Beyond Online Sampling: Bridging Offline-to-Online Alignment via Dynamic Data Transformation for LLMs

Zhang Zhang[1], Guhao Feng[2], Jian Guan[3], Di He[*2], Wei Wu[*3]
[1]School of EECS, Peking University
[2]State Key Laboratory of General Artificial Intelligence,
School of Intelligence Science and Technology, Peking University
[3]Ant Group
{zzhang,fenguhao}@stu.pku.edu.cn, dihe@pku.edu.cn
{jianguanthu,wuwei19850318}@gmail.com

Abstract

While Direct Preference Optimization (DPO) eliminates complex reward modeling in aligning large language models (LLMs) with human preferences, its online variant faces significant efficiency bottlenecks due to costly real-time preference sampling and the reward model annotation. We propose a novel framework that bridges offline-to-online alignment by systematically transforming static datasets into dynamically adaptive equivalents, without the need for an explicit reward model. Our approach employs paraphrasing techniques to preserve response correctness while aligning data distributions with model-generated outputs, circumventing the need for resource-intensive online interactions. Experiments on mathematical reasoning and conversational tasks demonstrate that our method matches or exceeds the performance of a fully online DPO. This work establishes a computationally sustainable paradigm for LLM alignment, particularly benefiting scenarios requiring iterative preference updates and domain adaptation.

1 Introduction

Large language models (LLMs) have demonstrated remarkable capabilities in many natural language processing tasks (OpenAI, 2024; Vaswani et al., 2017). During the training of LLMs, reinforcement learning from human feedback (RLHF) (Christiano et al., 2017; Stiennon et al., 2020; Ouyang et al., 2022; Bai et al., 2022; Stiennon et al., 2022) has been widely adopted to refine model behaviors based on human evaluations.

Beyond the traditional RL Methods such as Proximal Policy Optimization (PPO) (Schulman et al., 2017), Direct Preference Optimization (DPO) (Rafailov et al., 2023a) has recently emerged as an alternative approach that simplifies alignment by directly optimizing preference comparisons without explicitly constructing a reward function. Despite its effectiveness, standard DPO methods are applied to static datasets, limiting their adaptability to evolving user preferences and new task distributions (Song et al., 2024; Dong et al., 2024). To address this limitation, online DPO (Guo et al., 2024) extends the principles of DPO to an interactive setting, dynamically collecting and using preference data during training. This iterative approach enables adaptive model updates and has been shown to outperform offline methods. However, online data collection introduces significant efficiency challenges, as sampling sufficient high-quality preference data can be computationally expensive and time-consuming (Guo et al., 2024), and a large quantity of work has been working on reducing the cost of online sampling (Stiennon et al., 2022; Schulman et al., 2018; Nikolenko, 2019; Furlanello et al., 2016).

To address the efficiency problem of online DPO, we propose a novel approach, Dynamic Data Transformation (DDT), for transforming offline datasets into online-equivalent datasets, mitigating the inefficiencies of online data collection while preserving the benefits of adaptive learning, without needing a reward model to annotate the generated answers. Our method systematically enhances offline datasets by leveraging paraphrasing techniques that maintain response correctness while improving alignment with model-generated distributions. We validate our approach through experiments on mathematical problem-solving tasks, where deterministic evaluation allows for precise accuracy measurements, as well as on more diverse and subjective chat-based tasks. Our results demonstrate that our method achieves comparable, and in some cases, superior, performance to fully online DPO while significantly reducing the computational burden. Besides, by utilizing the information in the offline dataset, our method does not need

[*]Correspondence: dihe@pku.edu.cn, wuwei19850318@gmail.com

a reward model for annotating the generated answers, thus getting rid of the reliance on the reward model and saving annotation time. By bridging the gap between offline and online preference optimization, our work contributes to more efficient and scalable alignment strategies for LLMs, paving the way for improved real-world applicability in domains requiring precise and adaptive responses.

2 Preliminary

Direct Preference Optimization. DPO is a fine-tuning method designed to align LLMs with human preferences. Unlike traditional RLHF methods that require reward modeling and policy optimization, DPO directly optimizes the model based on preference comparisons $(x, y_w, y_l) \sim \mathcal{D}$, where x represents the prompt and y_w, y_l represents the winning and losing response, respectively. Given a dataset of ranked responses, DPO updates the model by maximizing the probability of preferred responses over disfavored ones without explicitly constructing a reward function. Specifically, the DPO process is implemented through optimizing the following objective:

$$\mathcal{L}_{\text{DPO}}(\pi_\theta; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log \sigma\left(\beta \log \frac{\pi_\theta(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_\theta(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)}\right)\right] \tag{1}$$

where π_{ref} represents the original model and π_θ represents the current model. Rafailov et al. (2023b) proves that the traditional RL optimal solution is contained in the DPO optimal solution. Furthermore, with unlimited data, it guarantees the same optimal solution as the traditional RL methods.

Online DPO. Although simplicity and theoretical guarantees, with limited offline data, DPO does not guarantee that optimizing to the DPO objective gives the best solution to the KL-constrained reward. To address this problem, Guo et al. (2024) extends DPO to an iterative setting, where preference data are collected dynamically, enabling adaptive model updates. Xiong et al. (2024) further gives a theoretical explanation for why online iterative RLHF is better than offline RLHF, which further proves the crucial role of online iterative data collection. However, the low sampling efficiency inherent to online sampling limits the practical effectiveness of Online DPO.

3 Method

This section introduces our method pipeline and generalizes to the iterative online DPO process. Subsequently, we give our understanding for why our method works.

3.1 Offline to Online Data Conversion

Many previous works have demonstrated the significance of sampling the online data (Xiong et al., 2024; Han et al., 2024), *i.e*, the data sampling from $\pi_\theta(y \mid x)$. Due to the sampling inefficiency of the online dataset, it would be more efficient while still effective if we could convert an offline dataset into an online one. Here we provide our method that can fulfill this requirement:

Initialization At the beginning we have an offline dataset $\mathcal{D}_{\text{off}}$ and a reference policy π_{ref} that is waiting for fine-tuning. The offline dataset $\mathcal{D}_{\text{off}}$ has an accurate reward but lacks the online property, which has a small log probability on π_{ref}.

Online Data Generation Given an offline dataset $\mathcal{D}_{\text{off}}$ with reliable reward accuracy, we leverage the current model to paraphrase $\mathcal{D}_{\text{off}}$ into a new dataset $\mathcal{D}_{\text{DDT}}$. This process yields a reformulated dataset that more closely resembles an online setting. Formally, this conversion can be expressed as:

$$\mathcal{D}_{\text{DDT}} = \{(x, y'_w, y'_l) \big| (x, y_w, y_l) \sim \mathcal{D}_{\text{off}}, \\ y'_w \sim \pi_{\text{ref}}(y|x, y_w, z), y'_l \sim \pi_{\text{ref}}(y|x, y_l, z)\}, \tag{2}$$

where z delineates the prompt that guides the language model to paraphrasing the original response in $\mathcal{D}_{\text{off}}$ while maintaining the main content.

Iterative Enhancement We can easily generalize our method to the iterative setting. As shown in Figure 1, for each iteration, we use our current policy to paraphrase the given dataset and train with DPO to get the policy for the next iteration.

Assumption Our method is based on a key observation that whether the response y to a given prompt x looks like an online sample or not is highly correlated with the speaking style of y. However, the reward value for the response is more correlated with the content than the speaking style. With this observation, the sample $(y'_w, y'_l) \sim (\pi_{\text{ref}}(y|x, y_w, z), \pi_{\text{ref}}(y|x, y_l, z))$, achieves online property due to their preservation of π_{ref}'s speaking style. Concurrently, they maintain the initial reward, as the prompt z provides instructions that ensure the content and quality remain intact. Therefore, the altered dataset can be considered akin to an online dataset by our online dataset metrics.

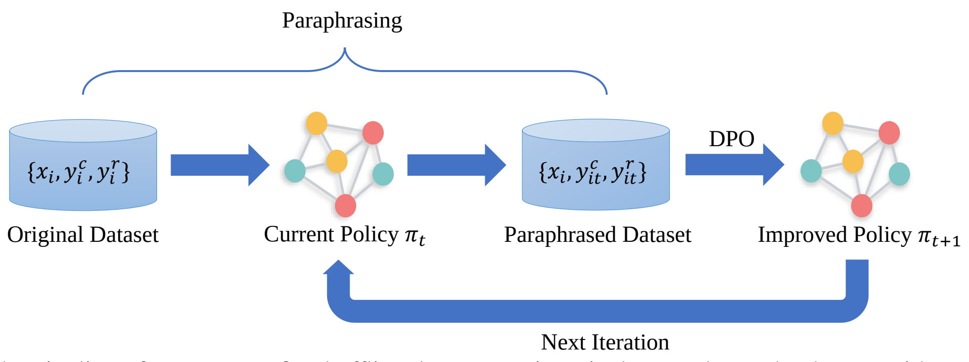

Figure 1: The pipeline of DDT. For a fixed offline dataset, we iteratively paraphrase the dataset with DDT and train with DPO to get the policy for the next iteration. In particular, π_1 represents the original policy.

GSM8k

	Mistral-7B-Instruct-v0.2				**Meta-Llama-3-8B-Instruct**			
Approach	Original	Offline	Online	DDT	Original	Offline	Online	DDT
ACC	41.32	48.74	<u>50.87</u>	**51.48**	74.37	73.01	**79.30**	<u>79.00</u>

AlpacaEval

	Mistral-7B-Instruct-v0.2				**Meta-Llama-3-8B-Instruct**			
Approach	Original	Offline	Online	DDT	Original	Offline	Online	DDT
LC	17.10	<u>27.31</u>	24.53	**28.94**	20.86	30.45	**34.61**	<u>31.01</u>
WR	14.70	<u>27.88</u>	27.31	**30.96**	21.18	30.65	**35.31**	<u>33.51</u>

Table 1: Results on MetaMathQA and Ultrafeedback using Dynamic Data Transformation (DDT) and custom methods. For GSM8k, we use the model trained on MetaMathQA with different approaches to test the accuracy of GSM8k. For AlpacaEval, we use the model trained on UltraFeedback with different approaches and test the win rate and length-controlled win rate on the AlpacaEval benchmark.

4 Experiments

4.1 Math

First, we test our method on mathematics, which is deterministic, well-defined, and easy to test. We list our basic settings, and for more details, please refer to Appendix B.1.

4.1.1 Experiment Setup

Base Model In our experiments we use Mistral-7B-Instruct-v0.2[1] and Meta-Llama-3-8B-Instruct[2] as our base model.

Original Dataset Our initial dataset is MetaMathQA[3], which provides numerous questions and the corresponding ground truth answers.

Evaluation Metrics We evaluated our model's performance based on the accuracy of the gsm8k[4] dataset. We measured its answer accuracy using our custom implementation for inference and answer extraction. We set the temperature at 0.7 for all evaluations.

Dataset Categories Our offline dataset $\mathcal{D}_{\text{off}}$ is a DPO dataset whose winning response is directly copied from the original MetaMathQA. In contrast, the losing response is sampled from the current model and judged by the answer-extracting function, ensuring that the losing response provides an incorrect answer to the problem. We generate answers in many rounds until half of the prompts accept a generated wrong answer. The truly online generated dataset $\mathcal{D}_{\text{on}}$ is sampled directly from the current model and judged by the answer extraction function. We also generate multiple rounds until half of the prompts accept both a correct and a wrong answer. Our online dataset $\mathcal{D}_{\text{DDT}}$ is a current model modified version of $\mathcal{D}_{\text{off}}$, whose generation process is described in Section 3.1.

4.1.2 Sampling Inefficiency for Online Data

To sample the online dataset for math tasks, the sampled results should contain at least one correct and one incorrect answer. We have the intuition

[1]https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2

[2]https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

[3]https://huggingface.co/datasets/meta-math/MetaMathQA

[4]https://huggingface.co/datasets/openai/gsm8k

that sampling two rounds gives at most half of the dataset, since if the model gives the correct answer with a probability at least $\frac{1}{2}$, then it would be difficult to sample a wrong answer, and vice versa. Theoretically, in Appendix A we proved that the sampling time in expectation is at least $\frac{3}{2}$ times for generating an online dataset than our method, with equality taken only in perfect condition. Empirically, in Appendix B.1.3 we tested the sampling time and found that our method is about 4 times faster than the online sampling process for Mistral-7B-Instruct-v0.2 and Meta-Llama-3-8B-Instruct, and would be faster for a model with accuracy farther away to $\frac{1}{2}$.

4.2 Chat

For detailed training process, please refer to Appendix B.2. Most of the details are the same as Appendix B.1, except for the following mentioned implementation:

4.2.1 Experimental Setup

Original Dataset Our initial dataset is UltraFeedback[5], which contains 64k single-round question-answering. In the experiment, we utilize the dataset mistral-instruct-ultrafeedback[6] and llama3-ultrafeedback[7] in the work Meng et al. (2024). In detail, we use Meta-Llama-3-8B-Instruct-sampled, llm-blender/PairRM[8] annotated dataset llama3-ultrafeedback[9] as the offline dataset for Mistral-7B-Instruct-v0.2 and the online dataset for Meta-Llama-3-8B-Instruct, and vice versa.

Evaluation Metrics We use AlpacaEval (Dubois et al., 2024) as our evaluation metric. We set the temperature to 0.7 for Mistral-7B-Instruct-v0.2 and 0.9 for Meta-Llama-3-8B-Instruct when sampling answers on the AlpacaEval dataset. We adopt OpenAI's GPT-4-1106-preview model as both the annotator and the evaluation baseline.

Dataset Categories The online and offline datasets have been described in Section 4.1.1. The DDT dataset is paraphrased directly from the offline dataset since ultrafeedback contains paired responses.

[5]https://huggingface.co/datasets/openbmb/UltraFeedback

[6]https://huggingface.co/datasets/princeton-nlp/mistral-instruct-ultrafeedback

[7]https://huggingface.co/datasets/princeton-nlp/llama3-ultrafeedback

[8]https://huggingface.co/llm-blender/PairRM

[9]https://huggingface.co/datasets/princeton-nlp/llama3-ultrafeedback

Meta Math Multiple Round				
	Llama-3-8B		**Mistral-7B**	
Approach	Online	DDT	Online	DDT
Original	74.37	74.37	41.32	41.32
1st Round	**79.30**	<u>79.00</u>	<u>50.87</u>	**51.48**
2nd Round	**81.80**	<u>79.76</u>	**54.44**	<u>52.92</u>

Table 2: Accuracy (%) of multiple-round DDT on MetaMathQA. For the online method, we sample online data and judge the correctness iteratively. For DDT, the original dataset provides the correct answer, and we sample the wrong answer for each round and then paraphrase to get our modified dataset.

4.2.2 Sampling Inefficiency for Online Data

Sampling online chat data faces similar inefficiency issues as in math tasks (see Section 4.1.2). However, in this case, response quality is more ambiguous, and rewards are more continuous. To ensure high-quality data with clear reward gaps, each prompt is sampled multiple times. For example, both llama3-ultrafeedback and mistral-instruct-ultrafeedback are sampled five times before annotation by the reward model; otherwise, the reward gap between winning and losing responses would be insufficient, degrading performance.

4.3 Results

Table 1 shows the performance on our DDT dataset and the custom datasets. In both chat and math, our method surpasses the original model and the model trained on the offline dataset. On Mistral-7B-Instruct-v0.2, our method even surpasses the online method in both math and chat, which may be due to the high quality of the offline dataset sampled from Meta-Llama-3-8B-Instruct. Table 2 shows the results for multiple round experiments, as described in Figure 1. Our method provides a straightforward and reliable training pipeline, with steady improvements in model performance. For detailed experiment results and description please refer to Appendix B.3.

5 Conclusion

In this work, we introduced an approach to enhance offline datasets for DPO, transforming them into online-equivalent datasets that retain the benefits of adaptive learning while mitigating the reliance on a reward model. Through experiments on both mathematical problem-solving and chat-based tasks, we demonstrated that our method enhances the performance on offline datasets and achieves performance comparable to online DPO while significantly reducing computational costs. By bridging the gap between offline and online preference optimization, our approach provides a convenient alignment strategy for large language models, improving their adaptability and real-world applicability.

6 Limitations

There may exist algorithms or techniques for transforming the offline dataset into an online dataset within the DPO setting. In this study, we primarily utilize the original models to paraphrase the datasets. We leave the exploration of more advanced methods to future research.

7 Acknowledgement

DH is supported by the National Science and Technology Major Project (2022ZD0114902) and the National Science Foundation of China (NSFC62276007).

References

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas. Association for Computational Linguistics.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, Nicholas Joseph, Saurav Kadavath, Jackson Kernion, Tom Conerly, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Danny Hernandez, Tristan Hume, Scott Johnston, Shauna Kravec, Liane Lovitt, Neel Nanda, Catherine Olsson, Dario Amodei, Tom Brown, Jack Clark, Sam McCandlish, Chris Olah, Ben Mann, and Jared Kaplan. 2022. Training a helpful and harmless assistant with reinforcement learning from human feedback.

Paul F. Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. 2017. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 4299–4307.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. 2024. Rlhf workflow: From reward modeling to online rlhf.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators.

Tommaso Furlanello, Jiaping Zhao, Andrew M. Saxe, Laurent Itti, and Bosco S. Tjan. 2016. Active long term memory networks.

James Goodman, Andreas Vlachos, and Jason Naradowsky. 2016. Noise reduction and targeted exploration in imitation learning for Abstract Meaning Representation parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1–11, Berlin, Germany. Association for Computational Linguistics.

Shangmin Guo, Biao Zhang, Tianlin Liu, Tianqi Liu, Misha Khalman, Felipe Llinares, Alexandre Rame, Thomas Mesnard, Yao Zhao, Bilal Piot, Johan Ferret, and Mathieu Blondel. 2024. Direct language model alignment from online ai feedback.

Jiaqi Han, Mingjian Jiang, Yuxuan Song, Jure Leskovec, Stefano Ermon, and Minkai Xu. 2024. f-po: Generalizing preference optimization with f-divergence minimization.

Mary Harper. 2014. Learning from 26 languages: Program management and science in the babel program. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, page 1, Dublin, Ireland. Dublin City University and Association for Computational Linguistics.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pages 611–626.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward.

Sergey I. Nikolenko. 2019. Synthetic data for deep learning.

OpenAI. 2024. Gpt-4 technical report.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2023a. Direct preference optimization: Your language model is secretly a reward model. In *NeurIPS*.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D Manning, and Chelsea Finn. 2023b. Direct preference optimization: Your language model is secretly a reward model. *ArXiv preprint*, abs/2305.18290.

John Schulman, Xi Chen, and Pieter Abbeel. 2018. Equivalence between policy gradients and soft q-learning.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms.

Yuda Song, Gokul Swamy, Aarti Singh, Drew Bagnell, and Wen Sun. 2024. The importance of online data: Understanding preference fine-tuning via coverage. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Nisan Stiennon, Long Ouyang, Jeff Wu, Daniel M. Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul Christiano. 2022. Learning to summarize from human feedback.

Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. 2020. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Wei Xiong, Hanze Dong, Chenlu Ye, Ziqi Wang, Han Zhong, Heng Ji, Nan Jiang, and Tong Zhang. 2024. Iterative preference learning from human feedback: Bridging theory and practice for rlhf under kl-constraint. In *Forty-first International Conference on Machine Learning*.

A Sampling Time in Calculation for Online Datasets

In this section, we provide two remarks on the sampling time for online datasets on math problems. These two remarks proved that our method enhances the sampling efficiency for at least $\frac{3}{2}$ times in theory, and the acceleration would be more obvious in practice due to the strict condition of equality in the remark.

A.1 Sampling Time

Remark A.1. Assume that we construct the online dataset by multi-round sampling. For each round, we only sample the prompts that lack either a correct or a wrong answer. We define sampling time as the number of prompts the current model receives. If we want the expectation of the length of the final generated dataset to be at least $f(f \leq 1)$ fraction of the original dataset, then the total sampling time would be at least $(1+2f)N$ in expectation, where N represents the total number of prompts in the original dataset. The equality is taken only when the accuracy on all prompts equals $\frac{1}{2}$.

Proof. Suppose the remaining number after s rounds is R_s, then the sampling time for the $s+1$-th round would be R_s. Assume after S rounds, the expectation of the length of the generated dataset is at least f fraction of the original dataset, *i.e.* $\mathbb{E}[R_S] \leq (1-f)N$. We also assume that the correct probability on the i-th prompt is p_i. The expectation of the sampling time would be $T = \mathbb{E}[N + \sum_{s=1}^{S-1} R_s] = N + \sum_{s=1}^{S-1} \mathbb{E}[R_s]$.

Now we calculate the expression of $\mathbb{E}[R_s]$. Let I_{si} denote a random variable that satisfies $I_{si} = 1$ iff after s rounds the i-th prompt lacks either a correct answer or an incorrect answer. Thus we would obtain:

$$\mathbb{E}[R_s] = \sum_{j=1}^{N} \mathbb{E}[I_{sj}] = \sum_{j=1}^{N} \mathbb{P}[I_{sj} = 1] = \sum_{j=1}^{N} ({p_j}^s + (1-p_j)^s) \tag{3}$$

By summing over s we would obtain the expression of T:

$$\begin{aligned} T &= N + \sum_{s=1}^{S-1}\sum_{j=1}^{N} ({p_j}^s + (1-p_j)^s) \\ &= N + \sum_{j=1}^{N}\sum_{s=1}^{S-1} ({p_j}^s + (1-p_j)^s) \\ &= N + \sum_{j=1}^{N} (\frac{p_j - {p_j}^S}{1-p_j} + \frac{1-p_j-(1-p_j)^S}{p_j}) \\ &= N + \sum_{j=1}^{N} (\frac{1 - {p_j}^S}{1-p_j} + \frac{1-(1-p_j)^S}{p_j} - 2) \\ &= \sum_{j=1}^{N} (\frac{1 - {p_j}^S}{1-p_j} + \frac{1-(1-p_j)^S}{p_j}) - N \\ &\geq \sum_{j=1}^{N} 2((1-{p_j}^S) + (1-(1-p_j)^S)) - N \\ &= 3N - 2\sum_{j=1}^{N} ({p_j}^S + (1-p_j)^S) \\ &= 3N - 2\mathbb{E}[R_S] \\ &\geq 3N - 2(1-f)N \\ &= (1+2f)N \end{aligned} \tag{4}$$

In the proof we utilized an inequality:

$$\frac{1 - {p_j}^S}{1-p_j} + \frac{1-(1-p_j)^S}{p_j} \geq 2((1-{p_j}^S) + (1-(1-p_j)^S)) \tag{5}$$

This is because:

$$\begin{aligned}
&\frac{1-{p_j}^S}{1-p_j}+\frac{1-(1-p_j)^S}{p_j}-2((1-{p_j}^S)+(1-(1-p_j)^S))\\
&=\frac{1-{p_j}^S}{1-p_j}+\frac{1-(1-p_j)^S}{p_j}-(2(1-p_j)\frac{1-{p_j}^S}{1-p_j}+2p_j\frac{1-(1-p_j)^S}{p_j})\\
&=(2p_j-1)\frac{1-{p_j}^S}{1-p_j}+(1-2p_j)\frac{1-(1-p_j)^S}{p_j}\\
&=(2p_j-1)(\frac{1-{p_j}^S}{1-p_j}-\frac{1-(1-p_j)^S}{p_j})\\
&=(2p_j-1)(\sum_{i=0}^{S-1}({p_j}^i-(1-p_j)^i))\\
&=\sum_{i=0}^{S-1}(2p_j-1)({p_j}^i-(1-p_j)^i)\\
&\geq 0
\end{aligned} \tag{6}$$

The last step is because $(2p_j-1)({p_j}^i-(1-p_j)^i)\geq 0$, which can be verified either by expanding the expression ${p_j}^i-(1-p_j)^i$ or by discussing whether $p_j>\frac{1}{2}$. From the above proof we can see that equality is possible only when $p_j=\frac{1}{2},\forall j\in[N]$. $\square$

From the above remark, we can see that the average sample time per data pair (a data pair contains a winning response and a losing response) would be at least $\frac{(1+2f)N}{fN}=2+\frac{1}{f}\geq 3$. While DDT only needs to sample twice per data pair, the average acceleration rate per data point in expectation would be at least $\frac{3}{2}$.

A.2 Sampling Rounds

Remark A.2. We use the same sampling method as described in Remark A.1. Assume the total accuracy on the dataset of the model to be p, *i.e.* $\sum_{i=1}^{N}p_i=pN$, where p_i represents the accuracy of the model on the i-th prompt, and N represents the total number of the dataset. If we want the expectation of the length of the final generated dataset to be at least f fraction of the original dataset, then the total sampling round S would need to satisfy $1-p^S-(1-p)^S\geq f$. The equality is taken only when $p_1=p_2=\cdots=p_N=p$.

Proof. From the calculation in Appendix A.1, we obtain $\mathbb{E}[R_S]=\sum_{j=1}^{N}({p_j}^S+(1-p_j)^S)$. Since $\mathbb{E}[R_S]\leq(1-f)N$, with the help of power mean inequality we have:

$$\begin{aligned}
(1-f)N&\geq\mathbb{E}[R_S]\\
&=\sum_{j=1}^{N}({p_j}^S+(1-p_j)^S)\\
&=\sum_{j=1}^{N}{p_j}^S+\sum_{j=1}^{N}(1-p_j)^S\\
&\geq N\left(\frac{\sum_{j=1}^{N}p_j}{N}\right)^S+N\left(\frac{\sum_{j=1}^{N}(1-p_j)}{N}\right)^S\\
&=Np^S+N(1-p)^S
\end{aligned} \tag{7}$$

Thus we have $p^S+(1-p)^S\leq 1-f$, where the equality is possible only when $p_j=p,\forall j\in[N]$. $\square$

B Experiment Details

In this section, we provide the detailed implementation of our experiments. All experiments are conducted using PyTorch on $8 \times$A100 GPUs. Our implementation is based on the open-source alignment-handbook[10] and the vLLM library (Kwon et al., 2023). Each experiment is executed once.

B.1 Math

This section describes the detailed training process on MetaMathQA with custom methods and our method.

B.1.1 Dataset Sampling

Mistral-7B-Instruct-v0.2 and Meta-Llama-3-8B-Instruct share the same sampling parameters.

For the offline dataset, we first select a subset containing 50% of the MetaMathQA prompts at random. We use a sampling temperature of 0.7, generating responses that are subsequently evaluated using our answer-extraction function. The sampling process terminates when 50% of the problems in the selected subset receive an incorrect answer from the current model and join them with the original answer from the MetaMathQA dataset. The generating process guarantees that the final sampled dataset should contain at least 25% elements in the original dataset, and each winning answer is from the original dataset, while the losing answer is sampled from the current model.

For the online dataset, we adopt the same sampling temperature of 0.7 and evaluate responses with the same answer-extraction function. As before, sampling continues until 25% of the problems in MetaMathQA yield both a correct and an incorrect answer from the model.

For the DDT dataset, we use a sampling temperature of 0.1 and carefully designed prompts to paraphrase the offline dataset. The winning and losing responses are generated using shared paraphrasing prompts, ensuring that their paraphrased outputs maintain the same original answers and thus correspond to the modified winning and losing responses.

B.1.2 Training

For Mistral-7B-Instruct-v0.2, we chose $\beta = 0.3$ and $lr = 5e^{-7}$, while for Meta-Llama-3-8B-Instruct we chose $\beta = 0.7$ and $lr = 5e^{-7}$. The training hyperparameters are shared for the three different kinds of datasets.

B.1.3 Sampling Time in Experiment

We test the real sampling time on 1000 prompts for the online setting. We set $f = \frac{1}{2}$, *i.e.* we want the length of the final dataset to be at least half of the original dataset. The total sampling time and the corresponding sampling round are shown in Table 3 below:

Meta Math Sampling Process				
Round	**Meta-Llama-3-8B-Instruct**		**Mistral-7B-Instruct-v0.2**	
	Total Num	**Remain Num**	**Total Num**	**Remain Num**
1st Round	987	987	987	987
2nd Round	987	740	987	734
3rd Round	740	630	734	612
4th Round	630	553	612	535
5th Round	553	492	535	481
Total Sampling Time	3897		3855	

Table 3: Meta Math Sampling Process on Meta-Llama-3-8B-Instruct and Mistral-7B-Instruct-v0.2. The table shows the total number of sampling answers and the remaining number of answers that do not satisfy the condition for each round.

[10] `https://github.com/huggingface/alignment-handbook`

Meta-Llama-3-8B-Instruct and Mistral-7B-Instruct-v0.2 has the overall accuracies of 74.37% and 41.32%, respectively. Both models undergo a total of five sampling rounds, with cumulative sampling times of 3897 and 3855, respectively. The average sampling time per prompt is 7.87 for Meta-Llama-3-8B-Instruct and 7.62 for Mistral-7B-Instruct-v0.2, both of which are substantially higher than that of our method, which maintains a constant average sampling of exactly 2. Besides, the accuracy of Mistral-7B-Instruct-v0.2 is closer to $\frac{1}{2}$ than Meta-Llama-3-8B-Instruct, and its total sampling times are fewer, which validates our calculation in Appendix A.1 that the equality is only possible when the accuracy on all prompts equals $\frac{1}{2}$.

B.1.4 Answer Accuracy

The answer accuracy is the proportion of paraphrased responses that retain the original answer. Specifically, we define the answer accuracy as:

$$Acc_w(\mathcal{D}_{\text{DDT}}) = \mathbb{E}_{(x,y'_w,y'_l)\sim\mathcal{D}_{\text{DDT}}}\mathbb{I}[\text{A}(y'_w) = \text{A}(y_w)] \tag{8}$$

$$Acc_l(\mathcal{D}_{\text{DDT}}) = \mathbb{E}_{(x,y'_w,y'_l)\sim\mathcal{D}_{\text{DDT}}}\mathbb{I}[\text{A}(y'_l) = \text{A}(y_l)] \tag{9}$$

Here, A denotes a regular-expression-based extraction function that retrieves the number following a fixed template, while y_w, y_l are the original answers prior to paraphrasing. We test the answer accuracy for both Meta-Llama-3-8B-Instruct and Mistral-7B-Instruct-v0.2 when paraphrasing the original MetaMathQA dataset. The detailed results are shown in the following Table 4.

	Reward Accuracy	
	Meta-Llama-3-8B-Instruct	**Mistral-7B-Instruct-v0.2**
DDT Win	0.9057	0.7787
DDT Lose	0.8736	0.7541

Table 4: The answer accuracy for Meta-Llama-3-8B-Instruct and Mistral-7B-Instruct-v0.2 when paraphrasing the MetaMathQA dataset.

From the above experiment, we can see that our prompt for math retains the original answer most of the time, which corresponds to keeping the original reward from DPO's perspective.

B.2 Chat

In this section, we describe the detailed training process on UltraFeedback with custom methods and our method.

B.2.1 Dataset Sampling

The original datasets are directly downloaded from princeton-nlp[11]. For our method, we adopt the same sampling temperature of 0.1 with our math setting.

B.2.2 Training

For Mistral-7B-Instruct-v0.2, we set $lr = 5e^{-7}$, while for Meta-Llama-3-8B-Instruct we chose $lr = 7e^{-7}$. β are set to 0.01 for both models. The training hyperparameters are shared for the three different kinds of datasets.

B.2.3 Reward Accuracy

We first define the reward accuracy of a dataset $\mathcal{D}$:

$$Acc_R(\mathcal{D}) = \mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}\mathbb{I}[r(x,y_w) > r(x,y_l)], \tag{10}$$

[11] https://huggingface.co/princeton-nlp

where $\mathbb{I}$ is the indicator function.

Then we test the reward accuracy on various datasets. In detail, we use ArmoRM-Llama3-8B-v0.1[12] as a reward model and test the reward accuracy Equation (10) on various datasets. The results are shown in the following Table 5:

Ultrafeedback Reward Accuracy		
	Meta-Llama-3-8B-Instruct	**Mistral-7B-Instruct-v0.2**
Offline	65.8	66.6
Online	66.6	65.8
DDT	**80.6**	**78.7**

Table 5: Reward Accuracy on Meta-Llama-3-8B-Instruct and Mistral-7B-Instruct-v0.2. The offline, online, and DDT datasets are described in Section 4.2.1. The reward accuracy is defined in Equation (10).

The results show that our method significantly improves reward accuracy, even when the model is provided with only a winning or a losing response individually, without explicitly generating a pairwise winning–losing comparison. This improvement may be attributed to the possibility that paraphrasing a response reveals underlying preference-related features—for instance, it may clarify the content and better expose the implicit intent or quality of the original response.

B.3 Multiple Round Experiment

In this section, we will talk about the multiple-round experiment for metamath and ultrafeedback datasets.

B.3.1 Meta Math

We ran a multiple-round experiment using our method and the custom online data generation method. For the online method, we sample new data from our current model and judge the correctness, then run DPO on the sampled dataset with the current model as the reference model. For our method DDT, we first sample an incorrect answer with the current model to get a losing answer and combine it with the ground truth answer in the dataset to get a paired response. This time we call this the offline dataset since the winning answers are fixed offline responses, even though the losing answers are online-sampled. Then we use our current model to paraphrase both to get the DPO dataset and train on it. This process is repeated for three rounds. As shown in Table 2, our method can continuously improve performance while reducing the total cost.

B.3.2 Ultrafeedback

For the ultrafeedback dataset, the online dataset is sampled from the current policy five times, and we let a stronger reward model ArmoRM-Llama3-8B-v0.1 choose the best one and the worst one. For the DDT setting, the only difference is that this time we fix the offline dataset as the original dataset for the original model. For Meta-Llama-3-8B-Instruct, the offline dataset is the dataset sampled from Mistral-7B-Instruct-v0.2, and vice versa. As shown in Table 6, although our method is not as good as the online-sampled dataset, it still continuously enhance the performance while reducing the total cost.

B.4 Prompts

This section details the prompts employed in the training-stage paraphrasing and in answer sampling for GSM8k accuracy testing.

B.4.1 Prompts for Math

Here we provide our prompts for DDT on the math task in Table 7.

B.4.2 Prompts for Chat

Here we provide our prompts for DDT on the conversation task in Table 8.

[12]https://huggingface.co/RLHFlow/ArmoRM-Llama3-8B-v0.1

	Meta Math Multiple Round				Ultrafeedback Multiple Round	
	Meta-Llama-3-8B-Instruct		Mistral-7B-Instruct-v0.2		Meta-Llama-3-8B-Instruct	
Approach	Online	DDT	Online	DDT	Online	DDT
Original	74.37	74.37	41.32	41.32	20.86	20.86
1st Round	**79.30**	79.00	50.87	**51.48**	**47.89**	40.95
2nd Round	**81.80**	79.76	**54.44**	52.92	**50.28**	45.89

Table 6: Combined results of Meta Math and Ultrafeedback multiple rounds.

MetaMathQA Prompt
You are an AI whose job is to generate answers to the given math problems. You will be given a problem and a reference answer, and you should generate your own answer with the same result and logical reasoning but with your own speaking style. Conclude with 'The answer is: ' followed by the answer as a number. Please provide the rewritten response in the following format: <Rewritten Response>: <your rewritten response> Here is the information you need: <Prompt>: prompt <Response>: response

Table 7: Prompt for both Llama-3-8B-Instruct and Mistral-7B-Instruct in paraphrasing answers for MetaMathQA.

B.4.3 Prompts for GSM8k

Here we provide our prompts for testing the accuracy on GSM8k for Meta-Llama-3-8B-Instruct in Table 9 and Mistral-7B-Instruct-v0.2 in Table 10. We use a regular expression for answer extraction. Specifically, we find the number followed by "The answer is" in the response as our extracted answer. Due to the weaker command following ability for Mistral-7B-Instruct-v0.2, we designed the prompt more carefully and manually excluded the possible mistakes in the prompt.

Ultrafeedback Prompt
I have a response for a given prompt, and I want you to rewrite the response while maintaining its original quality, intent and meaning. Please provide the rewritten response in the following format: <Rewritten Response>: <your rewritten response> Here is the information you need: <Prompt>: prompt <Response>: response

Table 8: Prompt for both Llama-3-8B-Instruct and Mistral-7B-Instruct in paraphrasing answers for Ultrafeedback.

GSM8k Prompt for Llama
Please answer the following question and conclude with ‘The answer is: ’ followed by the answer as a number.

Table 9: Prompt for Meta-Llama-3-8B-Instruct in inferencing answers for GSM8k.

GSM8k Prompt for Mistral
Please answer the following question and conclude with ‘The answer is: ’ followed by the answer as a number. Please first generate your reasonings, and then conclude with ‘The answer is: <number>’ where <number> is the answer as a single number. Please don’t put any other words after the answer.

Table 10: Prompt for Mistral-7B-Instruct-v0.2 in inferencing answers for GSM8k.