SimPO: Simple Preference Optimization with a Reference-Free Reward

Yu Meng[1*] **Mengzhou Xia**[2*] **Danqi Chen**[2]
[1]Computer Science Department, University of Virginia
[2]Princeton Language and Intelligence (PLI), Princeton University
yumeng5@virginia.edu
{mengzhou,danqic}@cs.princeton.edu

Abstract

Direct Preference Optimization (DPO) is a widely used offline preference optimization algorithm that reparameterizes reward functions in reinforcement learning from human feedback (RLHF) to enhance simplicity and training stability. In this work, we propose SimPO, a simpler yet more effective approach. The effectiveness of SimPO is attributed to a key design: using the *average* log probability of a sequence as the implicit reward. This reward formulation better aligns with model generation and eliminates the need for a reference model, making it more compute and memory efficient. Additionally, we introduce a target reward margin to the Bradley-Terry objective to encourage a larger margin between the winning and losing responses, further enhancing the algorithm's performance. We compare SimPO to DPO and its latest variants across various state-of-the-art training setups, including both base and instruction-tuned models like Mistral and Llama3. We evaluate on extensive instruction-following benchmarks, including AlpacaEval 2, MT-Bench, and the recent challenging Arena-Hard benchmark. Our results demonstrate that SimPO consistently and significantly outperforms existing approaches without substantially increasing response length. Specifically, SimPO outperforms DPO by up to 6.4 points on AlpacaEval 2 and by up to 7.5 points on Arena-Hard. Our top-performing model, built on Llama3-8B-Instruct, achieves a remarkable 53.7 length-controlled win rate on AlpacaEval 2—surpassing Claude 3 Opus on the leaderboard, and a 36.5 win rate on Arena-Hard—making it the strongest 8B open-source model.[1]

Presented by Anton Lavrouk and Loránd Cheng

From Last Time…

Objective: Maximize reward while minimizing the KL divergence from the supervised baseline model

$$R(x, y) = \overbrace{r_\theta(x, y)}^{\text{Normal RL Objective (Reward from Reward Model)}} - \beta \overbrace{\log[\pi_\phi^{\mathrm{RL}}(y|x)/\pi^{\mathrm{SFT}}(y|x)]}^{\text{KL Divergence from Supervised Baseline}}$$

$$R(x,y) = \overbrace{r_\theta(x,y)}^{\text{Normal RL Objective (Reward from Reward Model)}} - \beta \overbrace{\log[\pi_\phi^{\mathrm{RL}}(y|x)/\pi^{\mathrm{SFT}}(y|x)]}^{\text{KL Divergence from Supervised Baseline}}$$

$$r(x,y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\mathrm{ref}}(y \mid x)} + \beta \log Z(x),$$

A closed form optimization solution for the reward model

$$r(x, y) = \beta \log \frac{\pi_\theta(y \mid x)}{\pi_{\text{ref}}(y \mid x)} + \beta \log Z(x),$$

A closed form optimization solution for the reward model

Plug into Bradley-Terry

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

An offline objective function for preference data - DPO

Two big problems….

Two big problems….

Using the reference model incurs additional memory and computation costs, and in general feels weird in an offline setting.

Two big problems….

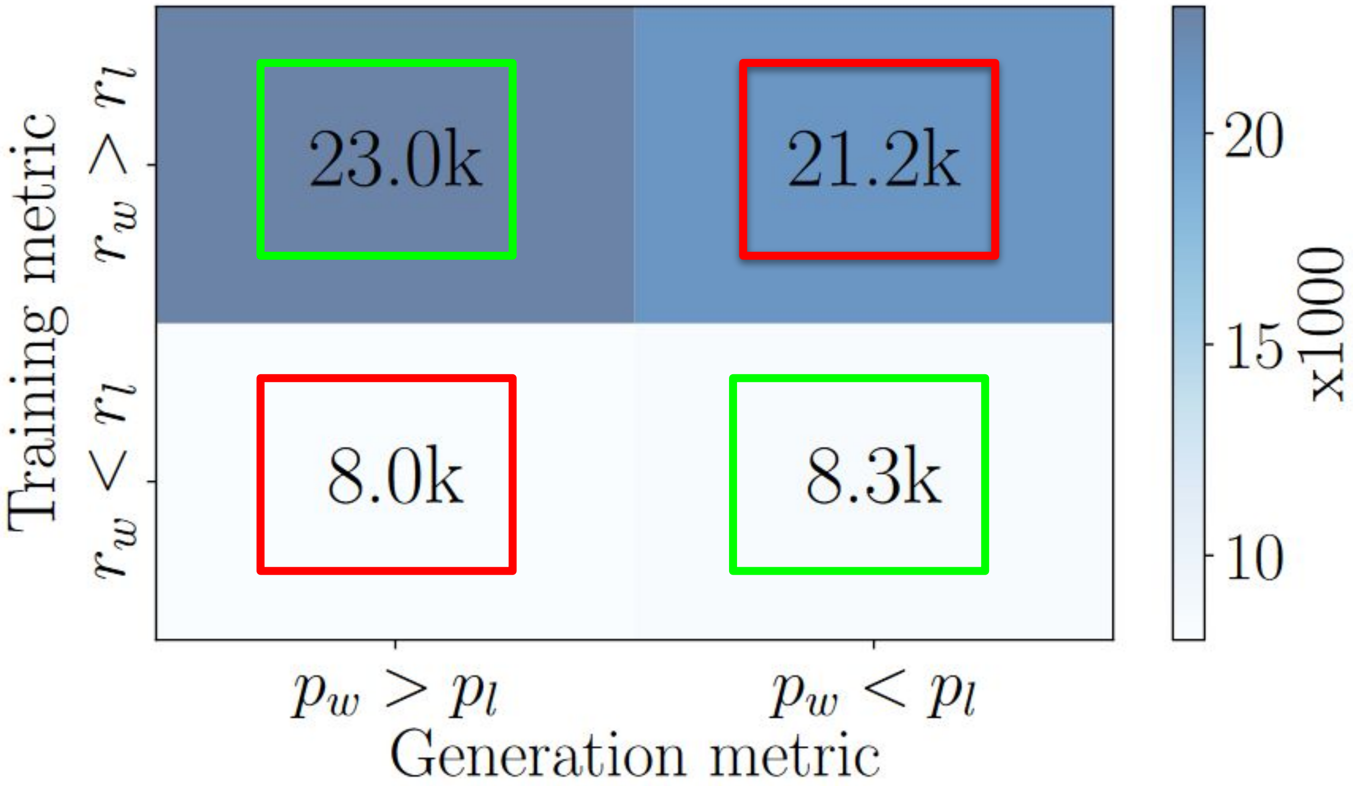
Using the reference model incurs additional memory and computation costs, and in general feels weird in an offline setting.

There is a discrepancy between this reward and what is actually used during text generation.

There is a discrepancy between this reward and what is actually used during text generation

$$p_\theta(y \mid x) = \frac{1}{|y|} \log \pi_\theta(y \mid x) = \frac{1}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i}).$$

Computationally intractable during inference: cannot test every single possibility



23.0k
21.2k
8.0k
8.3k
Training metric
$r_w > r_l$
$r_w < r_l$
$p_w > p_l$
$p_w < p_l$
Generation metric
20
15
10
x1000

What if we replace the reward function with the generation metric?

$$p_\theta(y \mid x) = \frac{1}{|y|} \log \pi_\theta(y \mid x) = \frac{1}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i}).$$

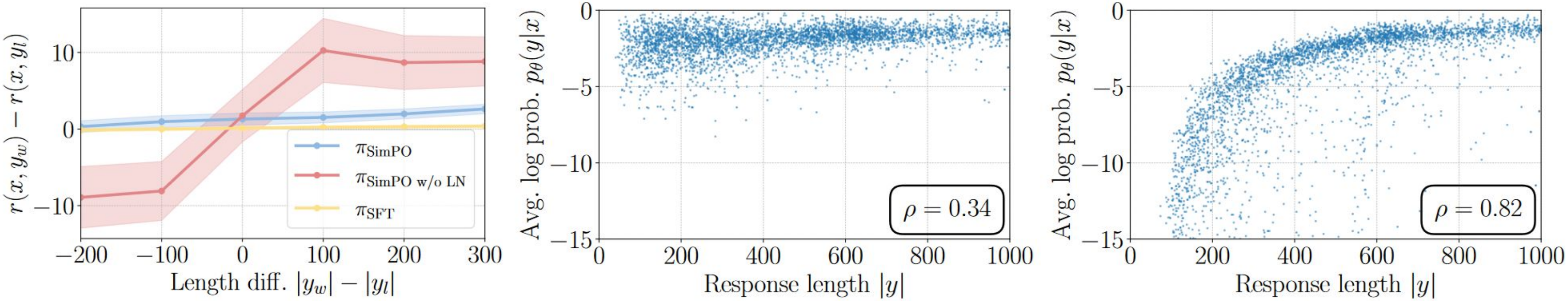
What if we replace the reward function with the generation metric?

$$p_\theta(y \mid x) = \frac{1}{|y|} \log \pi_\theta(y \mid x) = \frac{1}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i}).$$

$$r_{\text{SimPO}}(x, y) = \frac{\boxed{\beta}}{|y|} \log \pi_\theta(y \mid x) = \frac{\boxed{\beta}}{|y|} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i}),$$

Aside: length normalization

$$r_{\text{SimPO}}(x, y) = \frac{\beta}{\boxed{|y|}} \log \pi_\theta(y \mid x) = \frac{\beta}{\boxed{|y|}} \sum_{i=1}^{|y|} \log \pi_\theta(y_i \mid x, y_{<i}),$$



(a) Reward optimization. (b) SimPO. (c) SimPO without LN.

Plugging this into the Bradley-Terry model, and then maximizing the log probability of the winning generation over the losing one yields the SimPO objective function (almost)

$$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\frac{\beta}{|y_w|}\log\pi_\theta(y_w|x) - \frac{\beta}{|y_l|}\log\pi_\theta(y_l|x)\right)\right]$$

Plugging this into the Bradley-Terry model, and then maximizing the log probability of the winning generation over the losing one yields the SimPO objective function (almost)

$$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x, y_w, y_l)\sim\mathcal{D}}\left[\log\sigma\left(\frac{\beta}{|y_w|}\log\pi_\theta(y_w|x) - \frac{\beta}{|y_l|}\log\pi_\theta(y_l|x)\right)\right]$$

No reference model!

Plugging this into the Bradley-Terry model, and then maximizing the log probability of the winning generation over the losing one yields the SimPO objective function (almost)

$$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x, y_w, y_l) \sim \mathcal{D}} \left[\log \sigma \left(\frac{\beta}{|y_w|} \log \pi_\theta(y_w|x) - \frac{\beta}{|y_l|} \log \pi_\theta(y_l|x) \right) \right]$$

No reference model!

Aligned with language modeling objective!

One more thing…

One more thing…

$$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\frac{\beta}{|y_w|}\log\pi_\theta(y_w|x) - \frac{\beta}{|y_l|}\log\pi_\theta(y_l|x) - \boxed{\gamma}\right)\right]$$

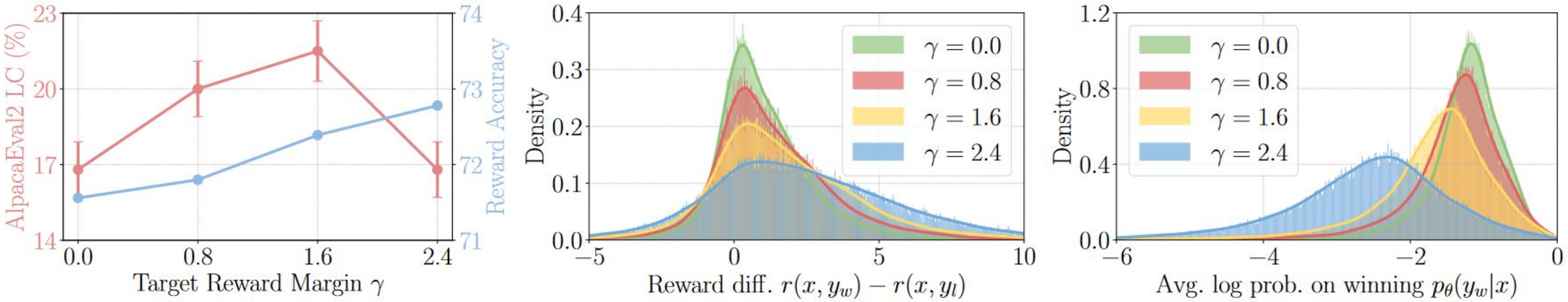
One more thing…

$$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\frac{\beta}{|y_w|}\log\pi_\theta(y_w|x) - \frac{\beta}{|y_l|}\log\pi_\theta(y_l|x) - \boxed{\gamma}\right)\right]$$

Ensures that the winning generation beats the losing generation by **at least γ**

One more thing…

$$\mathcal{L}_{\text{SimPO}}(\pi_\theta) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\left[\log\sigma\left(\frac{\beta}{|y_w|}\log\pi_\theta(y_w|x) - \frac{\beta}{|y_l|}\log\pi_\theta(y_l|x) - \boxed{\gamma}\right)\right]$$



(a) Performance w/ different γ. (b) Reward diff. distribution. (c) Log prob. distribution.

$$\mathcal{L}_{\mathbf{DPO}}(\pi_\theta; \pi_{\text{ref}}) =$$
$$-\mathbb{E}\left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right]$$

$$\mathcal{L}_{\mathbf{SimPO}}(\pi_\theta) =$$
$$-\mathbb{E}\left[\log \sigma\left(\frac{\beta}{|y_w|} \log \pi_\theta(y_w \mid x) - \frac{\beta}{|y_l|} \log \pi_\theta(y_l \mid x) - \gamma\right)\right]$$

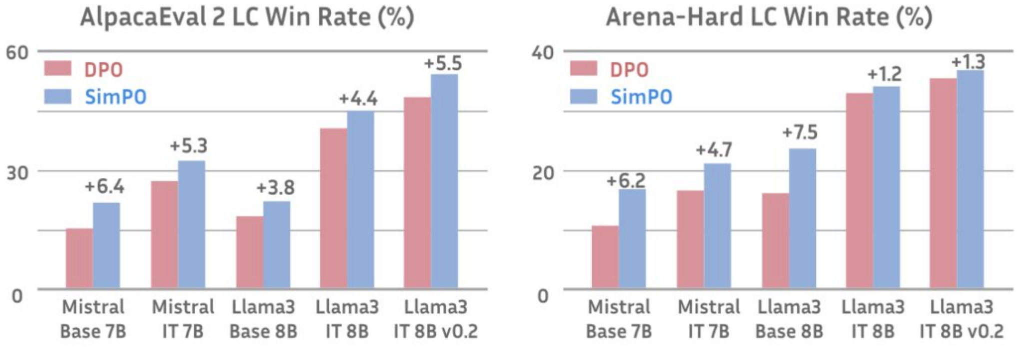


Figure 1: SimPO and DPO mainly differ in their reward formulation, as indicated in the shaded box. SimPO outperforms DPO across a wide range of settings on AlpacaEval 2 and Arena-Hard.

Experiments and Results

Base Models

Four models were used as the base LM for preference optimization:

Base versions:

- Mistral-7B-v0.1
- Llama-3-8B

Instruction-tuned versions:

- Mistral-7B-Instruct-v0.2
- Llama-3-8B-Instruct

Training Data

Base versions:

- First trained on UltraChat-200k to obtain SFT model
- Then trained on UltraFeedback preference data

Instruction-tuned versions:

- Take instruction-tuned model as SFT model
- Only trained on preference data
    - Re-generate UltraFeedback preference dataset responses using SFT model
    - Use a reward model (PairRM, ArmoRM) to obtain synthetic preferences

Benchmarks

Use 3 main benchmarks for evaluation:

1. **AlpacaEval 2:** 805 Qs from 5 datasets
2. **Arena-Hard:** 500 technical problem solving Qs
3. **MT-Bench:** 80 Qs from 8 categories

All use GPT-4 (+variants) for automatic evaluation

	# Exs.	Baseline Model	Judge Model	Scoring Type	Metric
AlpacaEval 2	805	GPT-4 Turbo	GPT-4 Turbo	Pairwise comparison	LC & raw win rate
Arena-Hard	500	GPT-4-0314	GPT-4 Turbo	Pairwise comparison	Win rate
MT-Bench	80	-	GPT-4/GPT-4 Turbo	Single-answer grading	Rating of 1-10

Baselines

Method	Objective
RRHF [87]	$\max\left(0, -\frac{1}{\lvert y_w \rvert}\log \pi_\theta(y_w \lvert x) + \frac{1}{\lvert y_l \rvert}\log \pi_\theta(y_l \lvert x)\right) - \lambda \log \pi_\theta(y_w \lvert x)$
SLiC-HF [92]	$\max\left(0, \delta - \log \pi_\theta(y_w \lvert x) + \log \pi_\theta(y_l \lvert x)\right) - \lambda \log \pi_\theta(y_w \lvert x)$
DPO [64]	$-\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - \beta \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)}\right)$
IPO [6]	$\left(\log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)} - \frac{1}{2\tau}\right)^2$
CPO [84]	$-\log \sigma\left(\beta \log \pi_\theta(y_w \lvert x) - \beta \log \pi_\theta(y_l \lvert x)\right) - \lambda \log \pi_\theta(y_w \lvert x)$
KTO [27]	$-\lambda_w \sigma\left(\beta \log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - z_{\text{ref}}\right) + \lambda_l \sigma\left(z_{\text{ref}} - \beta \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)}\right)$, where $z_{\text{ref}} = \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\beta \text{KL}\left(\pi_\theta(y \lvert x) \Vert \pi_{\text{ref}}(y \lvert x)\right)\right]$
ORPO [40]	$-\log p_\theta(y_w \lvert x) - \lambda \log \sigma\left(\log \frac{p_\theta(y_w \lvert x)}{1-p_\theta(y_w \lvert x)} - \log \frac{p_\theta(y_l \lvert x)}{1-p_\theta(y_l \lvert x)}\right)$, where $p_\theta(y \lvert x) = \exp\left(\frac{1}{\lvert y \rvert}\log \pi_\theta(y \lvert x)\right)$
R-DPO [62]	$-\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \lvert x)}{\pi_{\text{ref}}(y_w \lvert x)} - \beta \log \frac{\pi_\theta(y_l \lvert x)}{\pi_{\text{ref}}(y_l \lvert x)} - (\alpha \lvert y_w \rvert - \alpha \lvert y_l \rvert)\right)$
SimPO	$-\log \sigma\left(\frac{\beta}{\lvert y_w \rvert}\log \pi_\theta(y_w \lvert x) - \frac{\beta}{\lvert y_l \rvert}\log \pi_\theta(y_l \lvert x) - \gamma\right)$

Main Results

- SimPO outperforms in most evals

- MT-Bench obtains poor separability

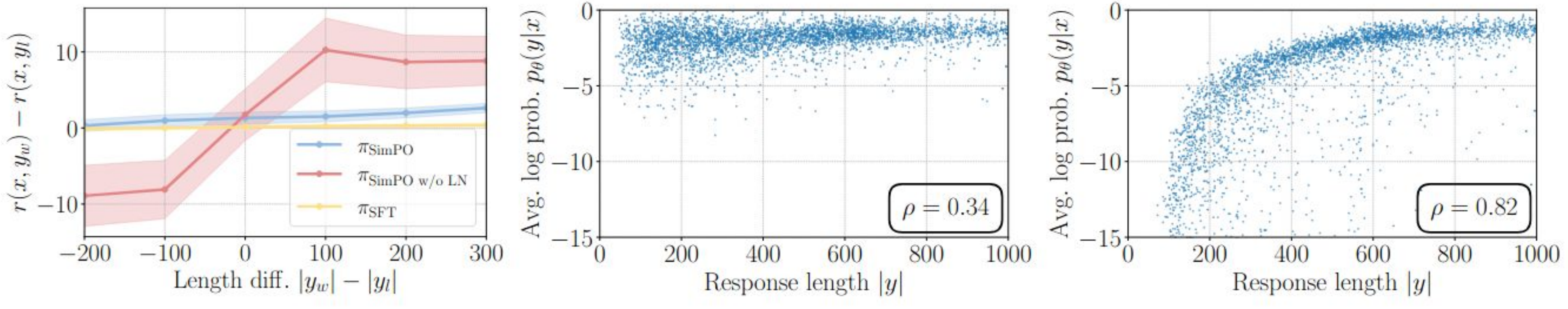
- Instruct versions perform better

- When controlling for length, SimPO does better

Method	Mistral-Base (7B)					Mistral-Instruct (7B)				
	AlpacaEval 2		Arena-Hard	MT-Bench		AlpacaEval 2		Arena-Hard	MT-Bench	
	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4
SFT	8.4	6.2	1.3	4.8	6.3	17.1	14.7	12.6	6.2	7.5
RRHF [87]	11.6	10.2	5.8	5.4	6.7	25.3	24.8	18.1	6.5	7.6
SLiC-HF [92]	10.9	8.9	7.3	5.8	**7.4**	24.1	24.6	18.9	6.5	**7.8**
DPO [64]	15.1	12.5	10.4	5.9	7.3	26.8	24.9	16.3	6.3	7.6
IPO [6]	11.8	9.4	7.5	5.5	7.2	20.3	20.3	16.2	6.4	**7.8**
CPO [84]	9.8	8.9	6.9	5.4	6.8	23.8	28.8	**22.6**	6.3	7.5
KTO [27]	13.1	9.1	5.6	5.4	7.0	24.5	23.6	17.9	6.4	7.7
ORPO [40]	14.7	12.2	7.0	5.8	7.3	24.5	24.9	20.8	6.4	7.7
R-DPO [62]	17.4	12.8	8.0	5.9	**7.4**	27.3	24.5	16.1	6.2	7.5
SimPO	**21.5**	**20.8**	**16.6**	**6.0**	7.3	**32.1**	**34.8**	21.0	**6.6**	7.6

Method	Llama3-Base (8B)					Llama3-Instruct (8B)				
	AlpacaEval 2		Arena-Hard	MT-Bench		AlpacaEval 2		Arena-Hard	MT-Bench	
	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4
SFT	6.2	4.6	3.3	5.2	6.6	26.0	25.3	22.3	6.9	8.1
RRHF [87]	12.1	10.1	6.3	5.8	7.0	31.3	28.4	26.5	6.7	7.9
SLiC-HF [92]	12.3	13.7	6.0	6.3	7.6	26.9	27.5	26.2	6.8	8.1
DPO [64]	18.2	15.5	15.9	6.5	7.7	40.3	37.9	32.6	**7.0**	8.0
IPO [6]	14.4	14.2	17.8	6.5	7.4	35.6	35.6	30.5	**7.0**	**8.3**
CPO [84]	10.8	8.1	5.8	6.0	7.4	28.9	32.2	28.8	**7.0**	8.0
KTO [27]	14.2	12.4	12.5	6.3	**7.8**	33.1	31.8	26.4	6.9	8.2
ORPO [40]	12.2	10.6	10.8	6.1	7.6	28.5	27.4	25.8	6.8	8.0
R-DPO [62]	17.6	14.4	17.2	**6.6**	7.5	41.1	37.8	33.1	**7.0**	8.0
SimPO	**22.0**	**20.3**	**23.4**	**6.6**	7.7	**44.7**	**40.5**	**33.8**	**7.0**	8.0

Length Normalization



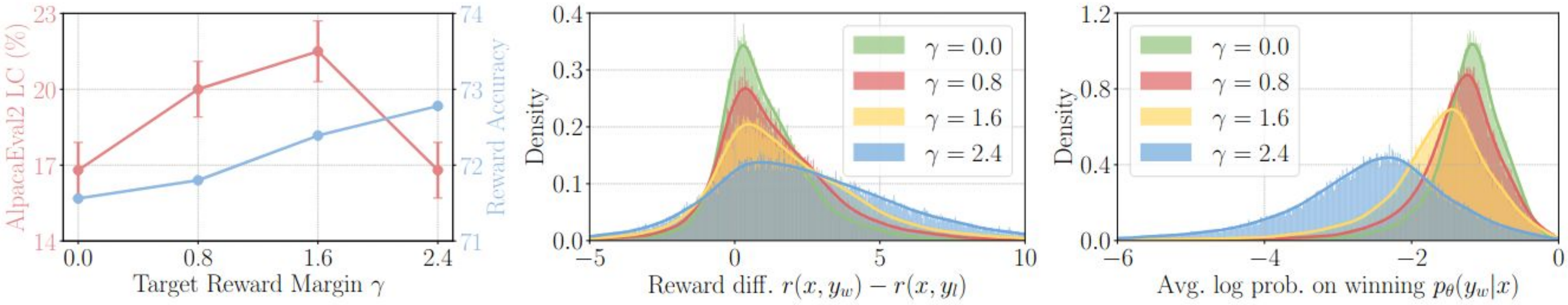
(a) Reward optimization.

(b) SimPO.

(c) SimPO without LN.

- LN leads to positive reward margin for all response pairs regardless of length
- Removing LN leads to signs of length exploitation

Target Reward Margin



(a) Performance w/ different γ. (b) Reward diff. distribution. (c) Log prob. distribution.

- Reward accuracy increases with target margin, but generation quality doesn't only depend on reward accuracy
- γ flattens the distributions of reward differences and log probabilities
- Added parameter that requires tuning

Ablation of Length Normalization and Reward Margin

Both elements are crucial to the success of SimPO

Method	Mistral-Base (7B) Setting					Mistral-Instruct (7B) Setting				
	AlpacaEval 2		Arena-Hard	MT-Bench		AlpacaEval 2		Arena-Hard	MT-Bench	
	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4
DPO	15.1	12.5	10.4	5.9	7.3	26.8	24.9	16.3	6.3	7.6
SimPO	21.5	20.8	16.6	6.0	7.3	32.1	34.8	21.0	6.6	7.6
w/o LN	11.9	13.2	9.4	5.5	7.3	19.1	19.7	16.3	6.4	7.6
$\gamma = 0$	16.8	14.3	11.7	5.6	6.9	30.9	34.2	20.5	6.6	7.7

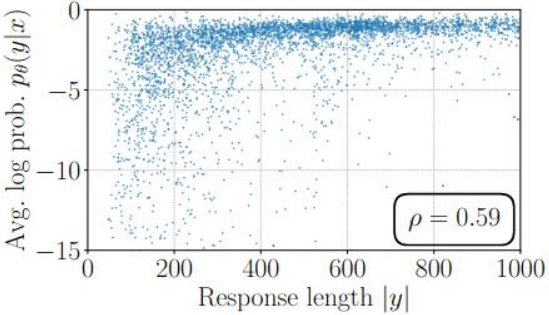
Ablation of Length Normalization and Reward Margin

Method	Mistral-Base (7B) Setting					Mistral-Instruct (7B) Setting				
	AlpacaEval 2		Arena-Hard	MT-Bench		AlpacaEval 2		Arena-Hard	MT-Bench	
	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4
DPO	15.1	12.5	10.4	5.9	7.3	26.8	24.9	16.3	6.3	7.6
SimPO	21.5	20.8	16.6	6.0	7.3	32.1	34.8	21.0	6.6	7.6
w/o LN	11.9	13.2	9.4	5.5	7.3	19.1	19.7	16.3	6.4	7.6
$\gamma = 0$	16.8	14.3	11.7	5.6	6.9	30.9	34.2	20.5	6.6	7.7

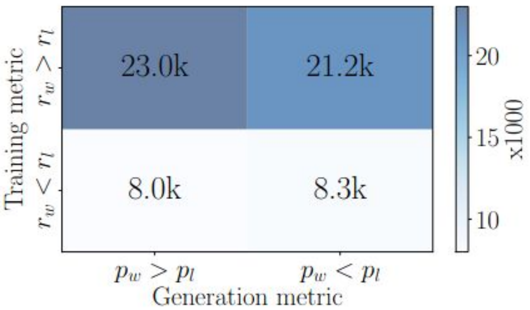
The same changes do not necessarily improve DPO

Method	Mistral-Base (7B) Setting					Mistral-Instruct (7B) Setting				
	AlpacaEval 2		Arena-Hard	MT-Bench		AlpacaEval 2		Arena-Hard	MT-Bench	
	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4	LC (%)	WR (%)	WR (%)	GPT-4 Turbo	GPT-4
SimPO	21.5	20.8	16.6	6.0	7.3	32.1	34.8	21.0	6.6	7.6
DPO	15.1	12.5	10.4	5.9	7.3	26.8	24.9	16.3	6.3	7.6
w/ LN	21.0	17.7	15.2	5.9	7.2	21.7	20.9	15.6	6.4	7.7
w/ γ	15.2	12.1	10.3	5.7	7.3	23.0	24.6	14.7	6.3	7.6

Comparisons with DPO

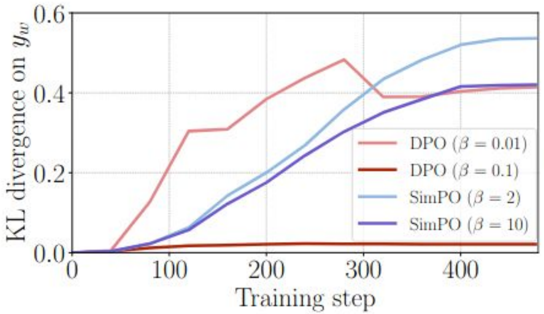


(a) Length correlation (DPO).

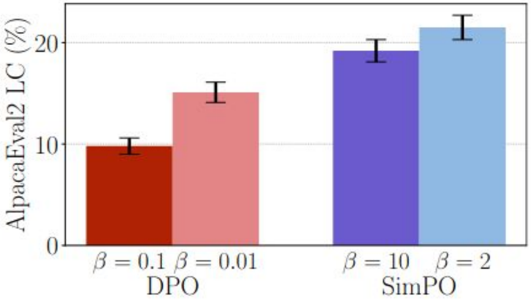


(b) Contingency table (DPO).

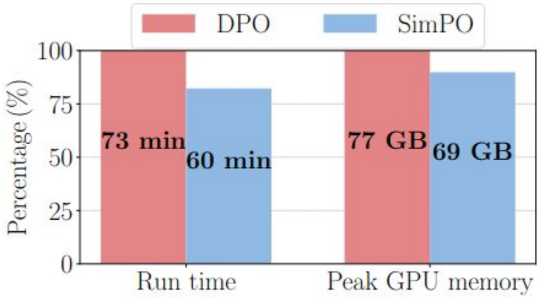
(c) Reward Accuracy.



(a) KL divergence w/ different β.



(b) Performance w/ different β.



(c) Efficiency of DPO vs. SimPO.

Other interesting points

- Empirically did not find much difference in other offline preference learning methods
- MMLU and general knowledge is largely retained across all methods
- Reading comprehension/common sense improves
- Truthfulness improves
- Math performance degrades
- Enhanced reward model (ArmoRM vs. PairRM) yields significant perf increase
- Strong SFT baselines and high-quality preference data make algorithm differences pretty minor
- SFT regularization added to SimPO leads to performance drop

Limitations

- No theoretical grounding or understanding of why this method works
- Evaluations only focus on helpfulness, not on other factors like safety or honesty which may be very important in real scenarios
- Performance drops on some downstream tasks, notably math