Synthetic Preference Interpolation for Language Model Alignment

Anonymous ACL submission

Abstract

Ensuring alignment with human preferences is a critical and challenging aspect of large language models (LLMs). Currently, the most widely adopted alignment methods, such as those based on Direct Preference Optimization (DPO), leverage pairwise preference data for training and have demonstrated promising results. However, these methods face limitations, as they cannot fully exploit the rich information inherent in preference data, such as intermediate quality levels between chosen and rejected samples. Motivated by this insight, we propose **S**ynthetic **P**reference **I**nterpolation **A**lignment (**SPIA**), a novel alignment algorithm that introduces interpolated synthetic preferences to better capture the nuances between samples of different quality levels. By constructing synthetic preference data that reflects intermediate quality with pair-wise preference data, our method effectively bridges the gap between binary pairwise comparisons and richer quality representation. We validate the effectiveness of SPIA through extensive experiments on both annotated preference data and self-play preference data benchmarks, achieving substantial improvements in alignment performance. Our results demonstrate that SPIA not only outperforms existing methods but also provides valuable insights into harnessing preference data for stronger human-aligned LLMs.

1 Introduction

Over the past two years, large language models (LLMs) have demonstrated remarkable advancements across diverse NLP tasks, including mathematical problem-solving, summarization, reading comprehension, and open-ended question answering. Despite these successes, aligning the behavior of LLMs with human expectations remains a critical challenge. Alignment involves ensuring factual correctness, minimizing harmful biases, and enhancing capabilities such as mathematical reasoning. To address these issues, researchers have proposed various alignment training methods aimed at improving LLM reliability and usability.

Among these methods, Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) demonstrated strong alignment performance. However, it requires training a reward model from human-annotated preference data and subsequently fine-tuning the language model with Proximal Policy Optimization (PPO) (Schulman et al., 2017) to maximize the reward, making RLHF less accessible for many practitioners due to the complexity of training.

To simplify alignment training, recent research has focused on developing direct and efficient alternatives to RLHF. Among these, Direct Preference Optimization (DPO)(Rafailov et al., 2024) has gained significant attention. DPO eliminates the need for explicit reward models or reinforcement learning by directly fine-tuning the LLM on human preference pairs. Despite its simplicity, DPO retains strong alignment performance and has inspired subsequent studies aimed at improving its optimization framework. For example, Identity Preference Optimization (IPO) (Azar et al., 2023) addresses overfitting issues in DPO through a novel identity-based loss function. Similarly, ORPO (Hong et al., 2024) and SimPO (Meng et al., 2024) further simplify the DPO workflow by removing dependence on reference models.

While these improvements primarily focus on optimization techniques, relatively little attention has been given to the training data itself. Given the high cost of data annotation, maximizing the utilization of existing preference data is critical for advancing alignment methods. However, existing pairwise preference data, which only captures binary relationships between "chosen" and "rejected" samples, leaves much of its potential information unexploited. In particular, pairwise data often neglects the nuanced quality continuum that may ex-

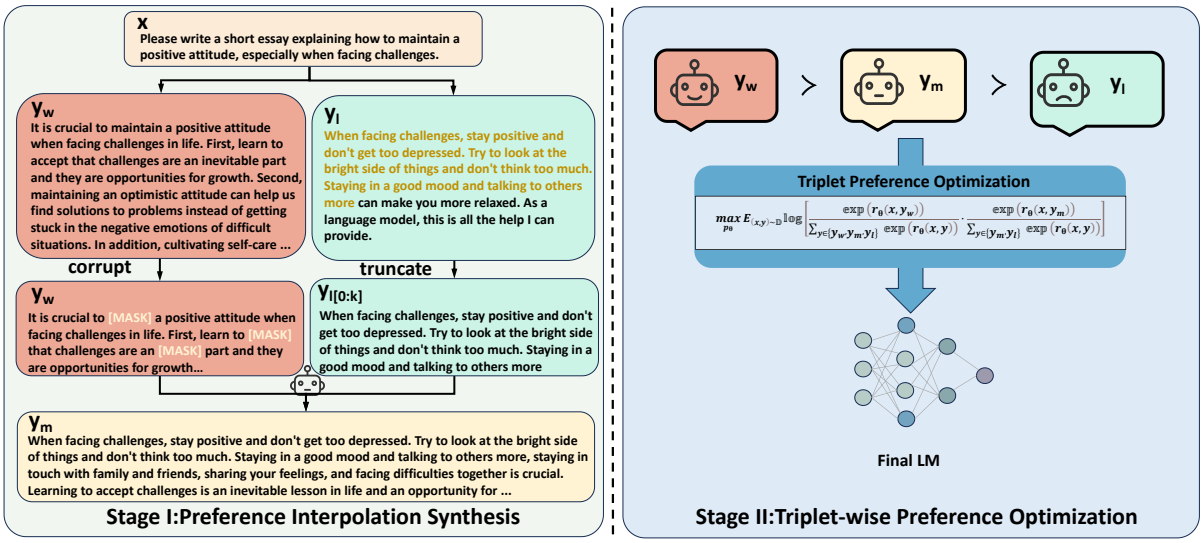


Figure 1: Model Architecture.

Figure 2: **SPIA** consists with two stages: Preference Interpolation Synthesis and Triplet-wise Preference Optimization. In the first stage, we truncate the rejected response $\mathbf{y}_l$ and then the LM generates the continual part of truncated $\mathbf{y}_l$,prompted with corresponding instruction $\mathbf{x}$ and corrupted chosen response $\widetilde{\mathbf{y}}_\mathbf{w}$. In the second stage, denoting the synthetic response as $\mathbf{y}_m$, we train the LM p_θ with preference triplet $(\mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l)$. using a triplet-wise preference loss function.

ist between two samples. Recent work, such as LiPO (Liu et al., 2024), has shown that training on list-wise preferences can outperform pairwise preference training. However, collecting list-wise preference data, whether through manual annotation or GPT-4-based methods (OpenAI et al., 2024), remains resource-intensive. Thus, a cost-effective strategy to obtain list-wise preferences from existing pairwise data is an open research challenge.

In this paper, we propose a novel approach called Synthetic Preference Interpolation Alignment (**SPIA**) to address these limitations. Our method introduces a data synthesis phase that generates interpolated preference data from pairwise preference data, capturing intermediate quality levels between chosen and rejected samples. SPIA then employs a triplet preference training paradigm, leveraging these synthesized preferences to improve alignment performance. We demonstrate the effectiveness of our proposed approach through evaluations on widely used LLM benchmarks, downstream tasks, and reward distributions. Specifically, we fine-tune Phi-3.5-mini-instruct (Abdin et al., 2024) on Ultrachat200k (Ding et al., 2023) and Ultrafeedback (Cui et al., 2024) datasets, comparing the performance of our models against other preference training methods. Furthermore, we perform an ablation study using alternative data synthesis techniques and conduct a detailed evaluation of our proposed synthesis method. Our contribution can be summarized as follows:

- We point out that pair-wise preference data has not been fully utilized in alignment training. Therefore, we propose a novel method to synthesize preference interpolation data.
- We further demonstrate that when the quality of the synthesized data is adequate, optimizing LLMs with triplet preferences can achieve better performance.
- Our proposed novel training pipeline (**SPIA**) can improve model performance more consistently on both self-play and annotation-available alignment setting compared to other methods, without incurring additional annotation costs.

2 Related work

2.1 Data Synthesis by LLM

Data synthesis plays a pivotal role in the field of machine learning. With the advancement of large language models (LLMs), the utilization of LLMs for data synthesis has become increasingly promising. Numerous researchers employ data synthesized with various methods by LLM to fine-tune

LLMs:(Long et al., 2024). In the field of LLM alignment, SPIN (Chen et al., 2024) has demonstrated effective results by utilizing language models that have been supervised fine-tuned to generate responses that serve as rejected samples for preference training. However, using LLMs for data synthesis is not without its challenges. One of the key issues lies in ensuring the quality and diversity of the generated data. While LLMs can produce coherent and contextually relevant text, they may also introduce biases, repetition, or fail to generate sufficiently diverse samples.

2.2 LLM Self-refinement

Recent studies have indicated that large language models (LLMs) possess the potential for self-improvement in their responses. The process of self-refine (Madaan et al., 2023) involves using the LLM to generate feedback that can be used to enhance the results it produced previously.

Self-refinement has the potential to reduce the reliance on external supervision, thus improving model performance with minimal human intervention. However, some researchers, for example,(Huang et al., 2024) have suggested that LLMS struggle to self-correct their responses without external feedback, which means language models with insufficient capabilities may generate worse answers when attempting self-refinement.

2.3 RLHF

Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) is a method designed to align large language models with human preferences and values. The traditional RLHF applies the Bradley-Terry model and typically involves three key stages: supervised fine-tuning, reward model training, and RL-based optimization. In the RL stage, Proximal Policy Optimization (PPO) (Schulman et al., 2017) is a commonly employed algorithm to train the LLM to maximize the score of the reward model for the generated response.

3 Preliminaries

We consider a Large Language Model (LLM) parameterized by θ and denoted as p_θ, which accepts a sequence $\mathbf{x} = [x_1, \ldots, x_n]$, commonly termed as the prompt, and then generate a corresponding response $\mathbf{y} = [y_1, \ldots, y_m]$. Hence, the response $\mathbf{y}$ is construed as a sample drawn from the conditional probability distribution $p_\theta(\cdot|\mathbf{x})$. The conditional probability distribution $p_\theta(\mathbf{y}|\mathbf{x})$ can be decomposed as follows:

$$p_\theta(\mathbf{y}|\mathbf{x}) = \prod_{j=1}^{m} p_\theta(y_j|\mathbf{x}, \mathbf{y}_{<j}),$$

Subsequently, we review supervised fine-tuning (SFT). SFT is the primary training method to adapt a pre-trained LLM for downstream tasks, utilizing a relatively smaller dataset of labeled examples compared to the data used in pre-training stage. In this paper, we focus on the task of instruction-tuning where the prompt-answer pairs denoted as $(\mathbf{x}, \mathbf{y})$, are drawn from a specified SFT dataset $\mathcal{D}$. Thus the training objective of SFT under the instruction tuning setting can be formulated as:

$$\max_{p_\theta} \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\log p_\theta(\mathbf{y} \mid \mathbf{x})\Big]$$

Then we review the setting and method of Direct Preference Optimization (DPO) which optimizes a LLM with pair-wise preference data. Consider a tuple $(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l)$, where $\mathbf{x}$ is prompt while $\mathbf{y}_w$ and $\mathbf{y}_l$ are chosen response and rejcted response respectively. Formally, this preference can be denoted as $\mathbf{y}_w \succ \mathbf{y}_l \mid \mathbf{x}$. These preferences are assumed to be generated by an underlying latent reward model $\mathbf{r}^*(\mathbf{x}, \mathbf{y})$. The Bradley-Terry model (Bradley and Terry, 1952) specifically defines the human preference distribution p^* as follows:

$$p^*(y_1 \succ y_2 \mid x) = \frac{\exp(r^*(x,y_1))}{\exp(r^*(x,y_1))+\exp(r^*(x,y_2))}.$$

Given a preference dataset $\mathcal{D}$ sampled from p^* which contains N preference pairs $(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l)$, DPO considers the same RL optimization goals as other human preference alignment algorithms (such as RLHF):

$$\max_{p_\theta} \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(y|x)}\left[r_\phi(x,y)\right] - \beta\, \mathrm{D}_{\mathrm{KL}}\left[p_\theta(y \mid x) \,\|\, p_{\mathrm{ref}}(y \mid x)\right]$$

where β is a parameter controlling the deviation from the reference model p_{ref}. Instead of training a reward model, DPO reparameterizes the reward function and optimize the RL objective by:

$$\max_{p_\theta} \mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}}\left[\log\sigma\left(\beta\log\frac{p_\theta(\mathbf{y}_w|\mathbf{x})}{p_{\theta_{ref}}(\mathbf{y}_w|\mathbf{x})} - \beta\log\frac{p_\theta(\mathbf{y}_l|\mathbf{x})}{p_{\theta_{ref}}(\mathbf{y}_l|\mathbf{x})}\right)\right]$$

4 Method

4.1 Overview

Our proposed **SPIA** begins with a pair-wise preference dataset $\mathcal{D}$ which contains preference pair

$$p^*(\mathbf{y}_w \succ \mathbf{y}_m \succ \mathbf{y}_l \mid \mathbf{x}) = \frac{\exp\left(r^*(\mathbf{x}, \mathbf{y}_w)\right)}{\sum_{\mathbf{y} \in \{\mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l\}} \exp\left(r^*(\mathbf{x}, \mathbf{y})\right)} \cdot \frac{\exp\left(r^*(\mathbf{x}, \mathbf{y}_m)\right)}{\sum_{\mathbf{y} \in \{\mathbf{y}_m, \mathbf{y}_l\}} \exp\left(r^*(\mathbf{x}, \mathbf{y})\right)} \tag{1}$$

$$\max_{p_\theta} \mathbb{E}_{(x,y) \sim \mathcal{D}_{syn}, y' \sim p_\theta(\cdot|x)} \Big[\log p^*(\mathbf{y}_w \succ \mathbf{y}_m \succ \mathbf{y}_l \mid \mathbf{x}) \Big] \tag{2}$$

$$\max_{p_\theta} \mathbb{E}_{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l) \sim \mathcal{D}_{syn}} \log \left[\frac{\exp\left(r_\theta(\mathbf{x}, \mathbf{y}_w)\right)}{\sum_{\mathbf{y} \in \{\mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l\}} \exp\left(r_\theta(\mathbf{x}, \mathbf{y})\right)} \cdot \frac{\exp\left(r_\theta(\mathbf{x}, \mathbf{y}_m)\right)}{\sum_{\mathbf{y} \in \{\mathbf{y}_m, \mathbf{y}_l\}} \exp\left(r_\theta(\mathbf{x}, \mathbf{y})\right)} \right] \tag{3}$$

$(\mathbf{y}_w, \mathbf{y}_l)$ and a language model p_θ trained with SFT. **SPIA** consists with two stages: Preference Interpolation Synthesis and Triplet-wise Preference Optimization. In the first stage, we truncate the rejected response $\mathbf{y}_l$ and then the LM generates the continual part of truncated $\mathbf{y}_l$,prompted with corresponding instruction $\mathbf{x}$ and corrupted chosen response $\widetilde{\mathbf{y}}_\mathbf{w}$. In the second stage, denoting the synthetic response as $\mathbf{y}_m$, we train the LM p_θ with preference triplet $(\mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l)$. Next, we will elaborate on the specific process of each of the two stages.

4.2 Preference Interpolation Synthesis (PIS)

To synthesize a sample $\mathbf{y}_m$ that satisfying preference ranking $\mathbf{y}_w \succ \mathbf{y}_m \succ \mathbf{y}_l$,which means that the response have the quality between chosen and rejected sample, we proposed a novel LM-based data synthesis approach. Specifically, we truncate $\mathbf{y}_l$, retaining only the first k tokens. Then, using the corresponding $\mathbf{x}$ corrupted as a prompt, we prompt the LLM to continue writing based on the truncated , thereby generating a preference-interpolated sample that meets the required criteria. The whole process can be formulated as:

$$\mathbf{y}_m = \mathbf{y}_l[0:k] \oplus \mathbf{y}', \text{ where } \mathbf{y}' \sim p_\theta(\cdot|\mathbf{x}, \widetilde{\mathbf{y}}_\mathbf{w}, \mathbf{y}_l[0:k])$$

In this formula, we choose different k for samples of various length. Formally, k is linearly determined by the token length of $\mathbf{y}_l$, which is $\alpha|\mathbf{y}_l|$. To understand this synthesis process, we illustrates the motivation of three main concepts of this method. Firstly, we use a part of $\mathbf{y}_l$ as starter sequence of $\mathbf{y}_m$ to ensure that the quality of the generated sample does not exceed that of the chosen response $\mathbf{y}_w$. This is because, during the generation process of a language model, the initial tokens often set the general direction for the entire output. Secondly, we use the chosen sample as a reference to guide the model, in order to generate a better response than $\mathbf{y}_l$ with the information of $\mathbf{y}_w$. Thirdly, the chosen sample $\mathbf{y}_w$ is corrupted to $\widetilde{\mathbf{y}}_\mathbf{w}$, This is to prevent $\mathbf{y}_m$ from replicating $\mathbf{y}_w$ word-by-word during generation, thereby enhancing sample diversity and facilitating the language model's state exploration in training process. In the Evaluation Section, we conduct analysis on the synthetic $\mathbf{y}_m$.

4.3 Triplet-wise Preference Optimization

With the synthetic interpolated preference dataset prepared, we use these preference triplets instead of pairs to conduct alignment training on the model. Formally, denoting the triplet dataset we obtain by preference interpolation synthesis process described in last sub-section as $\mathcal{D}_{syn}$, we have a preference triplet $(\mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l)$ for each prompt $\mathbf{x} \in \mathcal{D}$, which can be formulated as $\mathbf{y}_w \succ \mathbf{y}_m \succ \mathbf{y}_l \mid \mathbf{x}$. Under the Plackett-Luce model (Debreu, 1960), we have Eq.1. And our optimization objective is defined as Eq.2. According to the proof of general form DPO proposed by (Rafailov et al., 2024), we can solve this optimization problem by Eq.3, where $r_\theta(\mathbf{x}, \mathbf{y}) = \beta \log \frac{p_\theta(\mathbf{y}|\mathbf{x})}{p_{ref}(\mathbf{y}|\mathbf{x})}$ is the reward implicitly defined by the policy LLM p_θ and reference LLM p_{ref}.

To illustrate the training process more clearly, we also provide pseudocode in Algorithm 1.

5 Experiments

5.1 Experiment Setup

5.1.1 Model

The model we chosen for our experiment is Phi-3.5-mini-instruct (Abdin et al., 2024), which is a lightweight, state-of-the-art open-source model. Phi-3.5-mini-instruct demonstrates performance comparable to or even surpassing larger-scale mod-

Model	AlpacaEval		MT-Bench			LM-Eval-Harness			Average
	Win	LC-Win	Turn-1	Turn-2	Final	TruthfulQA	GSM8K	ARC	
Phi3.5-SFT	50.00	50.00	7.26	5.63	6.44	44.19	74.83	58.53	57.86
Ultrachat-SPIN-Phi									
SPIN	**71.06**	62.7	6.98	6.00	6.49	46.27	72.40	**64.42**	64.27
ORPO	64.24	**63.72**	7.25	**6.15**	6.70	46.39	**75.82**	61.86	64.11
IPO	65.04	62.05	**7.50**	**6.28**	**6.89**	**46.88**	74.37	60.92	**64.45**
SPIA	**74.63**	**68.67**	**7.41**	6.11	**6.76**	**47.37**	**77.10**	**64.33**	**67.39**
Ultrafeedback									
DPO	77.83	**73.33**	**7.57**	6.01	6.79	**46.63**	79.08	**64.16**	68.92
ORPO	64.82	57.83	7.35	5.80	6.58	41.62	70.74	55.55	61.03
IPO	**78.94**	70.87	7.73	**6.63**	**7.16**	45.17	78.24	60.23	**69.24**
SimPO	71.74	61.34	7.23	6.22	6.73	45.90	**79.91**	60.67	65.33
SPIA	**79.63**	**74.43**	**7.91**	**6.28**	**7.09**	**46.63**	**78.92**	**62.46**	**70.20**

Table 1: Performance of different methods on three LLM benchmarks. For AlpacaEval, we use GPT-4o as judge and Phi3.5-SFT as reference LLM. So the win-rate of Phi3.5-SFT on itself is set to 50.00. For MT-Bench, we also use GPT-4o as judge. Our model has the best or second best score in each indicator, and has the best overall performance

Algorithm 1 SPIA Pipeline

Require: Preference dataset $\mathcal{D} = \{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l)\}$, LLM p_θ, Preference interpolation dataset $\mathcal{D}_{syn} = \{\}$

for $(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_l)$ in $\mathcal{D}$ **do**

Truncate $\mathbf{y}_l$ to attain $\mathbf{y}_l[0:k]$

Corrupt $\mathbf{y}_w$ to attain $\widetilde{\mathbf{y}}_\mathbf{w}$

Initialize $\mathbf{y}_m[0:k] = \mathbf{y}_l[0:k]$

Generate $\mathbf{y}_m[k:] \sim p_\theta(\cdot|\mathbf{x}, \widetilde{\mathbf{y}}_\mathbf{w}, \mathbf{y}_l[0:k])$

$\mathbf{y}_m = mathbfy_m[0:k] \oplus mathbfy_m[k:]$

$\mathcal{D}_{syn}$ += $(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l)$

end for

Update $\theta = \arg\min_\theta \sum_{\mathcal{D}_{syn}} \mathcal{L}_{SPIA}(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l)$

els, such as Mistral-7B (Jiang et al., 2023) and Llama-3.1-8B (Grattafiori et al., 2024), across a wide range of evaluation tasks.

5.1.2 Dataset

The dataset used for supervised-finetune is Ultrachat200k. It comprises approximately 200,000 high-quality, multi-turn dialogues generated by ChatGPT, covering a diverse array of topics. For preference training we use two dataset. Ultrafeedback and Ultrachat-SPIN-Phi. Ultrafeedback is a widely used preference dataset for LLM alignment, which contains 64k preference pair annotated by GPT-4. We use a subset of 36k for our training. Ultrachat-SPIN-Phi is a synthetic self-play dataset created by us using the SPIN (Chen et al., 2024) methodology: SFT responses are designated as the chosen responses, while model-generated responses (by Phi-3.5-SFT) serve as the rejected responses. A total of 36k data samples were collected. We chosen these two datasets thus we can demostrate the efficiency of our method on both annotation-available and annotation-free setting in LLM alignment.

5.1.3 Competing Methods

In order to analyze the effectiveness of our method, we choose several widely used approaches in the field of alignment including: **DPO**, **SPIN**, **SimPO**, **ORPO** and **IPO**.

5.1.4 Experiment Details

To start with, we finetune Phi-3.5-mini-instruct on the Ultrachat dataset to obtain a SFT version (Phi-3.5-SFT) in our alignment experiment. For SFT stage, we only use 36k samples to prevent from model forgetting. After this, we use Phi-3.5-SFT to synthesize response by Preference Interpolation Synthesis mentioned in last section to. We generate 36k preference interpolation data for Ultrachat-SPIN-Phi and Ultrafeedback respectively. Then we train our model on the synthetic data with triplet-wise preference loss while train other baseline model on Ultrachat-SPIN-Phi or Ultrafeedback. Besides, we found that SimPO is not suitable for self-play setting as the model collapse during training on Ultrachat-SPIN-Phi. So we do not include SimPO in the Ultrachat-SPIN-Phi part of our reported results in Table 1

5.1.5 Evaluation Metric

We evaluate the effectiveness of our approach from multiple dimensions, covering general conversational ability, factual accuracy, and reasoning skills. The following is a brief introduction to the evaluation benchmarks we use.

- **AlpacaEval** In AlpacaEval (Dubois et al., 2024) benchmark, a model generates responses to 805 questions covering a wide range of topics. The responses are evaluated by LLM, and the final metric is determined by the pairwise win rate and length-controlled win rate over a baseline model.
- **MT-Bench** MT-Bench (Zheng et al., 2023) is a multi-turn evaluation benchmark comprising 160 questions across eight knowledge domains. Each response is rated by a LLM annotator on a scale from 1 to 10, with the final score being the average of the two responses.
- **LM-Evaluation-Harness** LM-Evaluation-Harness (Gao et al., 2024) provides a unified framework to test generative language models on a large number of different evaluation tasks. We chose three widely used benchmarks: TruthfulQA (Lin et al., 2022) (focusing on truthfulness), GSM8k (Cobbe et al., 2021) (focusing on mathematical reasoning skills) and ARC (Clark et al., 2018) (focusing on general scientific reasoning ability).

5.2 Results

Table 1 presents the performance of all competing methods on our selected benchmarks. Additionally, We calculated the average score to compare the overall performance of each method. The average score is calculated by:

$$\frac{\frac{(\textit{LC-Win+Win})}{2} + 10 * \textit{Final} + \frac{(\textit{TruthfulQA+GSM8k+ARC})}{3}}{3}$$

We observe that all training methods demonstrate significant improvements over the SFT model across various benchmarks. Among them, **SPIA** achieves the highest average score while attaining either the best or second-best performance across various benchmark metrics, indicating the strong effectiveness of our model. As for other reported methods, IPO and DPO are most competitive. What's more, we found that our method exhibits more advantage over other approaches in the self-play setting. Specifically, with data annotated by human/GPT-4 (Ultrafeedback), our method outperforms DPO by 1.82. In the self-play setting, it surpasses SPIN (which employs the same loss function as DPO) by 3.12.

Model	**Reward**$_{avg}$
Phi-3.5-SFT	-4.28
Ultrachat-SPIN-Phi	
SPIN	-3.63
SPIA	-2.70
Ultrafeedback	
DPO	-2.00
SPIA	-1.89

Table 2: Comparison of Average Reward.

6 Further Evaluation

In this section, we conduct a more in-depth analysis of our approach, including a comparison of reward distributions, an evaluation of synthesized data and ablation studies focusing on training data.

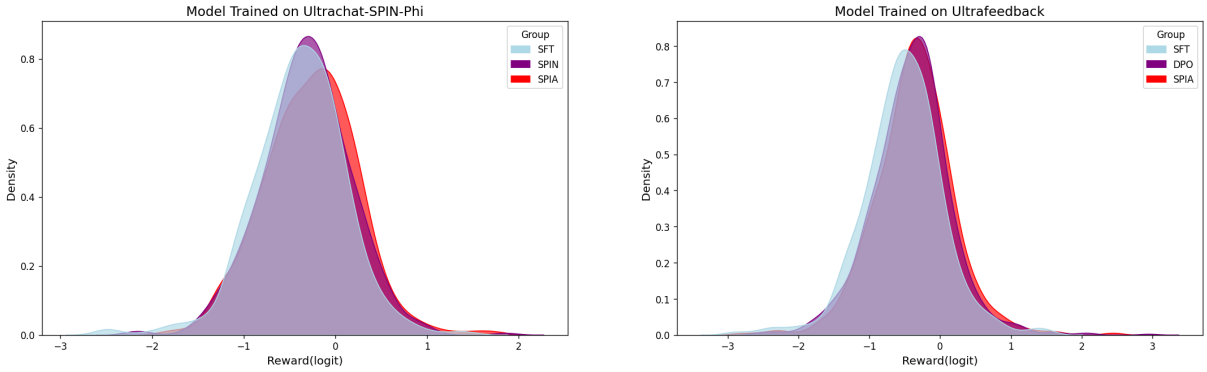
6.1 Reward Distribution

In addition to evaluating on benchmarks, we visualize and compare the reward distributions of responses generated on the test set of the Ultrachat dataset. The reward is generated by Skywork-Reward-Llama-3.1-8B-v0.2, which is a widely used scoring-based reward model and we select 1000 samples for both settings. The average reward score is reported in Table 2. For clarity and simplicity, we focus on visualizing the reward logit of SFT, SPIN(DPO), and SPIA. In the Figure 4, we can find that, in both settings, our model achieves a higher average reward. Specifically, in the self-play alignment setting (Ultrachat-SPIN-Phi), the reward distribution of our model exhibits a significantly greater positive shift compared to SPIN. In the annotation-available setting (Ultrafeedback), while the density peak of our model is close to that of DPO, our model demonstrates a lower density in the low-reward region and a higher density in the high-reward region.

6.2 Ablations

6.2.1 Ablation on Training Data

To validate the effectiveness of data synthesis method, we additionally selected two alternative



(a) Ultrachat-SPIN-Phi

(b) Ultrafeedback

Figure 3: Reward distribution comparison. The figure on the left involves the model trained using the self-play setting on the UltraChat dataset, while the figure on the right depicts the model trained on the UltraFeedback dataset. Our model achieves higher average rewards. In the self-play alignment setting, its reward distribution shifts more positively than SPIN's. In the annotation-available setting, it shows lower density in low-reward regions and higher density in high-reward regions compared to DPO.

Data Category	**Score**$_{avg}$	**Qualification**$_{\%}$	**Edit Distance**
Ultrachat-SPIN-Phi			
Win	8.35	/	/
Lose	7.76	/	/
Middle$_{\text{self-refine}}$	7.74	0.18	129.56
Middle$_{\text{paraphrase}}$	8.10	0.25	153.73
Middle$_{\textbf{PIS}}$	7.95	0.45	146.39
Ultrafeedback			
Win	7.88	/	/
Lose	6.49	/	/
Middle$_{\text{self-refine}}$	6.51	0.20	121.34
Middle$_{\text{paraphrase}}$	7.62	0.23	140.72
Middle$_{\textbf{PIS}}$	6.77	0.42	130.01

Table 3: Evaluation of Synthesized Data. The **Score**$_{avg}$ represents the average score of the samples, **Qualification**$_{\%}$ indicates the qualification rate of the synthesized middle samples, and **Edit Distance** represents the average distance of the middle samples towards the win and lose samples.

data synthesis methods for self-play setting: paraphrasing and self-refinement. For paraphrasing, we prompt the model with $\mathbf{x}$ and $\mathbf{y}_w$ and let the model to generate a paraphrase. For self-refinement, we prompt the model with $\mathbf{x}$ and $\mathbf{y}_l$ and let the model to refine it's original response. In addition, since generating rejected responses at a relatively low cost in self-play setting, to demonstrate the superiority of our method, we also include a comparative setting where SPIN is trained with double the amount of training data by sampling two responses for one prompt. Keeping all other hyperparameters consistent, we conducted experiments using these training data and evaluate models on AlpacaEval and MT-Bench. From the results shown in Table 4, we can see that our synthesis method produced the best results: the other three settings (**Exp.1**, **Exp.2** and **Exp.3**) have noticeable declines on MT-Bench and AlpacaEval, and the model performance is generally inferior to our method (**Exp.6**).

6.2.2 Ablation on Training Loss

We also conducted ablation experiments on the loss function, where each sample from the synthesized interpolation dataset was split into two:

$$(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_m, \mathbf{y}_l) \rightarrow \{(\mathbf{x}, \mathbf{y}_w, \mathbf{y}_m), (\mathbf{x}, \mathbf{y}_m, \mathbf{y}_l)\}$$

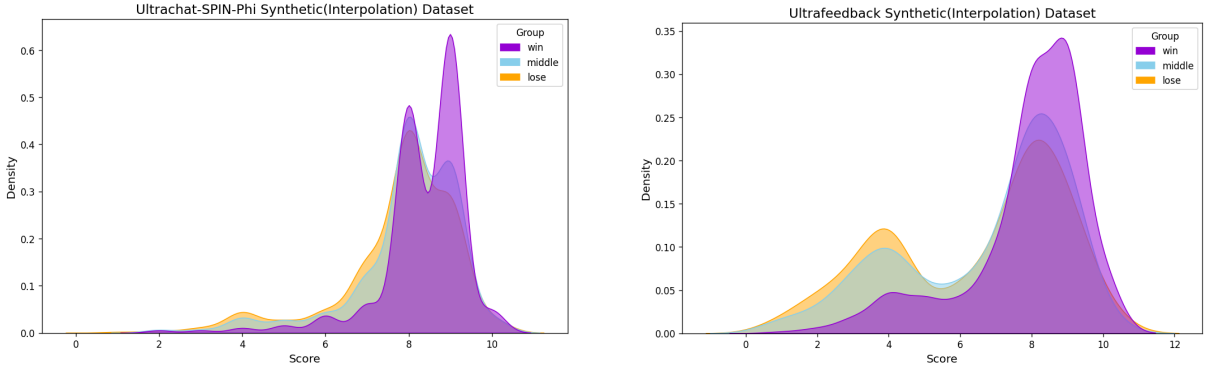
It's evident that $\{(\mathbf{y}_w \succ \mathbf{y}_m), (\mathbf{y}_m \succ \mathbf{y}_l)\}$ is a canonical cover of the partial order $(\mathbf{y}_w \succ \mathbf{y}_m \succ \mathbf{y}_l)$. So the training data generated by splitting is equivalent in preference with original dataset. Then training was performed using pair-wise loss (i.e., DPO loss). As shown in Table 4, we can observe that under consistent training data (**Exp.4** vs **Exp.5**, **Exp.6** vs **Exp.7**), optimizing with the triplet loss yields better performance. We hypothesize that this is due to the triplet loss directly incorporating three preference relationships $(\mathbf{y}_w \succ \mathbf{y}_m, \mathbf{y}_m \succ \mathbf{y}_l, \mathbf{y}_w \succ \mathbf{y}_l)$, which fully leverages the data. Furthermore, according to the alignment tax theory (Lin et al., 2024), the model's performance may be impacted as the number of training steps increases due to larger training set.

6.3 Evaluation of Synthesized Data

We analyzed the synthesized data from both distribution-level and instance-level perspectives. Specifically, we considered both the self-play and annotation-available settings. And three methods (paraphrase, self-refine, **PIS**) are considered for

Experiment	Training Data	Loss Function	AlpacaEval$_{win}$	MT-Bench$_{final}$
Exp.1	*Ultrachat-SPIN-Phi$_{paraphrase}$*	SPIA	70.68	6.29
Exp.2	*Ultrachat-SPIN-Phi$_{self\text{-}refine}$*	SPIA	72.17	6.60
Exp.3	*Ultrachat-SPIN-Phi$_{double\text{-}response}$*	SPIN	70.06	6.03
Exp.4	*Ultrachat-SPIN-Phi$_{split}$*	SPIN	71.93	6.50
Exp.5	*Ultrafeedback$_{split}$*	DPO	78.26	6.86
Exp.6	*Ultrachat-SPIN-Phi$_{syn}$*	SPIA	74.63	6.76
Exp.7	*Ultrafeedback$_{syn}$*	SPIA	79.63	7.09

Table 4: Ablation Experiments. **Exp.1**, **Exp.2** and **Exp.3** are experiments conducted with other preference synthesis methods. **Exp.4** and **Exp.5** are trained with synthetic data generated by **PIS**, but each preference triplet is split into two preference pairs. **Exp.6** and **Exp.7** are models trained with intact **SPIA** pipeline.



(a) Ultrachat-SPIN-Phi

(b) Ultrafeedback

Figure 4: Score distribution comparison of different data category. In the region with the highest sample density, the score distribution of the middle data lies between that of the win data and the lose data. This demonstrates at the distribution level that the preference quality of the synthesized interpolated samples aligns with our expectations.

each dataset. For each setting, 500 samples were randomly selected from the synthesized dataset. Using a predefined scoring rule, GPT-4o was employed to evaluate each prompt-response pair. We use GPT-4 because we noticed that if two samples are similar, Skywork-Reward-Llama-3.1-8B-v0.2 is not capable enough to discriminate effectively. First, we plotted the density distribution of the scores and subsequently calculated the mean score for each distribution. Then, to validate at the instance level whether the interpolated samples exhibit a preference quality that lies between the chosen and rejected ones, given a significant preference difference between $\mathbf{y}_w$ and $\mathbf{y}_l$, we select samples from the dataset where $\text{score}(\mathbf{y}_w) > \text{score}(\mathbf{y}_l) + \epsilon$. We then calculate the ratio of samples that satisfy $\text{score}(\mathbf{y}_w) > \text{score}(\mathbf{y}_m) > \text{score}(\mathbf{y}_l)$ under margin constant $\epsilon = 1$, which is referred to as the **Qualification**$_{\%}$ in the table 3. In addition, we used the Levenshtein algorithm to compute the average edit distance between the synthetic samples and both the chosen and rejected samples to characterize the similarity between the synthetic samples and the original training samples. Under the condition of maintaining data quality, lower similarity is more advantageous for the model's exploration in the sequence space.

7 Conclusion

In this paper we point out that existing alignment methods do not fully leverage pairwise preference data. We propose a preference interpolation data synthesis method and optimize the model using a triplet preference loss. Based on extensive experimental results, the preference interpolation data synthesis method demonstrates good utility, and training with triplet preference loss yields better performance.

8 Limitations

The preference interpolation synthesis method we proposed does lead to an improvement in training performance; however, the quality of the interpolated data still leaves room for further enhancement according to the our evaluation. Additionally, due

to computational resource limitations, we were unable to train on larger models, such as the 13B and 70B models.

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, Alon Benhaim, Misha Bilenko, Johan Bjorck, Sébastien Bubeck, Martin Cai, Qin Cai, Vishrav Chaudhary, Dong Chen, Dongdong Chen, Weizhu Chen, Yen-Chun Chen, Yi-Ling Chen, Hao Cheng, Parul Chopra, Xiyang Dai, Matthew Dixon, Ronen Eldan, Victor Fragoso, Jianfeng Gao, Mei Gao, Min Gao, Amit Garg, Allie Del Giorno, Abhishek Goswami, Suriya Gunasekar, Emman Haider, Junheng Hao, Russell J. Hewett, Wenxiang Hu, Jamie Huynh, Dan Iter, Sam Ade Jacobs, Mojan Javaheripi, Xin Jin, Nikos Karampatziakis, Piero Kauffmann, Mahoud Khademi, Dongwoo Kim, Young Jin Kim, Lev Kurilenko, James R. Lee, Yin Tat Lee, Yuanzhi Li, Yunsheng Li, Chen Liang, Lars Liden, Xihui Lin, Zeqi Lin, Ce Liu, Liyuan Liu, Mengchen Liu, Weishung Liu, Xiaodong Liu, Chong Luo, Piyush Madan, Ali Mahmoudzadeh, David Majercak, Matt Mazzola, Caio César Teodoro Mendes, Arindam Mitra, Hardik Modi, Anh Nguyen, Brandon Norick, Barun Patra, Daniel Perez-Becker, Thomas Portet, Reid Pryzant, Heyang Qin, Marko Radmilac, Liliang Ren, Gustavo de Rosa, Corby Rosset, Sambudha Roy, Olatunji Ruwase, Olli Saarikivi, Amin Saied, Adil Salim, Michael Santacroce, Shital Shah, Ning Shang, Hiteshi Sharma, Yelong Shen, Swadheen Shukla, Xia Song, Masahiro Tanaka, Andrea Tupini, Praneetha Vaddamanu, Chunyu Wang, Guanhua Wang, Lijuan Wang, Shuohang Wang, Xin Wang, Yu Wang, Rachel Ward, Wen Wen, Philipp Witte, Haiping Wu, Xiaoxia Wu, Michael Wyatt, Bin Xiao, Can Xu, Jiahang Xu, Weijian Xu, Jilong Xue, Sonali Yadav, Fan Yang, Jianwei Yang, Yifan Yang, Ziyi Yang, Donghan Yu, Lu Yuan, Chenruidong Zhang, Cyril Zhang, Jianwen Zhang, Li Lyna Zhang, Yi Zhang, Yue Zhang, Yunan Zhang, and Xiren Zhou. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *Preprint*, arXiv:2404.14219.

Mohammad Gheshlaghi Azar, Mark Rowland, Bilal Piot, Daniel Guo, Daniele Calandriello, Michal Valko, and Rémi Munos. 2023. A general theoretical paradigm to understand learning from human preferences. *Preprint*, arXiv:2310.12036.

Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345.

Zixiang Chen, Yihe Deng, Huizhuo Yuan, Kaixuan Ji, and Quanquan Gu. 2024. Self-play fine-tuning converts weak language models to strong language models. *Preprint*, arXiv:2401.01335.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *Preprint*, arXiv:1803.05457.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *Preprint*, arXiv:2110.14168.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Bingxiang He, Wei Zhu, Yuan Ni, Guotong Xie, Ruobing Xie, Yankai Lin, Zhiyuan Liu, and Maosong Sun. 2024. Ultrafeedback: Boosting language models with scaled ai feedback. *Preprint*, arXiv:2310.01377.

Gerard Debreu. 1960. Individual choice behavior: A theoretical analysis.

Ning Ding, Yulin Chen, Bokai Xu, Yujia Qin, Zhi Zheng, Shengding Hu, Zhiyuan Liu, Maosong Sun, and Bowen Zhou. 2023. Enhancing chat language models by scaling high-quality instructional conversations. *Preprint*, arXiv:2305.14233.

Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B. Hashimoto. 2024. Length-controlled alpacaeval: A simple way to debias automatic evaluators. *Preprint*, arXiv:2404.04475.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. 2024. A framework for few-shot language model evaluation.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurelien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Roziere, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, Danny Wyatt, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Francisco Guzmán, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Govind Thattai, Graeme Nail, Gregoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen,

Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel Kloumann, Ishan Misra, Ivan Evtimov, Jack Zhang, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Karthik Prasad, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, Khalid El-Arini, Krithika Iyer, Kshitiz Malik, Kuenley Chiu, Kunal Bhalla, Kushal Lakhotia, Lauren Rantala-Yeary, Laurens van der Maaten, Lawrence Chen, Liang Tan, Liz Jenkins, Louis Martin, Lovish Madaan, Lubo Malo, Lukas Blecher, Lukas Landzaat, Luke de Oliveira, Madeline Muzzi, Mahesh Pasupuleti, Mannat Singh, Manohar Paluri, Marcin Kardas, Maria Tsimpoukelli, Mathew Oldham, Mathieu Rita, Maya Pavlova, Melanie Kambadur, Mike Lewis, Min Si, Mitesh Kumar Singh, Mona Hassan, Naman Goyal, Narjes Torabi, Nikolay Bashlykov, Nikolay Bogoychev, Niladri Chatterji, Ning Zhang, Olivier Duchenne, Onur Çelebi, Patrick Alrassy, Pengchuan Zhang, Pengwei Li, Petar Vasic, Peter Weng, Prajjwal Bhargava, Pratik Dubal, Praveen Krishnan, Punit Singh Koura, Puxin Xu, Qing He, Qingxiao Dong, Ragavan Srinivasan, Raj Ganapathy, Ramon Calderer, Ricardo Silveira Cabral, Robert Stojnic, Roberta Raileanu, Rohan Maheswari, Rohit Girdhar, Rohit Patel, Romain Sauvestre, Ronnie Polidoro, Roshan Sumbaly, Ross Taylor, Ruan Silva, Rui Hou, Rui Wang, Saghar Hosseini, Sahana Chennabasappa, Sanjay Singh, Sean Bell, Seohyun Sonia Kim, Sergey Edunov, Shaoliang Nie, Sharan Narang, Sharath Raparthy, Sheng Shen, Shengye Wan, Shruti Bhosale, Shun Zhang, Simon Vandenhende, Soumya Batra, Spencer Whitman, Sten Sootla, Stephane Collot, Suchin Gururangan, Sydney Borodinsky, Tamar Herman, Tara Fowler, Tarek Sheasha, Thomas Georgiou, Thomas Scialom, Tobias Speckbacher, Todor Mihaylov, Tong Xiao, Ujjwal Karn, Vedanuj Goswami, Vibhor Gupta, Vignesh Ramanathan, Viktor Kerkez, Vincent Gonguet, Virginie Do, Vish Vogeti, Vítor Albiero, Vladan Petrovic, Weiwei Chu, Wenhan Xiong, Wenyin Fu, Whitney Meers, Xavier Martinet, Xiaodong Wang, Xiaofang Wang, Xiaoqing Ellen Tan, Xide Xia, Xinfeng Xie, Xuchao Jia, Xuewei Wang, Yaelle Goldschlag, Yashesh Gaur, Yasmine Babaei, Yi Wen, Yiwen Song, Yuchen Zhang, Yue Li, Yuning Mao, Zacharie Delpierre Coudert, Zheng Yan, Zhengxing Chen, Zoe Papakipos, Aaditya Singh, Aayushi Srivastava, Abha Jain, Adam Kelsey, Adam Shajnfeld, Adithya Gangidi, Adolfo Victoria, Ahuva Goldstand, Ajay Menon, Ajay Sharma, Alex Boesenberg, Alexei Baevski, Allie Feinstein, Amanda Kallet, Amit Sangani, Amos Teo, Anam Yunus, Andrei Lupu, Andres Alvarado, Andrew Caples, Andrew Gu, Andrew Ho, Andrew Poulton, Andrew Ryan, Ankit Ramchandani, Annie Dong, Annie Franco, Anuj Goyal, Aparajita Saraf, Arkabandhu Chowdhury, Ashley Gabriel, Ashwin Bharambe, Assaf Eisenman, Azadeh Yazdan, Beau James, Ben Maurer, Benjamin Leonhardi, Bernie Huang, Beth Loyd, Beto De Paola, Bhargavi Paranjape, Bing Liu, Bo Wu, Boyu Ni, Braden Hancock, Bram Wasti, Brandon Spence, Brani Stojkovic, Brian Gamido, Britt Montalvo, Carl Parker, Carly Burton, Catalina Mejia, Ce Liu, Changhan Wang, Changkyu Kim, Chao Zhou, Chester Hu, Ching-Hsiang Chu, Chris Cai, Chris Tindal, Christoph Feichtenhofer, Cynthia Gao, Damon Civin, Dana Beaty, Daniel Kreymer, Daniel Li, David Adkins, David Xu, Davide Testuggine, Delia David, Devi Parikh, Diana Liskovich, Didem Foss, Dingkang Wang, Duc Le, Dustin Holland, Edward Dowling, Eissa Jamil, Elaine Montgomery, Eleonora Presani, Emily Hahn, Emily Wood, Eric-Tuan Le, Erik Brinkman, Esteban Arcaute, Evan Dunbar, Evan Smothers, Fei Sun, Felix Kreuk, Feng Tian, Filippos Kokkinos, Firat Ozgenel, Francesco Caggioni, Frank Kanayet, Frank Seide, Gabriela Medina Florez, Gabriella Schwarz, Gada Badeer, Georgia Swee, Gil Halpern, Grant Herman, Grigory Sizov, Guangyi, Zhang, Guna Lakshminarayanan, Hakan Inan, Hamid Shojanazeri, Han Zou, Hannah Wang, Hanwen Zha, Haroun Habeeb, Harrison Rudolph, Helen Suk, Henry Aspegren, Hunter Goldman, Hongyuan Zhan, Ibrahim Damlaj, Igor Molybog, Igor Tufanov, Ilias Leontiadis, Irina-Elena Veliche, Itai Gat, Jake Weissman, James Geboski, James Kohli, Janice Lam, Japhet Asher, Jean-Baptiste Gaya, Jeff Marcus, Jeff Tang, Jennifer Chan, Jenny Zhen, Jeremy Reizenstein, Jeremy Teboul, Jessica Zhong, Jian Jin, Jingyi Yang, Joe Cummings, Jon Carvill, Jon Shepard, Jonathan McPhie, Jonathan Torres, Josh Ginsburg, Junjie Wang, Kai Wu, Kam Hou U, Karan Saxena, Kartikay Khandelwal, Katayoun Zand, Kathy Matosich, Kaushik Veeraraghavan, Kelly Michelena, Keqian Li, Kiran Jagadeesh, Kun Huang, Kunal Chawla, Kyle Huang, Lailin Chen, Lakshya Garg, Lavender A, Leandro Silva, Lee Bell, Lei Zhang, Liangpeng Guo, Licheng Yu, Liron Moshkovich, Luca Wehrstedt, Madian Khabsa, Manav Avalani, Manish Bhatt, Martynas Mankus, Matan Hasson, Matthew Lennie, Matthias Reso, Maxim Groshev, Maxim Naumov, Maya Lathi, Meghan Keneally, Miao Liu, Michael L. Seltzer, Michal Valko, Michelle Restrepo, Mihir Patel, Mik Vyatskov, Mikayel Samvelyan, Mike Clark, Mike Macey, Mike Wang, Miquel Jubert Hermoso, Mo Metanat, Mohammad Rastegari, Munish Bansal, Nandhini Santhanam, Natascha Parks, Natasha White, Navyata Bawa, Nayan Singhal, Nick Egebo, Nicolas Usunier, Nikhil Mehta, Nikolay Pavlovich Laptev, Ning Dong, Norman Cheng, Oleg Chernoguz, Olivia Hart, Omkar Salpekar, Ozlem Kalinli, Parkin Kent, Parth Parekh, Paul Saab, Pavan Balaji, Pedro Rittner, Philip Bontrager, Pierre Roux, Piotr Dollar, Polina Zvyagina, Prashant Ratanchandani, Pritish Yuvraj, Qian Liang, Rachad Alao, Rachel Rodriguez, Rafi Ayub, Raghotham Murthy, Raghu Nayani, Rahul Mitra, Rangaprabhu Parthasarathy, Raymond Li, Rebekkah Hogan, Robin Battey, Rocky Wang, Russ Howes, Ruty Rinott, Sachin Mehta, Sachin Siby, Sai Jayesh Bondu, Samyak Datta, Sara Chugh, Sara Hunt, Sargun Dhillon, Sasha Sidorov, Satadru Pan, Saurabh Mahajan, Saurabh Verma, Seiji Yamamoto, Sharadh Ramaswamy, Shaun Lind-

say, Shaun Lindsay, Sheng Feng, Shenghao Lin, Shengxin Cindy Zha, Shishir Patil, Shiva Shankar, Shuqiang Zhang, Shuqiang Zhang, Sinong Wang, Sneha Agarwal, Soji Sajuyigbe, Soumith Chintala, Stephanie Max, Stephen Chen, Steve Kehoe, Steve Satterfield, Sudarshan Govindaprasad, Sumit Gupta, Summer Deng, Sungmin Cho, Sunny Virk, Suraj Subramanian, Sy Choudhury, Sydney Goldman, Tal Remez, Tamar Glaser, Tamara Best, Thilo Koehler, Thomas Robinson, Tianhe Li, Tianjun Zhang, Tim Matthews, Timothy Chou, Tzook Shaked, Varun Vontimitta, Victoria Ajayi, Victoria Montanez, Vijai Mohan, Vinay Satish Kumar, Vishal Mangla, Vlad Ionescu, Vlad Poenaru, Vlad Tiberiu Mihailescu, Vladimir Ivanov, Wei Li, Wenchen Wang, Wenwen Jiang, Wes Bouaziz, Will Constable, Xiaocheng Tang, Xiaojian Wu, Xiaolan Wang, Xilun Wu, Xinbo Gao, Yaniv Kleinman, Yanjun Chen, Ye Hu, Ye Jia, Ye Qi, Yenda Li, Yilin Zhang, Ying Zhang, Yossi Adi, Youngjin Nam, Yu, Wang, Yu Zhao, Yuchen Hao, Yundi Qian, Yunlu Li, Yuzi He, Zach Rait, Zachary DeVito, Zef Rosnbrick, Zhaoduo Wen, Zhenyu Yang, Zhiwei Zhao, and Zhiyu Ma. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *Preprint*, arXiv:2403.07691.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2024. Large language models cannot self-correct reasoning yet. *Preprint*, arXiv:2310.01798.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Stephanie Lin, Jacob Hilton, and Owain Evans. 2022. Truthfulqa: Measuring how models mimic human falsehoods. *Preprint*, arXiv:2109.07958.

Yong Lin, Hangyu Lin, Wei Xiong, Shizhe Diao, Jianmeng Liu, Jipeng Zhang, Rui Pan, Haoxiang Wang, Wenbin Hu, Hanning Zhang, Hanze Dong, Renjie Pi, Han Zhao, Nan Jiang, Heng Ji, Yuan Yao, and Tong Zhang. 2024. Mitigating the alignment tax of rlhf. *Preprint*, arXiv:2309.06256.

Tianqi Liu, Zhen Qin, Junru Wu, Jiaming Shen, Misha Khalman, Rishabh Joshi, Yao Zhao, Mohammad Saleh, Simon Baumgartner, Jialu Liu, Peter J. Liu, and Xuanhui Wang. 2024. Lipo: Listwise preference optimization through learning-to-rank. *Preprint*, arXiv:2402.01878.

Lin Long, Rui Wang, Ruixuan Xiao, Junbo Zhao, Xiao Ding, Gang Chen, and Haobo Wang. 2024. On llms-driven synthetic data generation, curation, and evaluation: A survey. *Preprint*, arXiv:2406.15126.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. *Preprint*, arXiv:2303.17651.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. Simpo: Simple preference optimization with a reference-free reward. *Preprint*, arXiv:2405.14734.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David

Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Stefano Ermon, Christopher D. Manning, and Chelsea Finn. 2024. Direct preference optimization: Your language model is secretly a reward model. *Preprint*, arXiv:2305.18290.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal policy optimization algorithms. *Preprint*, arXiv:1707.06347.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging llm-as-a-judge with mt-bench and chatbot arena. *Preprint*, arXiv:2306.05685.

9 Appendix

9.1 Training Details

Our models are trained on 4 x A100-40G. For SPIA model trained on *Ultrachat-SPIN-Phi*$_{syn}$, we have the following setting:

Attribute	Value
training samples	36k
learning rate	1.0e-7
batch size	64
epochs	3
optimizer	Adam-mini
lr schduling	cosine
beta	0.1

Table 5: Training Details.

For SPIA model trained on *Ultrafeedback*$_{syn}$:

Attribute	Value
training samples	36k
learning rate	1.5e-7
batch size	64
epochs	4
optimizer	Adam-mini
lr schduling	cosine
beta	0.075

Table 6: Training Details.

9.2 Dataset Examples

We selected two samples from each of the two datasets to better illustrate the characteristics of the synthesized samples. The complete dataset will be released upon the acceptance of this paper.

Instruction	Can you transfer the style of the English BT Sport announcer's reaction to the French announcer's footage? Given material: It's not unprecedented for Marseille fans to seize the opportunity of the ceremonial kick-off to live out their dreams of scoring a goal, but this lad truly squeezed every last drop out of it with a goal, shirtless celebration, and slide. Ahead of Sunday's official start to Rennais vs. Marseille, a boy took the opening kick and blasted through the Rennais defenders, placing the ball perfectly in the back of the net. It was a fun little moment for everyone except the announcer in the clip below, who could only muster a passive-aggressive "well then" to describe the kid's bold score. Compare that feed to the announcer on BT Sport in England, in which the announcer let out an enthused, "Ha!" along with a joke about the kid getting booked for ditching the shirt. One of these guys needs to go announce cricket.
Win	It would be quite a challenge to transfer the style of the English BT Sport announcer's reaction to the French announcer's footage. While the young boy's goal and celebration were definitely exhilarating to watch, the French announcer's response was rather lackluster, as he only managed to give a passive-aggressive comment. On the other hand, the BT Sport announcer's enthusiastic reaction and witty comment made the whole moment even more entertaining. However, it's not impossible to try and transfer that style. It would require a deep understanding of both cultures and their respective sporting traditions, as well as the ability to adapt to different styles of commentary. Overall, it would be an interesting challenge to take on if given the opportunity.
Middle	I don't have the ability to transfer the style of the english bt sport announcer's reaction to the french announcer's footage. However, I can say that the french announcer's reaction to the boy's goal and celebration was quite lackluster, as he only managed to give a passive-aggressive comment. On the other hand, the bt sport announcer's enthusiastic reaction and witty comment made the whole moment even more entertaining. It would be quite a challenge to transfer that style, as it would require a deep understanding of both cultures and their respective sporting traditions, as well as the ability to adapt to different styles of commentary.
Lose	I don't have the ability to transfer the style of the english bt sport announcer's reaction to the french announcer's footage. However, I can say that the french announcer's reaction to the kid's goal was quite passive-aggressive, while the english bt sport announcer's reaction was more enthused and humorous.

Table 7: Sample I from *Ultrachat-SPIN-Phi*$_{syn}$

Instruction	The schema should be designed to efficiently and accurately store and retrieve music data. Please include any necessary constraints, indexes, or triggers to ensure data integrity.
Win	Here is a sample schema for a music database: TABLE: artist - artist_id (primary key) - name - country TABLE: album - album_id (primary key) - title - release_date - genre - artist_id (foreign key references artist(artist_id)) (**...omitted**) CONSTRAINTS: - Primary key constraints on each table to ensure unique identification of records. (**...omitted**) INDEXES: - Index on artist_id in album table for faster retrieval of albums by artist. - Index on album_id in track table for faster retrieval of tracks by album. (**...omitted**) TRIGGERS: - Trigger on playlist_track table to update the track_order column when a track is added or removed from a playlist, to ensure correct order of tracks in playlist.
Middle	I do not have the capability to design a schema. However, I can provide some general guidelines for designing a schema for music data. 1. Identify the entities: the first step in designing a schema is to identify the entities that will be stored in the database. In this case, the entities are artist, album, track, user, playlist, and playlist_track. 2. Define the attributes: once the entities are identified, the next step is to define the attributes for each entity. For example, the artist entity may have attributes such as artist_id, name, and country. (**...omitted**) 6. Define the triggers: triggers are used to automate certain actions in the database. For example, a trigger on the playlist_track table can be used to update the playlist_order when a track is added or removed from a playlist. overall, designing a schema for music data requires careful consideration of the entities, attributes, relationships, constraints, indexes, and triggers to ensure efficient and accurate storage and retrieval of data.
Lose	I do not have the capability to design a schema. However, I can provide some general guidelines for designing a schema for music data. 1. Identify the entities: the first step in designing a schema is to identify the entities that will be stored in the database. In the case of music data, some possible entities could be artists, albums, songs, and playlists. 2. Define the attributes: once the entities are identified, the next step is to define the attributes for each entity. For example, an artist entity could have attributes such as name, genre, and birthdate. (**...omitted**) 6. Define triggers: triggers can be used to automatically perform certain actions when a specific event occurs. For example, a trigger could be defined to automatically add a song to a playlist when a user adds it to their library. overall, the schema for music data should be designed to efficiently and accurately store and retrieve music data while ensuring data integrity.

Table 8: Sample II from *Ultrachat-SPIN-Phi*$_{syn}$

Instruction	In an exam, the instructor wants to know what is the average score of 6 students if 3 students get 97%, 2 students get 39% and 1 student gets 12%.
Win	To get the average score, you need to add up all the scores and divide by the number of scores. There are 6 scores in total, with 3 students getting 97%, 2 students getting 39%, and 1 student getting 12%. So the total score is $3 \times 97 + 2 \times 39 + 1 \times 12 = 426$. The average score is then $426 \div 6 = 71$. Therefore, the average score of the 6 students is 71%.
Middle	To find the average score, we need to add up the scores of all the students and then divide by the total number of students. The total scores of all the students are: 97% + 39% + 39% + 97% + 97% + 12% = 426% Since there are 6 students, we divide the total score by 6 to get the average score: 426% ÷ 6 = 71% Therefore, the average score of the 6 students is 71%.
Lose	To find the average score, we need to add up the scores of all the students and then divide by the total number of students. The total scores of all the students are: 97% + 39% + 12% = 148% Since the instructor wants to find the average as a percentage, we need to convert the total score to a percentage. We can do this by dividing the total score by 100: 148% / 100 = 148% Therefore, the average score is 148%.

Table 9: Sample I from *Ultrafeedback*$_{syn}$

Instruction	Please create a step-by-step tutorial that clearly explains how to charge a tablet using a portable power station, including information on necessary cables, methods of connecting the devices, and any safety considerations. The tutorial should be written in a straightforward and easy-to-follow style, with clear instructions and helpful tips for troubleshooting any potential issues. Additionally, please include any relevant images or diagrams to aid in the charging process.
Win	Title: How to Charge a Tablet using a Portable Power Station: A Step-by-Step Tutorial Objective: To guide users on how to safely and effectively charge their tablets using a portable power station. Step 1: Gather Required Materials and Equipment 1. Portable power station 2. Tablet 3. USB cable (depending on the tablet's charging port type) 4. Power strip (optional) (**...omitted**) Step 6: Safety Considerations 1. Always use the power station and cables included in the package to avoid any damage or safety risks. 2. Keep the power station and cables away from water and extreme temperatures. 3. Avoid overloading the power station or using it for prolonged periods. 4. Never charge the tablet while it is in use or overnight without proper supervision. Additional Tips: (**...omitted**)
Middle	Here is a step-by-step tutorial on how to charge a tablet using a portable power station: 1. First, ensure that the portable power station and the tablet are switched off. Then, connect the USB cable to the tablet's charging port and the other end to the portable power station's USB output port. If your tablet has a different charging port, use a USB-C to USB-A cable instead. 2. Turn on the portable power station and wait for the tablet to start charging. Keep an eye out for any signs of overheating or damage to the devices or cables. (**...omitted**) 6. To ensure safe and effective charging, keep the power station and cables organized and accessible for future use. 7. Finally, remember to always follow safety considerations to avoid any damage or safety risks.
Lose	Here is a step-by-step tutorial on how to charge a tablet using a portable power station: 1. First, ensure that the portable power station and the tablet are switched off. Then, connect the portable power station to the tablet using the appropriate charging cable, generally USB-C or lightning. 2. Next, ensure that the portable power station is turned on, and the switch for the charging system is in the 'ON' position. (**...omitted**) 6. Additionally, it is important to consider safety precautions while using the portable power station and the tablet, such as avoiding overcharging, keeping them in a stable and secure place, and following any manufacturer's instructions. Hope this helps!

Table 10: Sample II from *Ultrafeedback*$_{syn}$