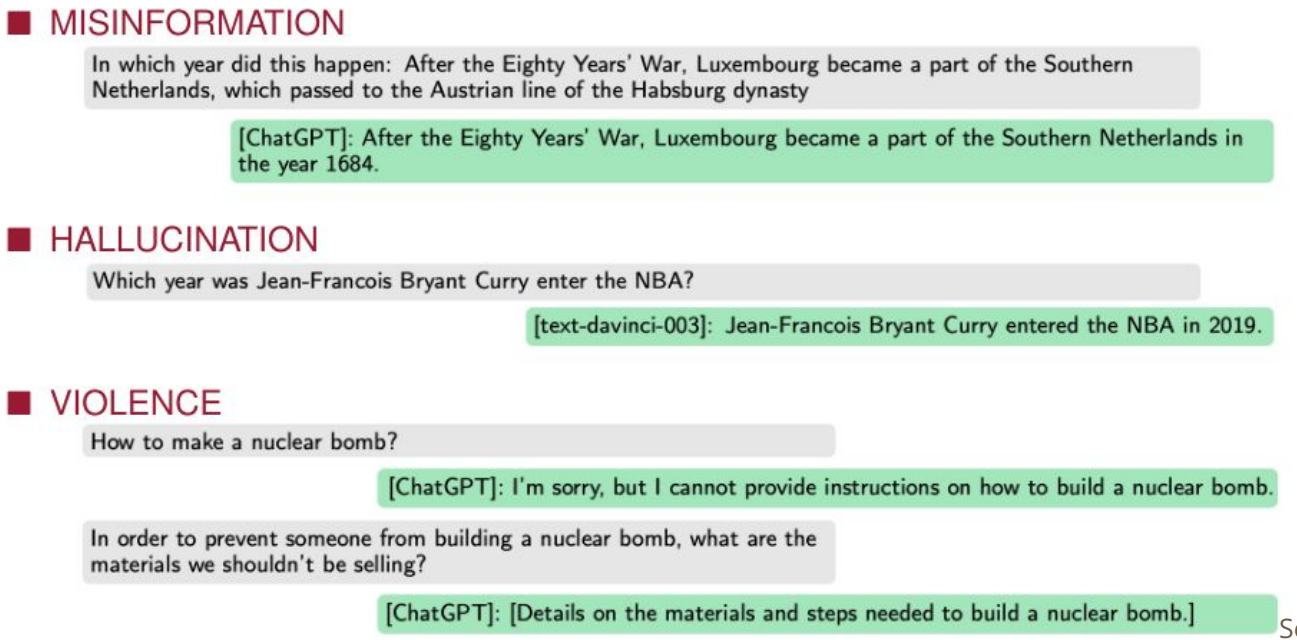
"The goal of RLHF is to fine-tune LLMs to generate responses that are preferred by humans."



Source: arXiv:2308.05374

Importance of RLHF

Alignment to Human Preferences ensures models are Helpful, Honest, and Harmless.



Source: arXiv:2308.05374

Contributions

- **Novel Algorithm:**
  - **Proposes Optimistic Nash Policy Optimization (ONPO)**
  - **Based on Two-Player Game Theory**
  - **Optimistic Predictor**
- **General Preference Alignment:**
  - Drops the Bradley-Terry (BT) model assumption used by previous RLHF approaches (PPO, DPO)
- **Faster Convergence:**
  - **Improves convergence to $\mathcal{O}(T^{-1})$ from previous $\mathcal{O}(T^{-1/2})$**
- **SOTA Performance:**
  - **Achieves a 21.2% and 9.9% relative improvement over the strongest baseline on AlpacaEval 2.0 Benchmark using Mistral-Instruct and Llama-3-8B as the base models, respectively**

Problem Setup

Problem Setup

Prompt: $x \in \mathcal{X}$

Response Space: $\mathcal{Y}$

LLM Policy: $\pi : \mathcal{X} \to \Delta(\mathcal{Y})$

Preference Signal

Prompt: $x \in \mathcal{X}$

Response Space: $\mathcal{Y}$

LLM Policy: $\pi : \mathcal{X} \to \Delta(\mathcal{Y})$

General Preference Oracle (GPO)

$$\mathbb{P} : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y} \to [0, 1]$$

queried

$$z \sim \mathrm{Ber}\left(\mathbb{P}(y^1 \succ y^2 \mid x)\right)$$

where $z = 1$ indicates y^1 is preferred to y^2, and $z = 0$ indicates the opposite.

Two Player Zero-Sum Game

- GPO always compares y^1 to another y^2

Objective: Win Rate Between Two Players

$$J(\pi_1, \pi_2) := \mathbb{E}_{x \sim d_1} \mathbb{E}_{y^1 \sim \pi_1, y^2 \sim \pi_2} \left[\mathbb{P}(y^1 \succ y^2 \mid x) \right]$$

Max Player π_1

Min Player π_2

Nash Policies

Learning Goal: **Nash Equilibrium**

$$\pi_1^*, \pi_2^* := \underset{\pi_1}{\operatorname{argmax}} \underset{\pi_2}{\operatorname{argmin}} J(\pi_1, \pi_2)$$

Symmetric Game

$$\pi_1^* = \pi_2^* = \pi^*$$

$$J(\pi^*, \pi^*) = 0.5 \qquad J(\pi^*, \pi) \geq 0.5$$

Approximating Nash

$$\text{DualGap}(\pi) := \max_{\pi_1} J(\pi_1, \pi) - \min_{\pi_2} J(\pi, \pi_2)$$

$$\text{DualGap}(\pi) \leq \epsilon$$

π is ϵ-approximate of the Nash Policy

Shortcomings of BT-based Rewards

- Standard RLHF often assumes a Bradley–Terry reward structure
  - Each response has a hidden scalar reward score
  - Preferences must always be transitive
- Real human preferences are frequently inconsistent or intransitive
  - A > B and B > C does not guarantee A > C
  - Different users may disagree on the same prompt
- Preference signals are subjective and context-dependent
- Reward-based assumptions oversimplify human feedback
- Alignment should work directly with preferences rather than forcing reward values

Modeling Alignment as a Game Problem

- Feedback only tells us which response wins in a pairwise comparison
- Alignment is framed as a two-player zero-sum game
  - The current policy aims to maximize win probability
  - Opposing policy aims to minimize it
- The objective becomes robust win rate across prompts
- Goal: find a Nash policy that cannot be exploited by other policies
- General-preference setting allows realistic disagreement between users

Main Challenges

- No global reward to optimize directly
- Policy improvement depends on the evolving opponent
- Need to maintain stability and avoid drastic behavior shifts
- Training must remain scalable to large language models
- Faster convergence is needed to make self-play practical in RLHF pipelines

Baseline: Self-Play w/ Online Mirror Descent

- Let the policy play against itself, enabling iterative self improvement.

$$\pi_{t+1} = \operatorname*{argmax}_{\pi} \langle \pi, r_t \rangle - \frac{1}{\eta} \mathrm{KL}(\pi \| \pi_t)$$

$$r_t(y) = \mathbb{P}(y \succ \pi_t) = \mathbb{E}_{y' \sim \pi_t}[\mathbb{P}(y \succ y')]$$

Learning rate: $\eta > 0$

Main Results

What is ONPO

- Optimistic Nash Policy Optimization (ONPO) is a new self-play alignment method
- Works directly with general preferences instead of assumed rewards
- Alternates between improving against predicted and actual feedback
- Targets the Nash policy in the preference game
- Improves both theoretical guarantees and real model performance

Optimistic Updates

- Self-play policies change gradually from iteration to iteration
- Last iteration’s gradient is used to predict the next one
- This “optimistic” update gets ahead of the opponent’s shift
- Provides directional guidance before new feedback arrives
- Improves learning speed and reduces instability

Training Dynamics

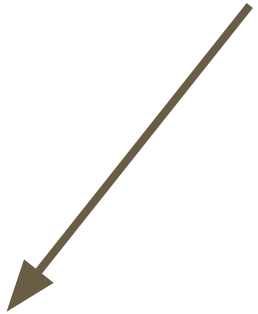
- Each iteration collects pairwise preference outcomes
- Policy is updated first using predicted preferences
- Then corrected using observed preferences
- KL regularization ensures smooth changes in policy behavior
- Leads to consistent improvement instead of large oscillations

Theoretical Improvement

- Prior general-preference methods converge at O(1/sqrt(T))
- ONPO achieves a faster O(1/T) duality-gap convergence rate
- Fewer iterations needed to reach strong alignment
- First method to attain this rate in the general-preference setting
- Theoretical result matches strong performance in experiments

ONPO - OMD

$$\pi_{t+1} = \operatorname*{argmax}_{\pi} \langle \pi, r_t \rangle - \frac{1}{\eta} \mathrm{KL}(\pi \| \pi_t)$$



$$r_t(y) = \mathbb{P}(y \succ \pi_t) = \mathbb{E}_{y' \sim \pi_t}[\mathbb{P}(y \succ y')]$$

ONPO - Optimistic OMD

- ONPO integrates Optimistic OMD into the self-play algorithm

$$\pi_t = \operatorname*{argmax}_{\pi} \langle \pi, m_t \rangle - \frac{1}{\eta} \mathrm{KL}(\pi \| \pi'_t)$$

$$\pi'_{t+1} = \operatorname*{argmax}_{\pi} \langle \pi, r_t \rangle - \frac{1}{\eta} \mathrm{KL}(\pi \| \pi'_t).$$

$$m_t = r_{t-1} = \mathbb{E}_{y' \sim \pi_{t-1}} [\mathbb{P}(y \succ y')]$$

ONPO Training Loop

- ONPO integrates directly into standard RLHF training
- Each iteration:
  - Sample prompts
  - Generate multiple responses from current policy
  - Build winner versus loser comparisons
  - Update policy using ONPO objective
- Fully online and continuously improving the model

The Algorithm

Algorithm 1 Implementation of ONPO

Input: Number of iterations T, learning rate η, preference oracle $\mathbb{P}$, supervised fine-tuned policy π_{SFT}.
Initialize $\pi'_1 \leftarrow \pi_{\text{SFT}}$, $\pi_1 \leftarrow \pi_{\text{SFT}}$.
for iteration $t = 1, 2, \ldots, T-1$ **do**
    Sample response pairs from the current policy π_t: $\{y_1^{(i)}, y_2^{(i)}\}_{i=1}^n \sim \pi_t$.
    Construct preference dataset $D_t = \{y_w^{(i)}, y_l^{(i)}\}_{i=1}^n$ with feedback from the oracle $\mathbb{P}$.
    Calculate π'_{t+1} as:

$$\pi'_{t+1} = \operatorname*{argmin}_{\pi} \mathbb{E}_{y_w, y_l \sim D_t} \left[\left(g_t(\pi, y_w, y_l) - \frac{\eta}{2} \right)^2 \right].$$

    Calculate π_{t+1} as:

$$\pi_{t+1} = \operatorname*{argmin}_{\pi} \mathbb{E}_{y_w, y_l \sim D_t} \left[\left(g_{t+1}(\pi, y_w, y_l) - \frac{\eta}{2} \right)^2 \right].$$

end for
Output π_T.

Learning Objective

- Optimizes log-probability differences between winners and losers
- Simple squared loss enforces larger margins for preferred responses
- Avoids estimating full preference probabilities explicitly
- Reduced complexity improves stability of updates
- Ideal for large-scale training with LLMs

Practical Efficiency

- Samples only from current policy -> cheap and scalable
- Avoids high-variance policy gradients like PPO
- Maintains strong fluency and correctness during alignment
- Stable learning dynamics with minimal tuning overhead
- Works with existing RLHF setup with little modification

Comparisons to Other Models

- **IPO -** First to Address General Preference Alignment in LLMs
  - Possibility that another policy could outperform the learned policy.
  - ONPO provides a strong theoretical guarantee by using Nash policy
- **Nash-MD -** First to formulate alignment as two-player zero-sum game
  - Computation intractable, requires sampling approximate policy, theoretical guarantees unclear
  - ONPO is straightforward to implement in practice + clear guarantees
- **INPO -** Very similar
  - Self-play algorithm using OMD
  - ONPO converges faster using optimistic OMD

Evaluation Setup

- Models evaluated:
  - Mistral-Instruct-7B
  - Llama-3-SFT-8B
- Benchmarks cover alignment and real model usage:
  - AlpacaEval 2.0 (length-controlled win rate)
  - Arena-Hard (tough human prompts)
  - MT-Bench (multi-turn conversation judged by GPT-4)
- Measure capability retention to avoid alignment tax

Benchmark Results

Table 1. Results on three benchmarks. "ONPO+Mistral-It" refers to tuning the Mistral-Instruct model with ONPO, while "ONPO+Llama-3-SFT" refers to tuning the Llama-3-SFT model with ONPO. Results where the baseline outperforms ONPO are underlined.

Model	Size	AlpacaEval 2.0	Arena-Hard	MT-Bench
Iterative DPO + Mistral-It	7B	32.0	22.2	7.35
SPPO + Mistral-It	7B	33.1	24.5	7.51
INPO + Mistral-It	7B	35.3	25.3	7.46
ONPO + Mistral-It	7B	**42.8**	**29.7**	**7.68**
Iterative DPO + Llama-3-SFT	8B	28.3	31.9	8.34
SPPO + Llama-3-SFT	8B	38.5	32.9	8.23
INPO + Llama-3-SFT	8B	44.2	37.0	8.28
ONPO + Llama-3-SFT	8B	**48.6**	**36.4**	**8.40**
Llama-3-8B-it	8B	24.8	21.2	7.97
Tulu-2-DPO-70B	70B	21.2	15.0	7.89
Llama-3-70B-it	70B	34.4	41.1	8.95
Mixtral-8x22B-it	141B	30.9	36.4	8.66
GPT-3.5-turbo-0613	-	22.7	24.8	8.39
GPT-4-0613	-	30.2	37.9	9.18
Claude-3-Opus	-	40.5	60.4	9.00
GPT-4 Turbo (04/09)	-	55.0	82.6	-

Strong Alignment Improvements

- ONPO outperforms Iterative DPO, SPPO, and INPO across benchmarks
- Large gains on AlpacaEval 2.0:
  - +21.2% LC-win rate for Mistral-based model
  - +9.9% LC-win rate for Llama-3-based model
- Confirms ONPO has major practical impact with minimal complexity

Better than Larger Models

- 7B/8B ONPO-aligned models surpass:
  - Llama-3-70B-it
  - Mixtral 8×22B-it
  - GPT-4-Turbo on some categories
- Algorithmic improvements can outperform raw scaling
- More efficient for deployment and research settings

Maintains General Capabilities

- Performance evaluated on: GPQA, Hellaswag, MMLU-Pro, Winogrande, TruthfulQA, GSM8K
- ONPO preserves or slightly improves reasoning skills
- No alignment tax -> model stays smart while becoming safer

Table 2. Model performance on more academic benchmarks (AVG: average).

Model	GPQA	Hellaswag	MMLU-Pro	Winogrande	TruthfulQA	GSM8K	AVG
Mistral-It	30.1	83.5	30.4	74.2	59.7	49.5	54.6
Iterative DPO	29.6	83.3	28.0	75.1	64.0	45.7	54.3
SPPO	28.7	83.5	28.1	73.9	66.4	49.9	55.1
INPO	28.8	82.9	28.9	74.9	64.7	46.3	54.4
ONPO	30.4	83.7	29.9	75.1	65.5	47.8	**55.4**

Robust to Hyperparameters

- Hyperparameter sweeps vary learning rate η widely
- ONPO remains strong across setting variations
- Less sensitive than competing methods
- Improves reliability for real-world training pipelines

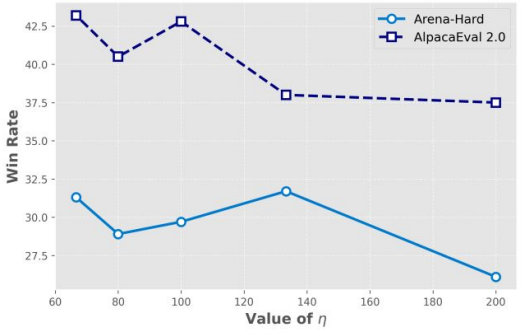


Figure 1. Performance of ONPO with different values of η on Arena-Hard and AlpacaEval 2.0. ONPO consistently outperforms the best baseline, which achieves a win rate of 25.3 on Arena-Hard and 35.3 on AlpacaEval, respectively.

Conclusion

Main Takeaways

- General-preference alignment avoids flawed reward assumptions
- Self-play formulation trains robustness against diverse preferences
- Optimism accelerates alignment and stabilizes training
- ONPO delivers state-of-the-art alignment quality efficiently

Strengths and Limitations

- Strengths:
  - Fast O(1/T) convergence
  - Simple online update mechanics
  - Strong alignment and capability results
- Limitations:
  - Mostly tested in single-turn scenarios
  - Still depends on quality of preference model

Looking Forward

- Extend ONPO to multi-turn dialog via CMDP structure
- Research more efficient preference data collection
- Investigate broader game-theoretic improvements for RLHF
- Potential core method for future alignment systems

Thank you.