# Self-Play Preference Optimization for Language Model Alignment

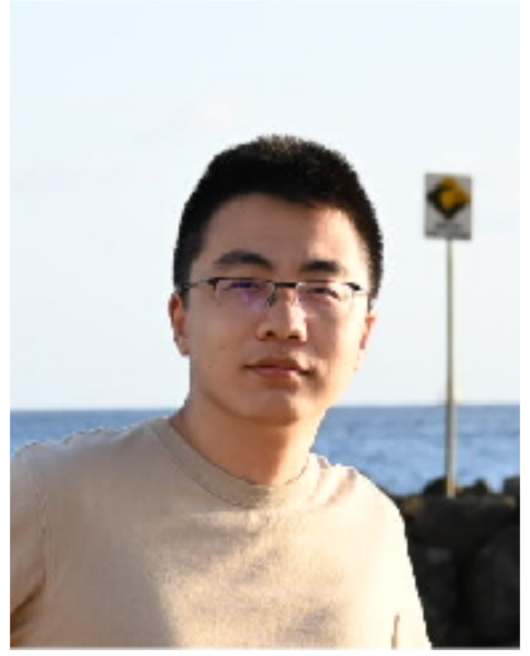
**Quanquan Gu**

Department of Computer Science, UCLA

Workshop on Applied Algorithms for Machine Learning

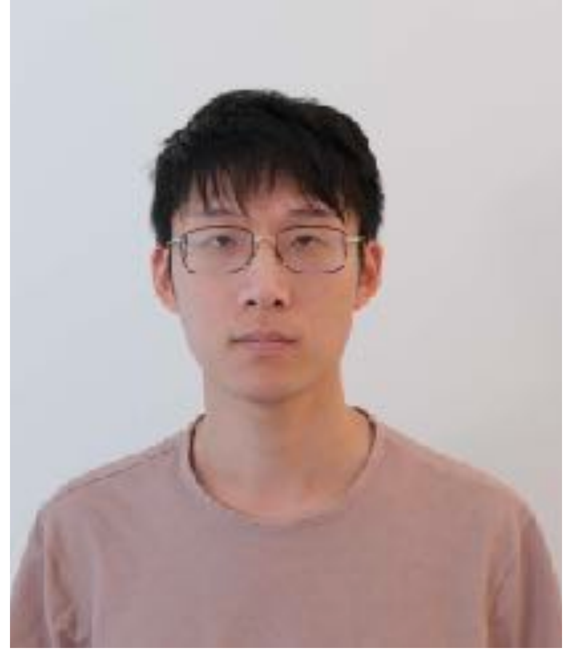
June 10, 2024

# Acknowledgment

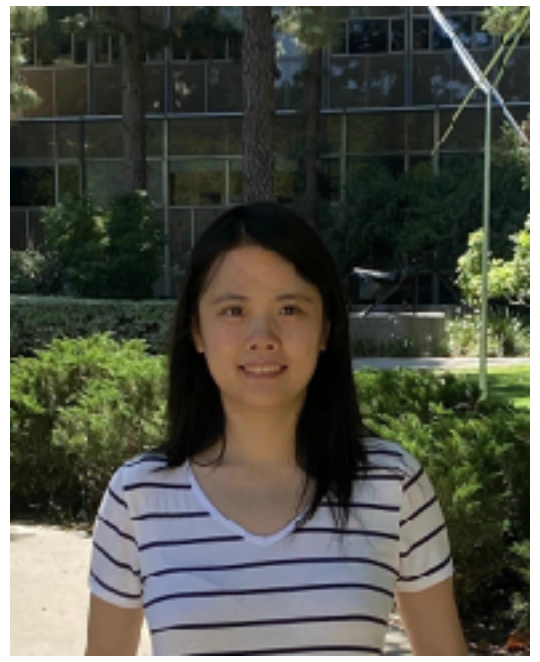
Yue Wu*

UCLA



Zhiqing Sun*

CMU



Huizhuo Yuan*

UCLA

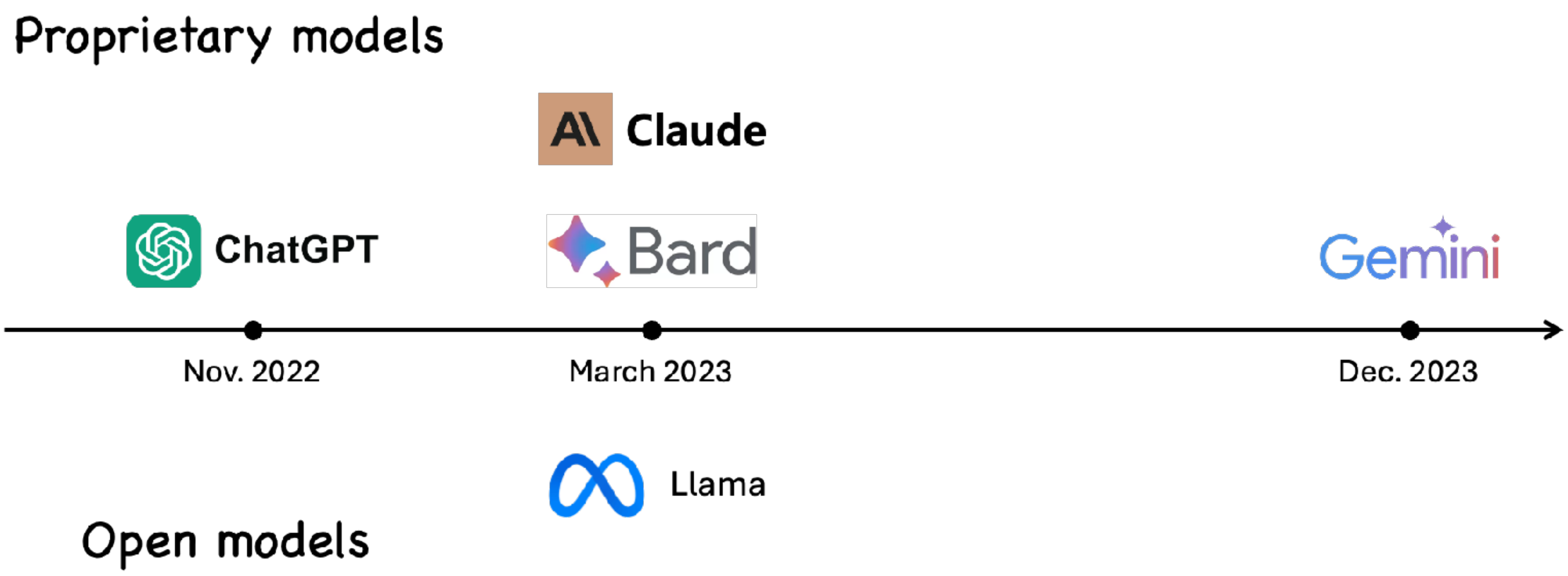
Kaixuan ji

UCLA

Yiming Yang

CMU

# The Surge of Large Large Models



# Fine-Tuning Pre-Trained LLMs for Specific Tasks

Supervised Fine-Tuning (SFT)

- High-quality demonstration dataset curated from humans responses or advanced LLM generations.
- Model is typically fine-tuned with supervised learning objective.

RL from Human Feedback (RLHF)

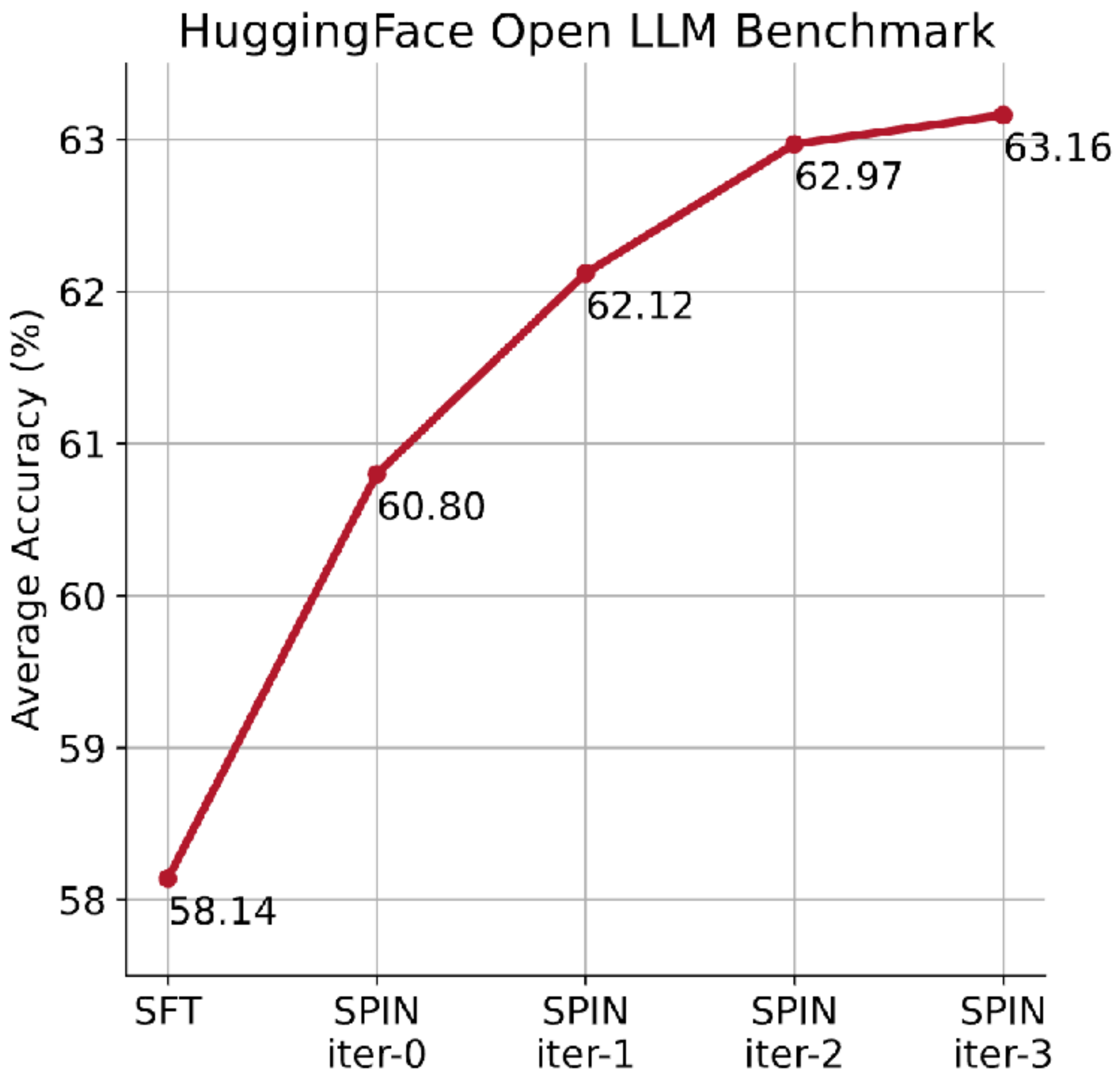
- Typically after SFT for better alignment.
- A preference dataset annotated either by humans (RLHF) or advanced LLMs (RLAIF).
- Model is fine-tuned by RL

**Data curations and annotations are expensive!**

# Self-Play Fine-Tuning (SPIN)

Iterative self-play on an SFT dataset.

- LLM generates its own training data for its upcoming iterations.
- LLM refines itself to discern these self-generated responses from those obtained from human-annotated data.



# Fine-Tuning Pre-Trained LLMs for Specific Tasks

## Supervised Fine-Tuning (SFT)

- High-quality demonstration dataset curated from humans responses or advanced LLM generations.
- Model is typically fine-tuned with supervised learning objective.

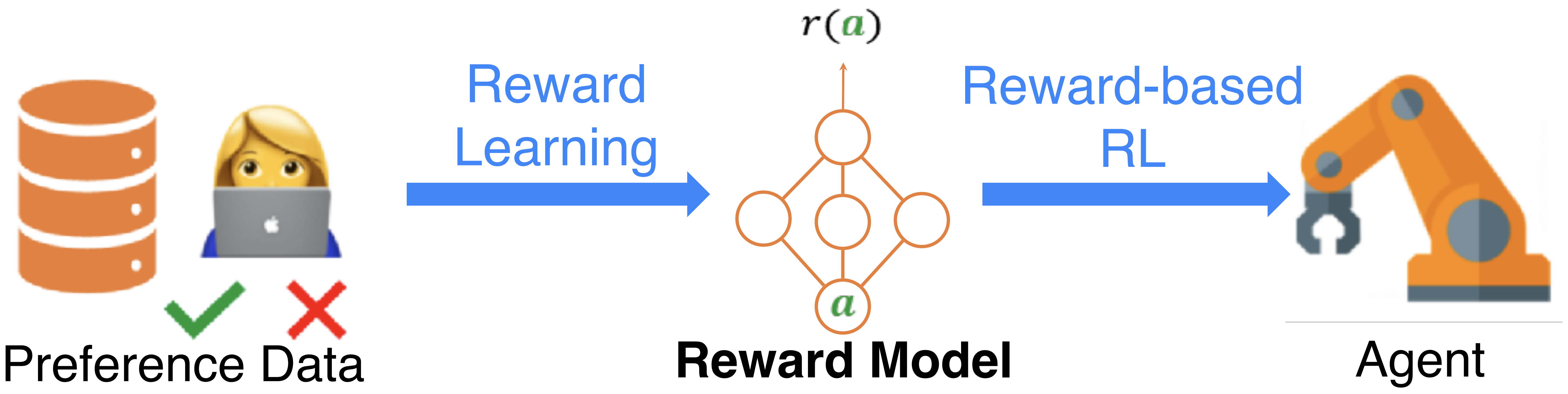
## RL from Human Feedback (RLHF)

- Typically after SFT for better alignment.
- A preference dataset annotated either by humans (RLHF) or advanced LLMs (RLAIF).
- Model is fine-tuned by RL

**Data curations and annotations are expensive!**

# Reinforcement Learning from Human Feedback (RLHF)

RLHF assumes a “**reward**” for every response.



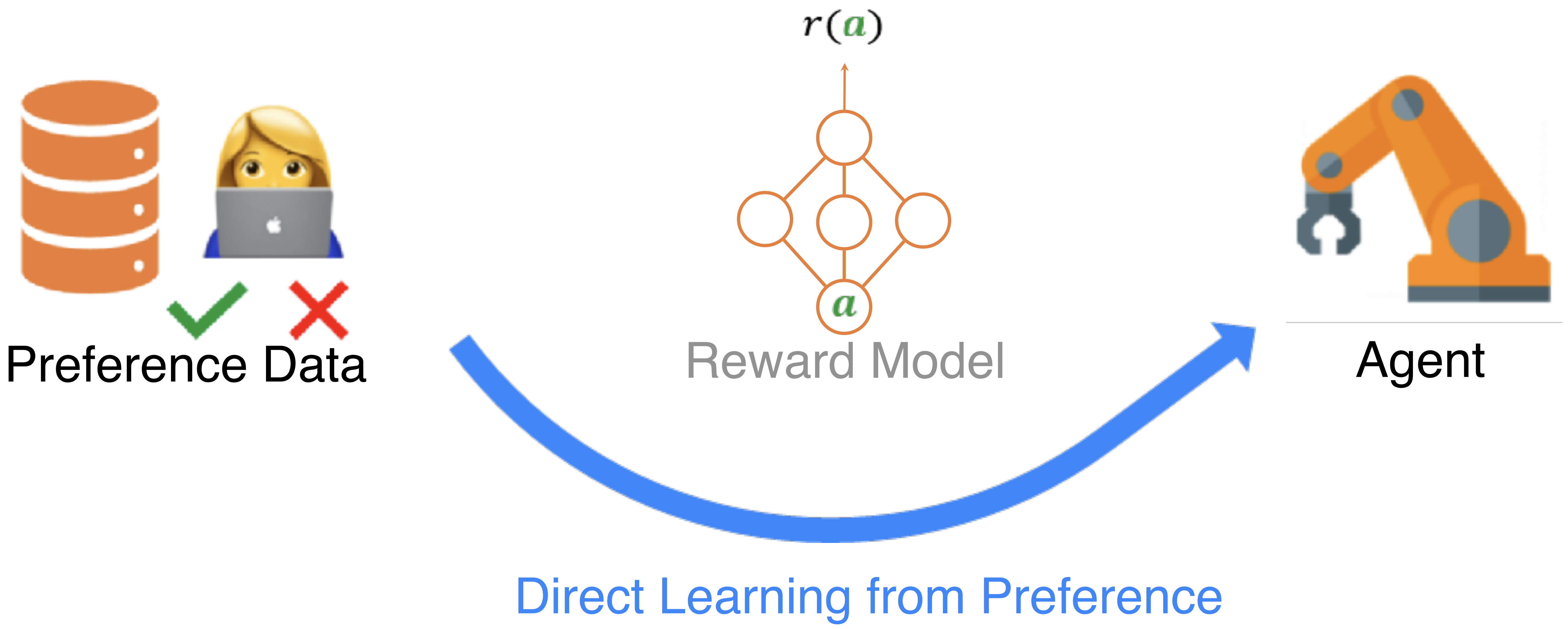

Bradley-Terry model

$$\mathbb{P}(a \succ b) = \frac{e^{r(a)}}{e^{r(a)} + e^{r(b)}}$$

Christiano et al., Deep Reinforcement Learning from Human Preferences, NIPS, 2017
Ziegler et al., Fine-tuning language models from human preferences, 2020

# Direct Preference Optimization (DPO)

DPO does not maintain a separate reward model, instead directly tunes LLM from the preference feedback.
However, it is still based on the Bradley-Terry model implicitly

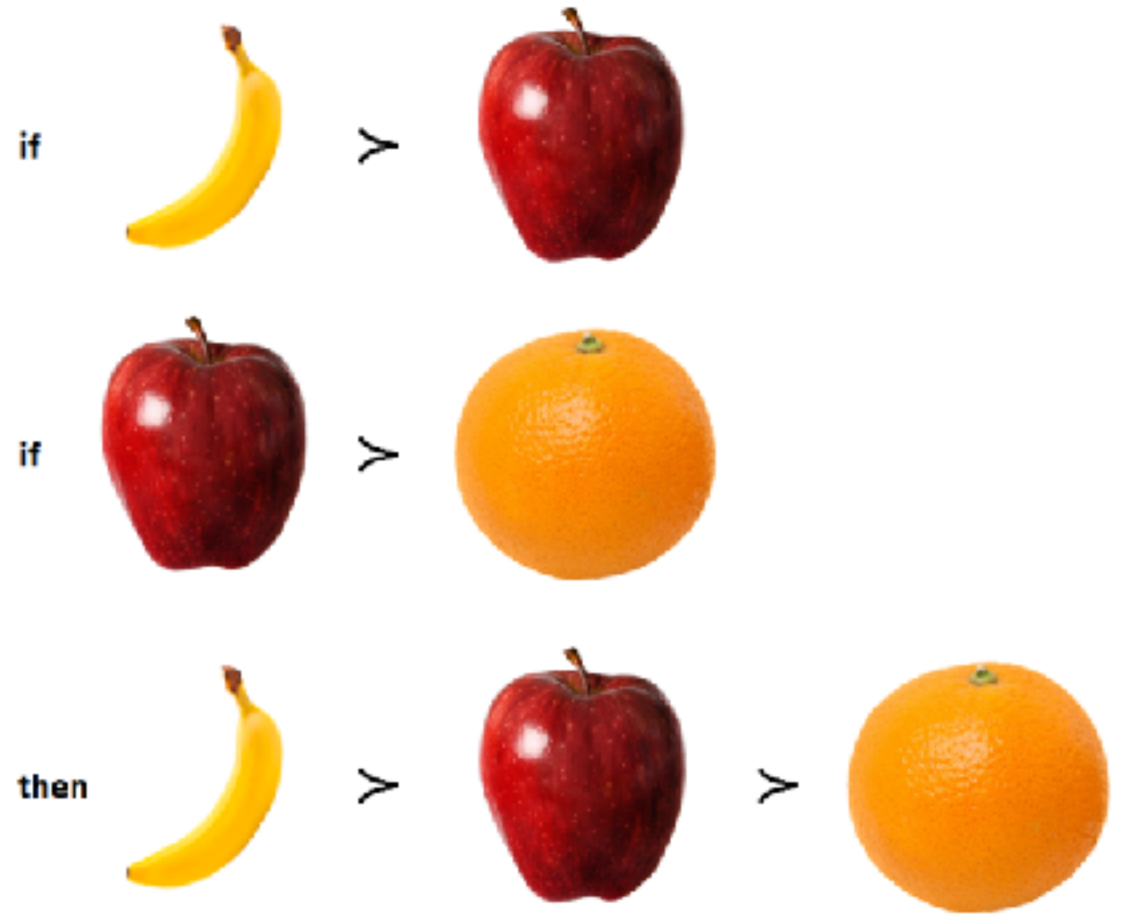



**Do human always exhibit a reward-guided preference?**

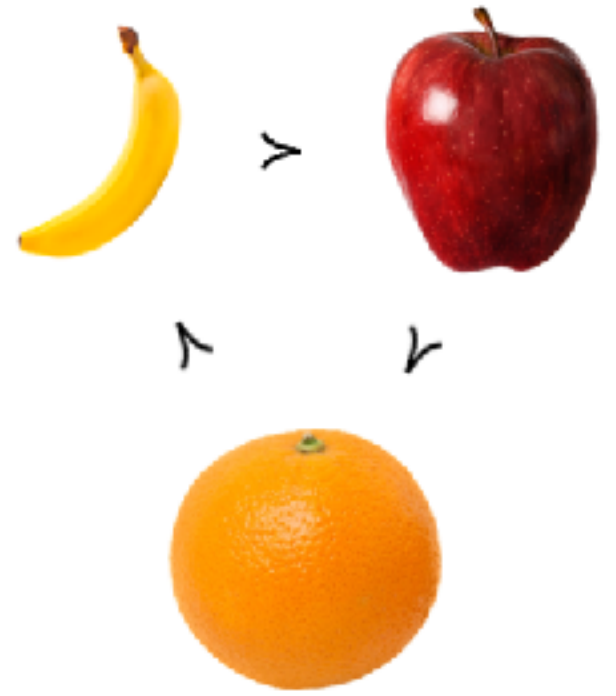
Rafailov et al., Direct Preference Optimization: Your Language Model is Secretly a Reward Model, NuerIPS 2023

# Human Preference is More Complicated than a Reward Model

Human preference is not fully rational or transitive (Tversky, 1969)
There can be loops in the preferences.




**Ideal**: the preference is ordered and internally consistent.



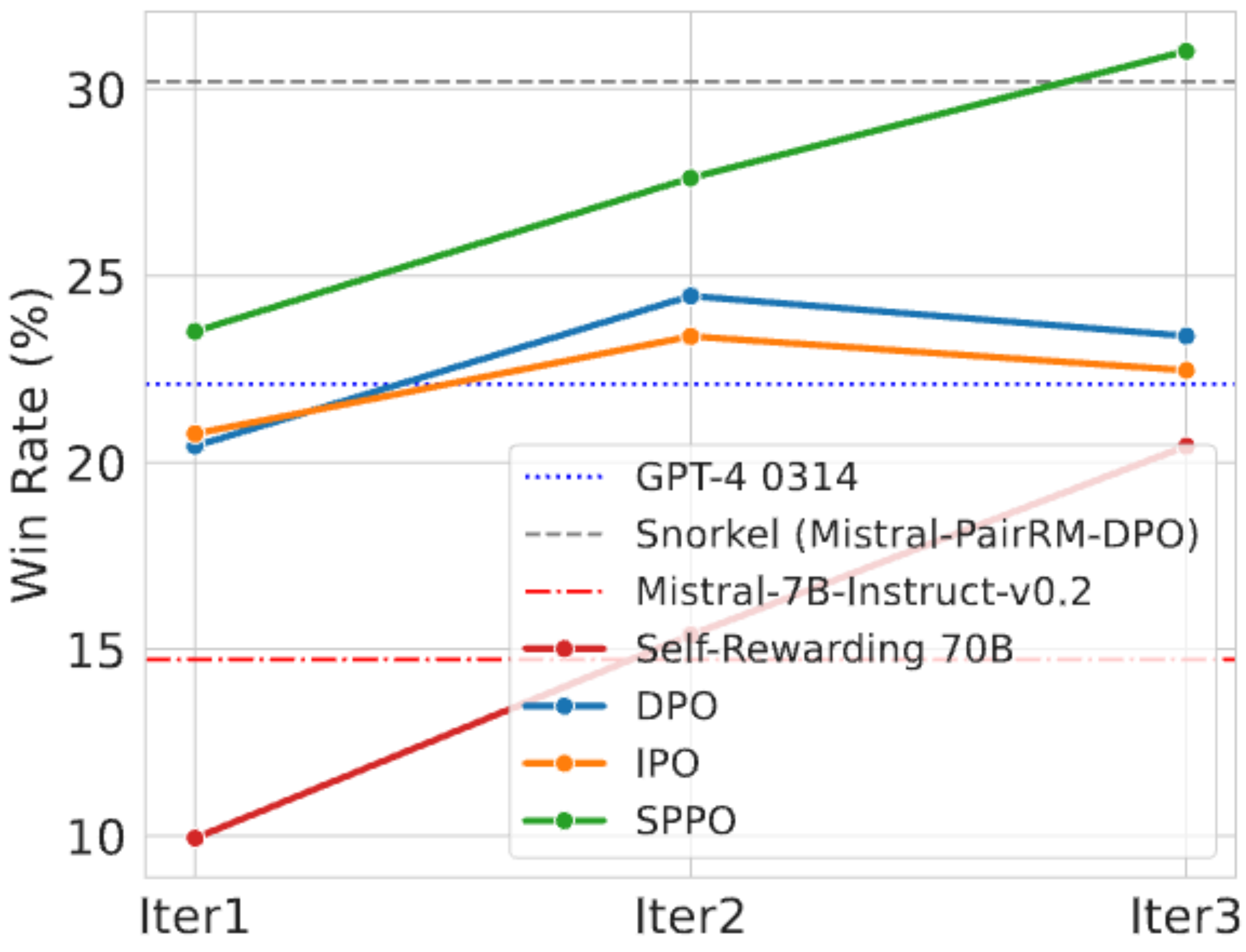
**Reality:**
No consistent ordering.

**Can we deal with general preference for better language model alignment?**

# Self-Play Preference Optimization (SPPO)

Iterative self-play with a preference oracle.

- LLM generates its own training data for its upcoming iterations.
- LLM refines itself to
  (1) output the preferred self-generated responses more often.
  (2) output the rejected self-generated responses less often.




AlpacaEval 2.0 win rate (against GPT4-Turbo)

Wu et al., Self-Play Preference Optimization for Language Model Alignment, 2024

# Self-Play Mechanism

Main player: (LLM at current iteration) aims to win over the opponent player / previous iteration.

Opponent player: (LLM from previous iteration) generates responses to approximate the policy.

SPPO consists of the following two steps at iteration $t+1$: (1) generate responses for the opponent player, and (2) updating the main player to win over the opponent player.

A picture created by DALL·E 3

# Problem Setting

Input sequence: $\mathbf{x} = [x_1, \ldots, x_n]$

Response sequence: $\mathbf{y} = [y_1, \ldots, y_m]$

Conditional Probability (LLM): $p_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x})$

Autoregressive model: $p_{\boldsymbol{\theta}}(\mathbf{y} \mid \mathbf{x}) = \prod_{j=1}^{m} p_{\boldsymbol{\theta}}(y_j \mid \mathbf{x}, \mathbf{y}_{<j})$

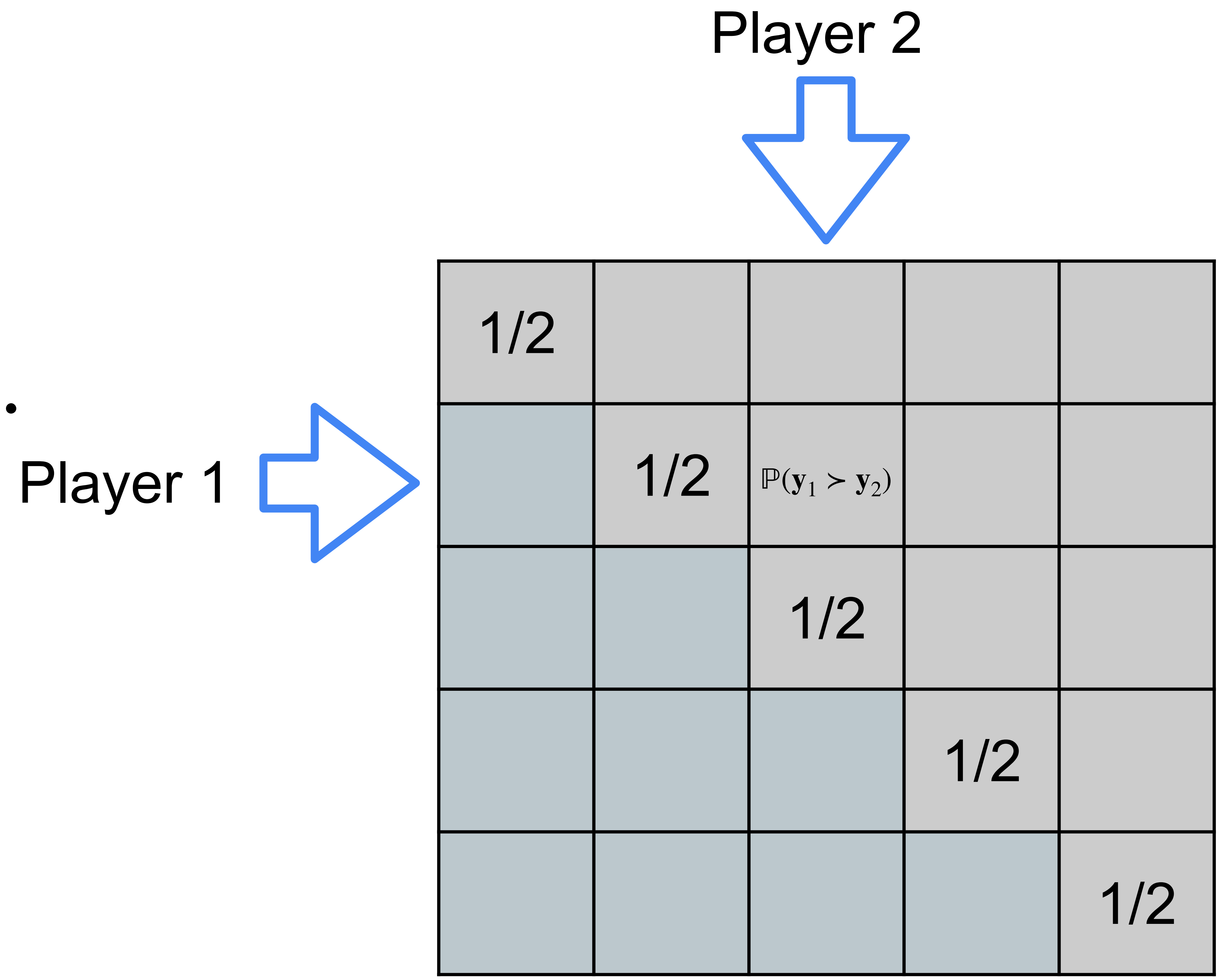
# A Two-Player Constant-Sum Game

For a given prompt $\mathbf{x}$, any two response $\mathbf{y}_1$ and $\mathbf{y}_2$,

One of them is preferred by $\mathbb{P}(\mathbf{y}_1 \succ \mathbf{y}_2 \,|\, \mathbf{x})$

The two-player game is defined as:

$$\max_{\pi} \min_{\pi'} \mathbb{E}_{\mathbf{x}\sim\mathcal{X},\mathbf{y}_1\sim\pi,\mathbf{y}_2\sim\pi'}[\mathbb{P}(\mathbf{y}_1 \succ \mathbf{y}_2 \,|\, \mathbf{x})]\,.$$




Dud´ık et al., Contextual Dueling Bandits, COLT 2015
Munos et al., Nash Learning from Human Feedback, ICML 2024

# Von Neumann Winner (Nash Equilibrium)

The two-player game:

$$\max_{\pi} \min_{\pi'} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}, \mathbf{y}_1 \sim \pi, \mathbf{y}_2 \sim \pi'} [\mathbb{P}(\mathbf{y}_1 \succ \mathbf{y}_2 \,|\, \mathbf{x})] \,.$$

The Von-Neumann winner is the Nash Equilibrium of this symmetric game:

$$(\pi^*, \pi^*) = \arg\max_{\pi} \min_{\pi'} \mathbb{E}_{\mathbf{x} \sim \mathcal{X}, \mathbf{y}_1 \sim \pi, \mathbf{y}_2 \sim \pi'} [\mathbb{P}(\mathbf{y}_1 \succ \mathbf{y}_2 \,|\, \mathbf{x})] \,.$$

**How to solve this game efficiently for LLMs?**


Dud´ık et al., Contextual Dueling Bandits, COLT 2015

Munos et al., Nash Learning from Human Feedback, ICML 2024

# The Multiplicative Weights Algorithm

Freund & Schapire (1999) proposed an algorithm for two-player game.

It admits the multiplicative weights update rule:

$$\pi_{t+1}(\mathbf{y} \mid \mathbf{x}) \propto \pi_t(\mathbf{y} \mid \mathbf{x}) \exp(\eta \mathbb{P}(\mathbf{y} \succ \pi_t \mid \mathbf{x}))$$

Responses with higher win rate will be assigned higher probability.

Equivalently, it admits the following form:

$$\pi_{t+1}(\mathbf{y} \mid \mathbf{x}) = \frac{\pi_t(\mathbf{y} \mid \mathbf{x}) \exp(\eta \mathbb{P}(\mathbf{y} \succ \pi_t \mid \mathbf{x}))}{Z_{\pi_t}(\mathbf{x})}$$

Computationally intractable; how to approximate it efficiently?

Freund & Schapire, Adaptive game playing using multiplicative weights, Games and Economic Behavior, 1999

# The SPPO objective

The solution has the form: $\log\left(\frac{\pi_{t+1}(\mathbf{y}\,|\,\mathbf{x})}{\pi_t(\mathbf{y}\,|\,\mathbf{x})}\right) = \eta\mathbb{P}(\mathbf{y} \succ \pi_t\,|\,\mathbf{x}) - \log Z_{\pi_t}(\mathbf{x})$

Using least-square regression to obtain the optimization objective:

$$\pi_{t+1} = \arg\min_{\pi} \mathbb{E}_{\mathbf{x}\sim\mathcal{X},\mathbf{y}\sim\pi_t(\cdot|\mathbf{x})}\left[\log\left(\frac{\pi(\mathbf{y}\,|\,\mathbf{x})}{\pi_t(\mathbf{y}\,|\,\mathbf{x})}\right) - \left(\eta\mathbb{P}(\mathbf{y} \succ \pi_t\,|\,\mathbf{x}) - \log \boxed{Z_{\pi_t}(\mathbf{x})\Big)\Big]^2}\right.$$

Replace $\log Z_{\pi_t}(\mathbf{x})$ with $\eta/2$

$$\pi_{t+1} = \arg\min_{\pi} \mathbb{E}_{\mathbf{x}\sim\mathcal{X},\mathbf{y}\sim\pi_t(\cdot|\mathbf{x})}\left[\log\left(\frac{\pi(\mathbf{y}\,|\,\mathbf{x})}{\pi_t(\mathbf{y}\,|\,\mathbf{x})}\right) - \eta\left(\mathbb{P}(\mathbf{y} \succ \pi_t\,|\,\mathbf{x}) - 1/2\right)\right]^2$$

# An End-to-End SPPO objective

$$\pi_{t+1} = \arg\min_{\pi} \mathbb{E}_{\mathbf{x}\sim\mathcal{X},\mathbf{y}\sim\pi_t(\cdot|\mathbf{x})}\left[\log\left(\frac{\pi(\mathbf{y}\,|\,\mathbf{x})}{\pi_t(\mathbf{y}\,|\,\mathbf{x})}\right) - \eta\Big(\mathbb{P}(\mathbf{y} \succ \pi_t\,|\,\mathbf{x}) - 1/2\Big)\right]^2$$

In practice, $\mathbb{P}(\mathbf{y} \succ \pi_t\,|\,\mathbf{x})$ is replaced with finite sample estimate.

Intuitively, if a tie occurs, i.e., $\mathbb{P}(\mathbf{y} \succ \pi_t\,|\,\mathbf{x}) = 1/2$, we do not update weight at $\mathbf{y}$.

If $\mathbf{y}$ wins over $\pi_t$ on average, i.e., $\mathbb{P}(\mathbf{y} \succ \pi_t\,|\,\mathbf{x}) > 1/2$, we increase the probability at $\mathbf{y}$ to take the advantage of $\mathbf{y}$ over $\pi_t$.

# The SPPO Algorithm

---

**Algorithm 1** `Self-Play Preference Optimization(SPPO)`

---

**input**: base policy $\pi_{\boldsymbol{\theta}_1}$, preference oracle $\mathbb{P}$, learning rate $\eta$, number of generated samples $K$.

**for** $t = 1, 2, \ldots$ **do**

Generate synthetic responses by sampling $\mathbf{x} \sim \mathcal{X}$ and $\mathbf{y}_{1:K} \sim \pi_t(\cdot|\mathbf{x})$.

Annotate the win-rate $\mathbb{P}(\mathbf{y}_k \succ \mathbf{y}_{k'}|\mathbf{x}), \forall k, k' \in [K]$.

Select responses from $\mathbf{y}_{1:K}$ to form dataset $\mathcal{D}_t = \{(\mathbf{x}_i, \mathbf{y}_i, \widehat{P}(\mathbf{y}_i \succ \pi_t|\mathbf{x}_i))\}_{i\in[N]}$.

Optimize $\pi_{\boldsymbol{\theta}_{t+1}}$ according to (4.6):

$$\boldsymbol{\theta}_{t+1} \leftarrow \operatorname*{argmin}_{\boldsymbol{\theta}} \mathbb{E}_{(\mathbf{x},\mathbf{y},\widehat{P}(\mathbf{y}\succ\pi_t|\mathbf{x}))\sim\mathcal{D}_t} \left( \log\left(\frac{\pi_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x})}{\pi_t(\mathbf{y}|\mathbf{x})}\right) - \eta\left(\widehat{P}(\mathbf{y} \succ \pi_t|\mathbf{x}) - \frac{1}{2}\right)\right)^2. \tag{4.7}$$

**end for**

---

# SPPO v.s. DPO and IPO

Consider a winner $\mathbf{y}_w$ and $\mathbf{y}_l$ with deterministic preference $\mathbb{P}(\mathbf{y}_w \succ \mathbf{y}_l) = 1$.

Denote $a = \beta \log\left(\frac{\pi_\theta(\mathbf{y}_w \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_w \mid \mathbf{x})}\right)$, $b = \beta \log\left(\frac{\pi_\theta(\mathbf{y}_l \mid \mathbf{x})}{\pi_{\text{ref}}(\mathbf{y}_l \mid \mathbf{x})}\right)$

$$\ell_{\text{DPO}} = -\log \sigma(a - b) \qquad \sigma(x) = e^x/(1 + e^x)$$

$$\ell_{\text{IPO}} = [(a - b) - 1]^2$$

$$\ell_{\text{SPPO}} = (a-1/2)^2 + (b+1/2)^2$$

DPO and IPO loss enlarge the gap between the log-likelihood ratio

SPPO pushes the winner up and pulls the loser down

Azar et al., A general theoretical paradigm to understand learning from human preferences, 2023

# Theoretical Analysis

We show the algorithm can provably converge to the Nash equilibrium given sufficient samples.

**Theorem 4.1.** Assume the optimization problem (4.4) is realizable. Denote $\pi_t$ as the policy obtained via (4.4) and the mixture policy $\bar{\pi}_T = \frac{1}{T}\sum_{t=1}^{T}\pi_t$. By setting $\eta = \Theta(1/\sqrt{T})$, we have that

$$\max_{\pi}\big[\mathbb{P}(\pi \succ \bar{\pi}_T)\big] - \min_{\pi}\big[\mathbb{P}(\pi \prec \bar{\pi}_T)\big] = O(1/\sqrt{T}).$$

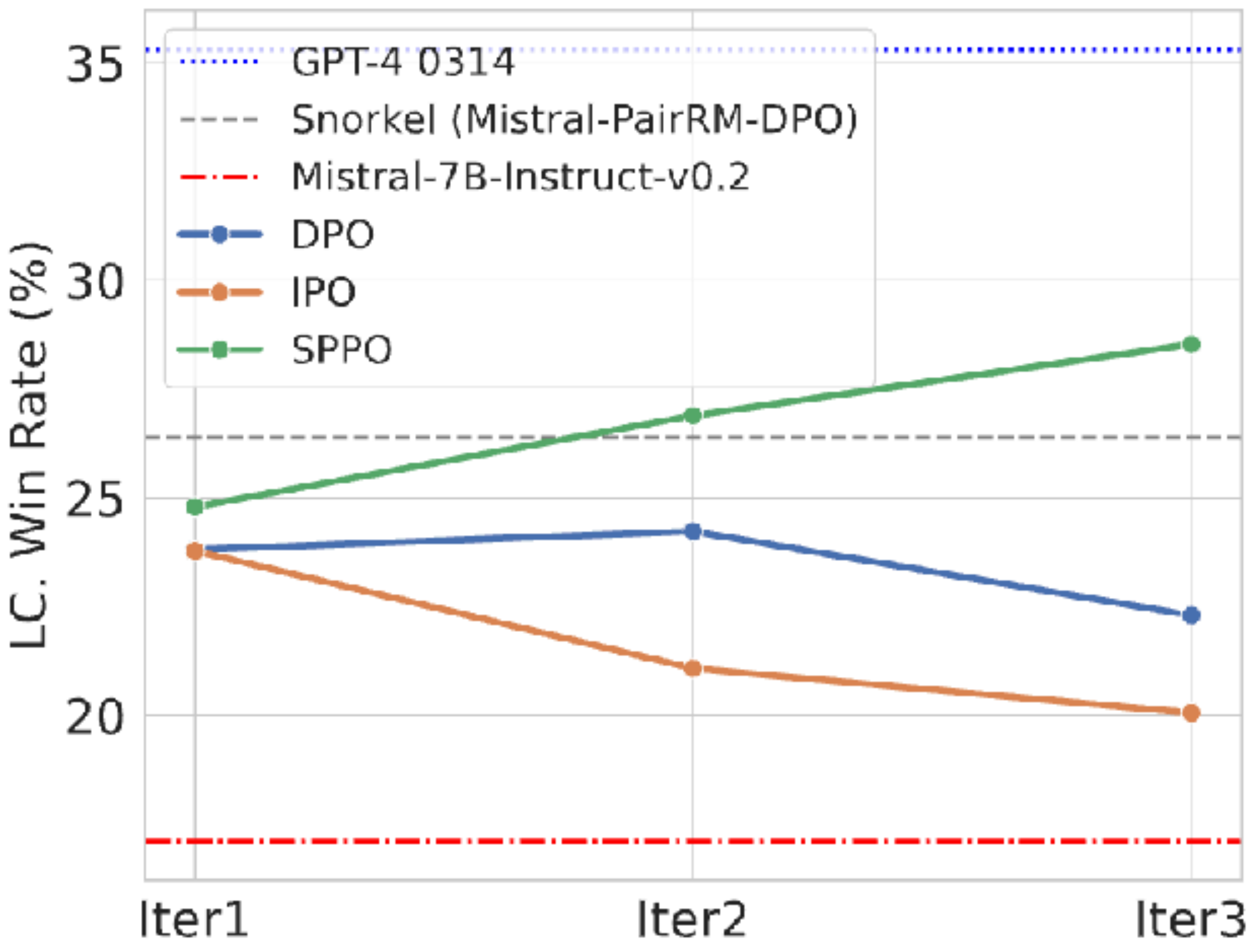
# Experiment Setup

- Model: **Mistral-7B-Instruct-v0.2**, an instruction fine-tuned version of Mistral-7B-v0.2.
- Preference Model: **PairRM**, an efficient pair-wise preference model of size 0.4B.
- Dataset: **UltraFeedback**, ~60k prompts from diverse sources
  - We split the 60k prompts three-fold into 3 epochs of iterative training.
- Evaluation: **AlpacaEval 2.0, MT-Bench,** HuggingFace **Open LLM Leaderboard**.

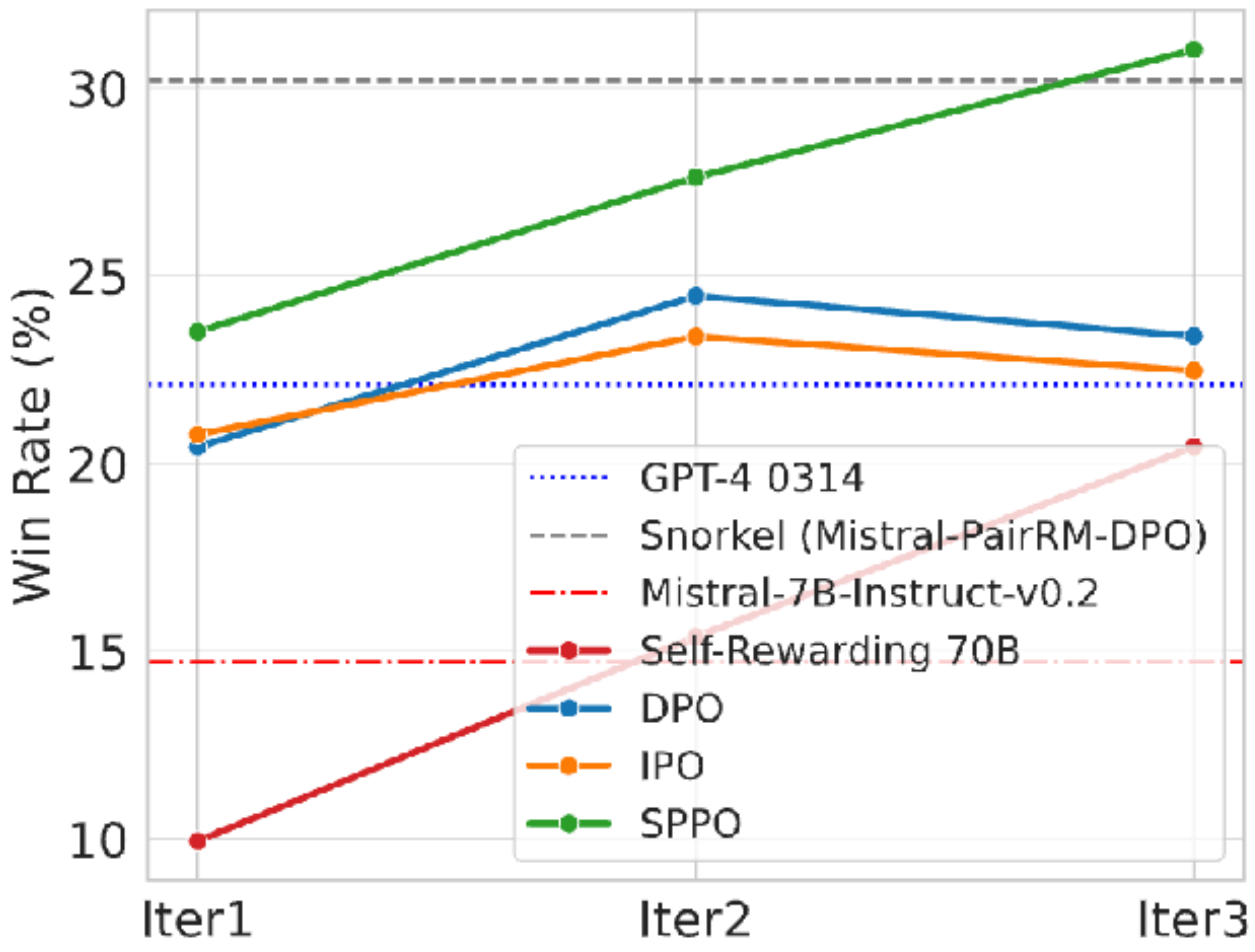
https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2
https://huggingface.co/llm-blender/PairRM

# Performance on AlpacaEval 2.0

LLM gets self-improved by SPPO. In particular, SPPO can surpass models fine-tuned with responses or preferences generated by GPT-4




(a)

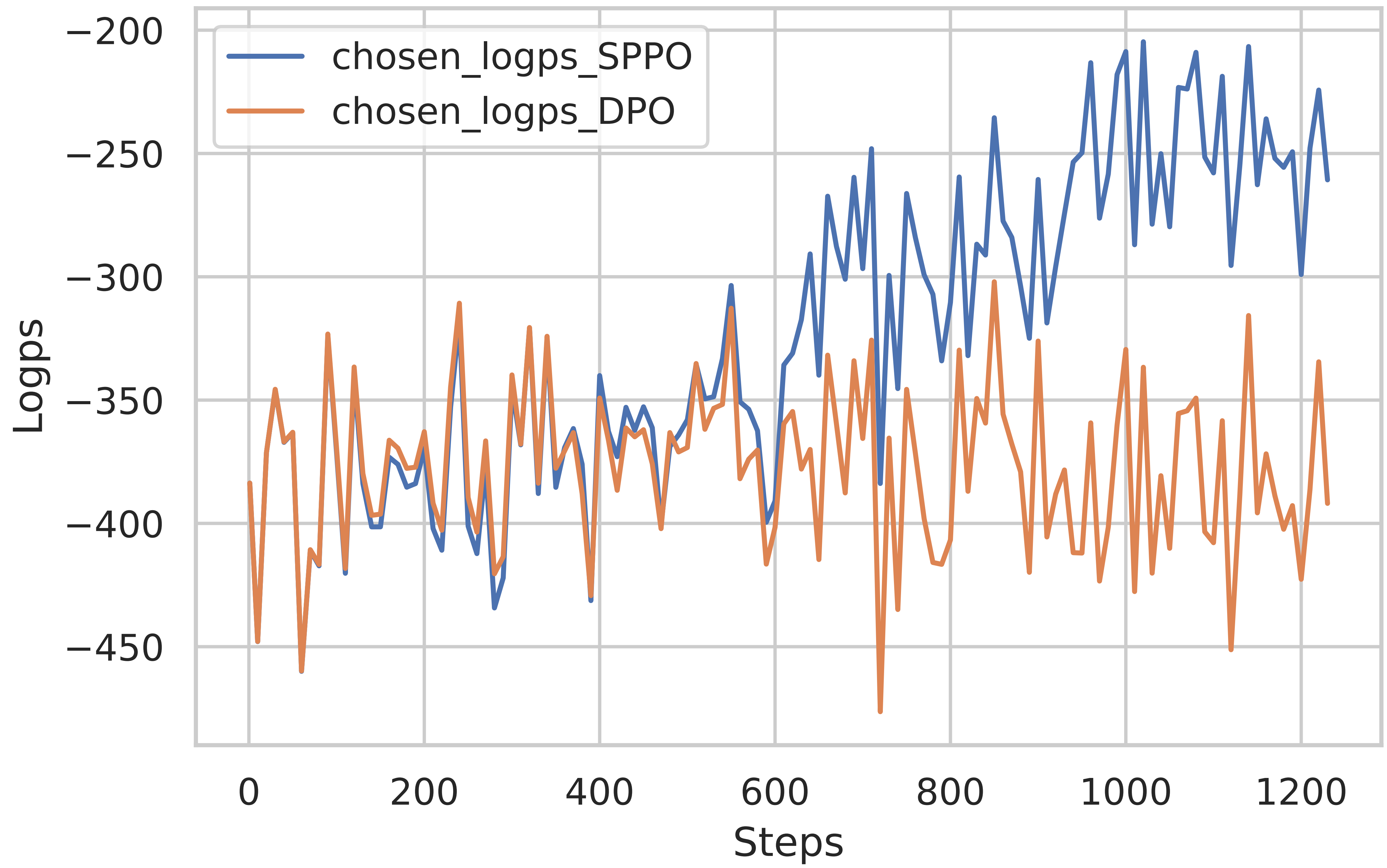


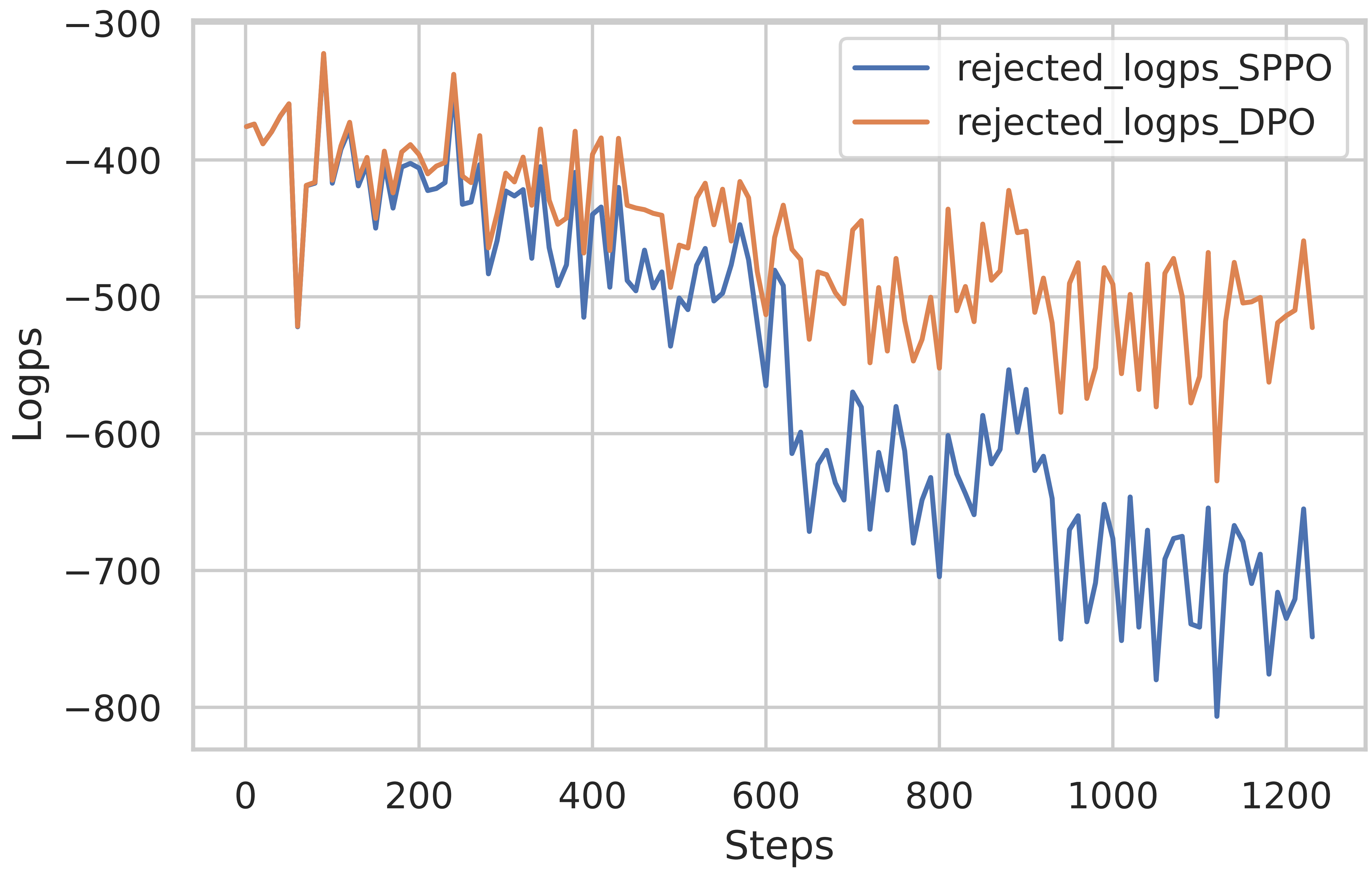

(b)

| Model | AlpacaEval 2.0 LC. Win Rate | AlpacaEval 2.0 Win Rate |
|---|---|---|
| GPT-4 Turbo | 50.0 | 50.0 |
| Claude 3 Opus | 40.5 | 29.1 |
| GPT-4 0314 | 35.3 | 22.1 |
| Llama 3 70B Instruct | 34.4 | 33.2 |
| SPPO Iter3 (best-of-16) | 32.1 | 34.9 |
| GPT-4 0613 | 30.2 | 15.8 |
| Snorkel (best-of-16) | 30.0 | 34.9 |
| Mistral Medium | 28.6 | 21.9 |
| SPPO Iter3 | 28.5 | 31.0 |
| Claude 2 | 28.2 | 17.2 |
| Snorkel | 26.4 | 30.2 |
| Gemini Pro | 24.4 | 18.2 |
| Mistral 8×7B v0.1 | 23.7 | 18.1 |
| Llama 3 8B Instruct | 22.9 | 22.6 |
| GPT-3.5 Turbo 0613 | 22.7 | 14.1 |
| Vicuna 33B v1.3 | 17.6 | 12.7 |

# SPPO can effectively boost up winning probability

Pairwise loss like DPO can only enlarge the relative probability gap between the winner and loser. SPPO can boost up the probability density of the winner.



# Performance on MT-Bench

SPPO exhibits performance gain across different benchmarks.

| Model | MT-Bench 1st Turn | MT-Bench 2nd Turn | MT-Bench Average |
|---|---|---|---|
| Mistral-7B-Instruct-v0.2 | 7.78 | 7.25 | 7.51 |
| Snorkel (Mistral-PairRM-DPO) | 7.83 | 7.33 | 7.58 |
| DPO Iter1 | 7.45 | 6.58 | 7.02 |
| DPO Iter2 | 7.57 | 6.56 | 7.06 |
| DPO Iter3 | 7.49 | 6.69 | 7.09 |
| SPPO Iter1 | 7.63 | 6.79 | 7.21 |
| SPPO Iter2 | 7.90 | 7.08 | 7.49 |
| SPPO Iter3 | 7.84 | 7.34 | **7.59** |


MT-Bench

# Performance on OpenLLM Leaderboard

SPPO exhibits performance gain across different benchmarks.

| Models | Arc | TruthfulQA | WinoGrande | GSM8k | HellaSwag | MMLU | Average |
|---|---|---|---|---|---|---|---|
| Mistral-7B-Instruct-v0.2 | 63.65 | 66.85 | 77.98 | 41.93 | 84.89 | 59.15 | 65.74 |
| Snorkel | 66.04 | 70.86 | 77.74 | 36.77 | 85.64 | 60.83 | 66.31 |
| DPO Iter1 | 63.14 | 68.39 | 77.19 | 40.33 | 85.25 | 59.41 | 65.62 |
| DPO Iter2 | 64.16 | 67.84 | 76.09 | 39.95 | 85.23 | 59.03 | 65.38 |
| DPO Iter3 | 65.19 | 67.89 | 77.27 | 32.30 | 85.49 | 59.00 | 64.52 |
| IPO Iter1 | 64.68 | 68.60 | 77.98 | 43.75 | 85.08 | 59.04 | 66.52 |
| IPO Iter2 | 62.12 | 66.30 | 77.51 | 39.20 | 83.15 | 59.70 | 64.66 |
| IPO Iter3 | 62.97 | 67.12 | 77.51 | 37.45 | 83.69 | 59.57 | 64.72 |
| SPPO Iter1 | 65.02 | 69.40 | 77.82 | 43.82 | 85.11 | 58.84 | 66.67 |
| SPPO Iter2 | 65.53 | 69.55 | 77.03 | 44.35 | 85.29 | 58.72 | **66.75** |
| SPPO Iter3 | 65.36 | 69.97 | 76.80 | 42.68 | 85.16 | 58.45 | 66.40 |

Open LLM Leaderboard

# Takeaway

- SPPO is a self-play framework that can efficiently align LLM with general human preference.
- SPPO admits a simple end-to-end objective function for preference optimization that can effectively boost up the probability of the chosen responses.
- SPPO has achieved remarkable performance improvement across various benchmarks, **without any strong external supervision** like GPT-4.

# Paper, Code and Models

## Self-Play Preference Optimization for Language Model Alignment

Yue Wu[*†] Zhiqing Sun[*‡] Huizhuo Yuan[*§] Kaixuan Ji[¶] Yiming Yang[‖] Quanquan Gu[**]

- **Paper:** https://arxiv.org/abs/2405.00675
- **Code:** https://github.com/uclaml/SPPO
- **Models:** https://huggingface.co/collections/UCLA-AGI/sppo-6635fdd844f2b2e4a94d0b9a

# Thank you for Listening!