FYP24080 Final Report

Personalization of Large Language Models for Diverse User Preferences

Student: LIU Meitong

Supervisor: Prof. LUO Ping

Department: Computer Science

Date of Submission: April 21, 2025

The University of Hong Kong

Abstract

Current Large Language Models (LLMs) post-training approaches fail to adequately capture the heterogeneous and often conflicting nature of human preferences. This final year project explores effective methodologies for personalizing LLMs by tackling the reward modeling phase, a critical step of the Reinforcement Learning from Human Feedback (RLHF) pipeline. On a high level, we propose a conditional reward modeling formulation that aims to learn user-specific preference distributions, enabling preservation of and adaptation to different opinions. To realize this conditional reward model, we first evaluate the training-free few-shot prompting method, finding it insufficient for effective personalization. Subsequently, we experiment with a Variational Auto-Encoder (VAE) based structure proposed in the literature, which outperforms the traditional user-agnostic model but exhibits severe training instability. To address this limitation, we develop an adaptive sampling strategy following continual learning principles that significantly stabilizes training and accelerates convergence. Other possible solutions are also examined, including gradient manipulation techniques used in Multi-Objective Optimization (MOO) and a contrastive learning paradigm. Further explorations of user preference simulation with fine-grained user stories in the prompt reveal meaningful results, with users in similar occupations demonstrating comparable tastes. This project deepens our understanding of the theories and practice of LLM personalization and contributes to equitable AI access across diverse user groups, enhancing human-centric AI alignment.

Acknowledgement

I would like to extend sincere gratitude to my supervisor, Prof. Luo Ping, for his guidance on this project, and to Prof. Zhao Han from the University of Illinois Urbana-Champaign for providing computational resources. I also want to thank the best parents of mine for their unconditional love and support, and my amazing friends for fruitful discussions, jokes, and four years of good times.

Contents

1 Introduction 9

1.1 Current RLHF Failing to Reflect Preference Heterogeneity 9

1.2 Project Objectives and Summary of Outcomes 10

1.3 Report Outline 12

2 Preliminary and Related Work 12

2.1 Reinforcement Learning from Human Feedback 12

2.1.1 Reward Modeling 13

2.1.2 Policy Training 14

2.2 Literature Review on Dataset Curation and Related Methods 14

3 Conditional Reward Modeling and Its Realization 15

3.1 Section Overview 15

3.2 Conditional Reward Modeling 15

3.3 Baseline: Few-Shot Prompting 17

3.3.1 Experimental Setup 17

3.3.2 Results and Analysis 19

3.4 Realizing Conditional Reward Modeling with Variational Auto-Encoder 20

3.4.1 Challenges of Direct Finetuning 21

3.4.2 VAE-based Model Structure 21

3.4.3 Objective Interpretation 23

3.5 Experimental Results and Major Limitations 24

3.5.1 Experimental Setup 24

3.5.2 Results and Limitations of Failed Training 26

4 Techniques for Stabilizing and Accelerating Training 28

4.1 Section Overview 28

4.2 Adaptive Sampling and Improved Results 28

4.2.1 Conflicts Between Different Users 28

4.2.2 A Simple Adaptive Sampling Strategy 30

4.2.3 Improved Results and Analysis 31

4.3 Attempt: Multi-Objective Gradient Manipulation 33

4.3.1 Introduction to Multi-Objective Optimization 33

4.3.2 MGDA and PCGrad 34

4.3.3 Results and Analysis 35

4.4 Attempt: Contrastive Learning 35

4.4.1 Motivation and Contrastive Loss 36

4.4.2 Results and Analysis 37

5 Prompting Larger Models with Fine-Grained User Stories 39

5.1 Section Overview 39

5.2 Experimental Setup 39

5.3 Mitigating Positional Bias 40

5.3.1 Inconsistency in User Choices and Homogeneous Preferences 40

5.3.2 Scoring Querying Mode 41

5.4 Results and Interpretation 43

5.4.1 Diverse Preferences of Different Users 43

5.4.2 Similar Preferences among Similar Occupations 44

5.4.3 Discussion 45

6 Conclusion, Limitation, and Future Work 45

A Sample Prompts **51**

A Few-shot prompting prompts 51

B Scoring criteria and roleplay prompts 51

B Complete Version of Partial Figures **52**

List of Figures

1 Illustration of the flaws of current preference alignment 9
2 Overview of the RLHF pipeline 13
3 Visualization of user opinions under two settings 18
4 Prediction accuracy of the few-shot prompting method under two settings . . . 20
5 VAE-based conditional reward model 22
6 Test results of multiple methods on the personalized UltraFeedback dataset . . 26
7 Evaluation curves of the conditional reward model across three runs 28
8 Evaluation accuracy of individual users in the two failed runs 29
9 Evaluation accuracy of the conditional reward model with adaptive sampling . . 31
10 Evaluation curves of the conditional reward model with adaptive sampling . . . 32
11 Comparison of convergence speed 32
12 Evaluation accuracy of individual users with adaptive sampling 33
13 Illustration of MGDA and PCGrad 35
14 Results using MGDA and PCGrad 36
15 t-SNE visualization of latent user embeddings in a successful run 37
16 Training contrastive loss and evaluation accuracy curves 38
17 t-SNE visualization of latent user embeddings with contrastive loss 38
18 Per user preference distribution (partial) 41
19 Per question preference distribution 41
20 Simulation results with scoring mode 42
21 Frequencies of the responses being selected 43
22 Pairwise similarity of user preferences 44
23 Connected graphs of mutual top five similar users 45
24 Full version of Figure 18a 52

25 Full version of Figure 18b 53

26 Full version of Figure 18c 53

27 Full version of Figure 20a 53

List of Tables

1 Hyperparameters used for the VAE-based model 25

2 Numerical test results of multiple methods 27

3 Numerical test results of the conditional reward model with adaptive sampling . 31

4 Training epochs and times 32

List of Abbreviations

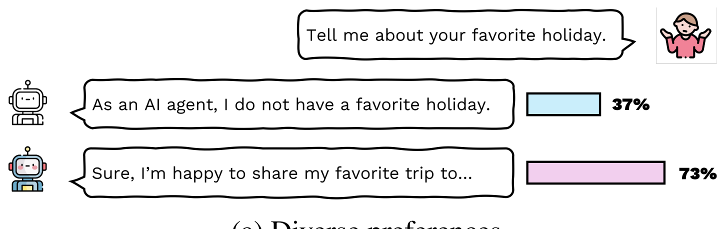
AI	Artificial Intelligence
BTL	Bradley-Terry-Luce
DPO	Direct Preference Optimization
LLM	Large Language Model
MOO	Multi-Objective Optimization
MGDA	Multiple Gradient Descent Algorithm
PCGrad	Projecting Conflicting Gradients (Algorithm)
PPO	Proximal Policy Optimization
RL	Reinforcement Learning
RLHF	Reinforcement Learning from Human Feedback
SFT	Supervised Finetuning
VAE	Varational Auto-Encoder

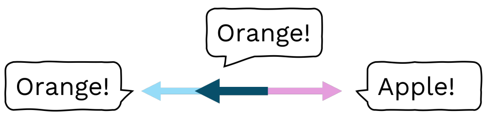
1 Introduction

1.1 Current RLHF Failing to Reflect Preference Heterogeneity

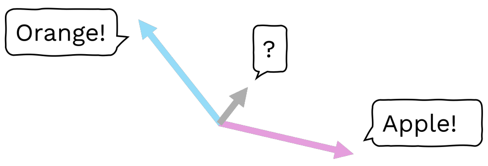
Recent years have witnessed a drastic development of Large Language Models (LLMs) (Chang et al., 2024). Demonstrating remarkable capabilities in natural language processing, their deployment in numerous applications, from document refinement (Wang, 2023) to web agents (Yao et al., 2022), has made them not only integral to industries but also a powerful daily life tool for people from all walks of life.

The workflow for training a Large Language Model typically involves two stages (Achiam et al., 2023): pre-training and post-training. During the pre-training phase, the model is trained on internet-level corpora to learn linguistic patterns and perceive a wide range of concepts. Following that, the post-training stage refines the model's abilities in instruction-following and aligns its performance with human morals and preferences. Based on the widely acknowledged framework proposed by OpenAI (Ouyang et al., 2022), the post-training stage also contains two steps: supervised fine-tuning (SFT) and preference alignment. SFT further fine-tunes the pre-trained model on curated datasets with expert-provided demonstrations, where the model learns to mimic high-quality human output. Preference alignment focuses on matching the model's behavior with user preferences and ethical guidelines. This is typically done through Reinforcement Learning from Human Feedback (RLHF), where responses favored by human annotators are encouraged, while those disliked are penalized. This project focuses on the preference alignment stage, where the current RLHF scheme suffers from inherent flaws that


Tell me about your favorite holiday.
As an AI agent, I do not have a favorite holiday.
37%
Sure, I'm happy to share my favorite trip to...
73%
(a) Diverse preferences



(b) Underrepresented minority



(c) Conflicting opinions cancelled out

Figure 1: Illustration of the flaws of current preference alignment

hinder an LLM's ability to personalize to diverse human tastes.

Human preferences are heterogeneous and often conflicting. Users with different backgrounds, for instance, race, gender, and personality, may favor different LLM responses. For instance, as shown in Figure 1a, those who prefer realistic, strictly programmed agents may favor the first response, while those who want a more human-like, vivid model would go for the second. However, common datasets used for preference alignment rely on anonymous annotators who rank multiple responses without accounting for their demographic or contextual information. This leads to a mixed, user-agnostic representation of opinions, and models trained on such data often struggle to reflect the broad spectrum of user preferences.

Specifically, current training methods treat each comparison sample equally. When different preferences arise, inducing conflicting forces that navigate how the model adapts, the model misinterprets some of them as data noise. In cases illustrated in Figure 1b, voices from the majority dominate. Consequently, the learned model tends to underrepresent minority groups (Durmus et al., 2023). Alternatively, equal and opposite preferences may be cancelled out, causing an eventual failure in learning any preference at all (Chakraborty et al., 2024).

To encapsulate, there are two bottlenecks for diverse preference modeling: insufficient profiling of annotators and ineffective algorithms to distinguish and preserve conflicting preferences. Recent advancements in dataset curation have alleviated the first limitation by incorporating detailed annotator information or fine-grained multi-dimensional ratings for each preference datapoint, allowing for a richer and more diverse representation of user preferences (Santurkar et al., 2023; Durmus et al., 2023; Castricato et al., 2024; Cui et al., 2023). Given such progress, this project aims to explore feasible frameworks that embed diverse user preferences in one model and enable it to personalize adaptively.

1.2 Project Objectives and Summary of Outcomes

This project investigates the personalization of LLMs during the preference alignment post-training stage. We focus on reward modeling, the first and most critical step of the RLHF pipeline, where a reward model is trained to select the correct, favored response given a querying prompt and several candidate responses. Different from the typical reward modeling process, where diverse preference information is lost with all signals mixed, we use **conditional reward modeling**, where individual tastes are separated, preserved, and able to be elicited with proper user identifiers. The conditional modeling changes the model specification on a fundamental level, providing key support for tackling preference heterogeneity.

We then explore practical methods that realize this formulation. The training-free few-shot prompting approach, although simple and cost-efficient, is shown to be ineffective in adapting model predictions to individual clients. Poddar et al. (2024) recently proposed a reward model structure with a Variational Auto-Encoder (VAE) backbone, which proves capable of inferring latent user preference distributions from demonstrations and navigating model outputs accordingly. However, we find that the training process of this model often fails to converge, potentially due to conflicts between signals from different users. We explore various solutions, including gradient manipulation techniques from the multi-objective optimization literature and a contrastive learning paradigm. Eventually, a simple adaptive sampling strategy conceptually following a continual learning fashion proves effective. It stabilizes training and significantly reduces the number of epochs required for convergence.

Apart from preference demonstrations, we also examine the possibility of user stories as identifiers. By prompting a larger language model with fine-grained client descriptions and adopting carefully designed techniques to overcome the issue of positional bias, we obtain simulation results that demonstrate diverse preferences. Similar tastes are detected between users with similar social occupations, verifying the validity of the simulation and providing insights into how preferences are related to social attributes.

Our major outcomes are summarized as follows:

- We propose conditional reward modeling to preserve diverse human preferences on a specification and conceptual level.
- We experiment with practical methods to realize conditional reward modeling, including few-shot prompting and a VAE-based structure from the literature.
- To solve the training instability issue, we examine several solutions, including multi-objective gradient manipulation techniques, contrastive learning, and propose an effective adaptive sampling strategy that stabilizes and accelerates convergence.
- We investigate using fine-grained user stories and identifiers and simulate preference profiles by prompting a larger model. Results demonstrate diverse opinions and align reasonably with social attributes.

Through this project, we gained a deeper understanding of LLM preference alignment and how personalization can be achieved algorithmically and practically. With these insights, we further promote the call out to the research community importance and possibility of equal and high-quality access to AI resources to every social member.

1.3 Report Outline

The main body of the report is organized into four sections. Section 2 introduces technical preliminaries related to RLHF and reward modeling, followed by a review of existing related works. Section 3 begins by elaborating on the proposed conditional reward modeling formulation, which serves as the major high-level methodology. It then presents results of the few-shot prompting approach, specifies the structure of the VAE-based model, and presents experimental results. Given the limitation in failed training, Section 4 continues in exploring several solutions. To maintain a consistent logic flow, the effective adaptive sampling strategy is discussed first, followed by reports on multi-objective gradient manipulation techniques and a contrastive learning paradigm. Section 5 illustrates the simulation of roleplaying a larger language model with fine-grained user stories. Finally, the report ends with a concluding statement in Section 6.

2 Preliminary and Related Work

This section introduces the preliminary technologies and related works. Section 2.1 goes through the RLHF pipeline, with details for reward modeling. Section 2.2 provides a literature review on user-specific or multi-dimensionally rated RLHF dataset curation, as well as methods motivated by the heterogeneity of human preferences.

2.1 Reinforcement Learning from Human Feedback

Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022) is a predominantly adopted method for the preference alignment stage in the post-training pipeline of LLMs and is also in applied robotics (Torne Villasevil et al., 2023). It aims to align model behaviours with human preferences and morals, effectively improving accuracy and safety.

Figure 2 (Dong et al., 2024) illustrates the overall workflow of RLHF. It begins with training a

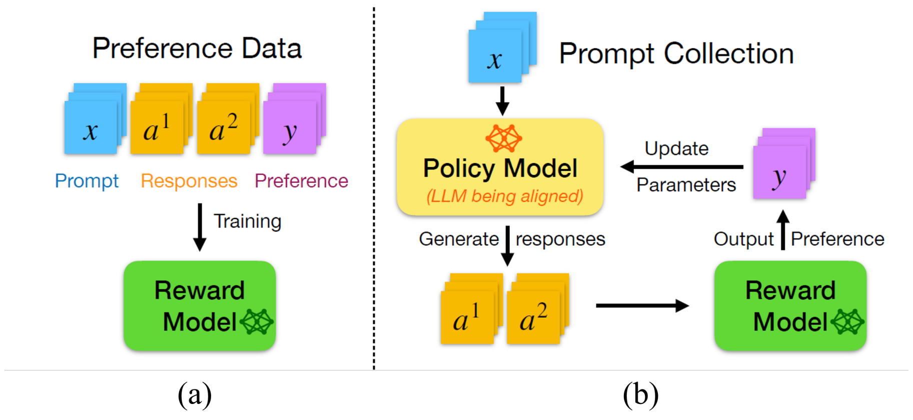


Figure 2: Overview of the RLHF pipeline

reward model capable of predicting human preferences via supervised learning on a dataset of (prompt, candidate responses, human annotated preference) entries. After the reward modeling stage, the policy training stage proceeds. The established reward model is incorporated into a Reinforcement Learning (RL) cycle, guided by typical RL algorithms (Haarnoja et al., 2018; Kostrikov et al., 2021), such as Proximal Policy Training (PPO) (Schulman et al., 2017). In the following subsections, we elaborate more on reward modeling and policy training, respectively.

2.1.1 Reward Modeling

The reward modeling stage targets a reward model, usually based on language models, that can predict the correct preferred response given a query and multiple candidate responses. It follows a supervised training paradigm. With labeled preference data, the objective is to maximize the likelihood that the ground-truth preferred response is favored. For example, on a typical RLHF dataset with binary comparison between responses, we have

$$\max_{r_\theta} \mathop{\mathbb{E}}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log p_\theta(y_w \succ y_l \mid x)\right], \tag{1}$$

where r_θ refers to the reward model parameterized by θ to be learned, $\mathcal{D}$ denotes the preference dataset, the triplet (x, y_w, y_l) denotes a datapoint, x is the prompt, y_w is the preferred response, and y_l is the rejected response. Here, $p_\theta(\cdot)$ is a probability calculated on the model outputs using the Bradley-Terry-Luce (BTL) choice model (Bradley and Terry, 1952):

$$p_\theta(y_w \succ y_l \mid x) = \frac{e^{r_\theta(y_w|x)}}{e^{r_\theta(y_w|x)} + e^{r_\theta(y_l|x)}}, \tag{2}$$

where $r_\theta(y \mid x)$ represents the reward score the model assigns to response y given prompt x. However, such reward modeling is user-agnostic, essentially integrating preference distributions over the population into a joint one. As a consequence, diverse opinions are ignored and lost in the learning process. In Section 3, we propose the conditional reward modeling formulation by incorporating user identities into the above equations. Different from the traditional reward models, conditional reward models can output different choices in answer to the same querying triplet for different clients, preserving and adapting to diverse tastes.

2.1.2 Policy Training

Policy training via RL usually requires loading multiple models and fully fine-tuning language agents, taking vast computational resources and time. Therefore, this project only focuses on the reward modeling process, which lays the foundation of a successful RLHF run. Nevertheless, we still briefly introduce the process of policy training.

As shown in Figure 2, during each iteration of the process, the policy model to be learned first generates several responses to an input prompt. Next, the reward model obtained from the previous stage assigns real-valued scores to each of the responses. Based on this feedback, the policy model is then optimized to maximize the reward score of its output, guided by the adopted RL algorithm. This process allows the model to learn and adapt, refining its behavior to meet user expectations.

2.2 Literature Review on Dataset Curation and Related Methods

Modeling diverse human preferences requires data. Instead of the commonly used datasets where preference labels are annotated by anonymous users and aggregated as a whole, personalization requires each preference to be distinguishable. One straightforward method is to provide user information. The OpinionsQA dataset (Santurkar et al., 2023) collects US citizens' opinions on controversial political issues, with each annotator's demographic information recorded in 8 attributes. Built on this, the GlobalOpinionsQA dataset (Durmus et al., 2023) further incorporates more countries and more questions. However, these two datasets only include questions from limited topics, mainly political, and thus cannot serve as a good general preference learning dataset for LLMs. In this project, we mainly use the UltraFeedback dataset (Cui et al., 2023), which contains ratings for each response from different aspects. Although lack of detailed annotator attributes, the fine-grained ratings enable data curation by creating users who attach different weights to each perspective. Detailed methods will be clarified later.

Due to the lack of proper preference datasets, there exists little work that targets preserving diverse user tastes in one model. Nevertheless, some algorithms have tried to solve an easier problem where one finds a compromising solution among the preferences of different groups. Chakraborty et al. (2024) adopts the typical RLHF workflow, training a separate reward model for each group and optimizing the policy model for the worst-performing one in each training round. Ramesh et al. (2024) uses the alternative DPO (Rafailov et al., 2024) algorithm equipped with mirror descent that also prioritizes undertrained groups. These methods aim to reconcile the conflicts among different preferences, yet cannot preserve them.

3 Conditional Reward Modeling and Its Realization

3.1 Section Overview

This section discusses the main methods used to achieve diverse preference learning for LLM reward models. Specifically, Section 3.2 introduces conditional reward modeling, which serves as the fundamental support on the model specification level. Before investigating reward model training, in Section 3.3, we report results of the training-free few-shot prompting method as a baseline. Next, Section 3.4 introduces an effective model structure from recent literature to realize condition reward model training using the Variational Auto-Encoder (Poddar et al., 2024). Finally, Section 3.5 presents its experimental performance and major limitations, which will be resolved by techniques documented in the Section 4.

3.2 Conditional Reward Modeling

Recall that the reward modeling stage, as the first step of the RLHF process, aims to obtain a model that predicts users' preferences among multiple responses to the same query. As introduced in Section 2.1, current reward models are trained in a supervised learning fashion on a mixed, user-agnostic preference dataset. However, this process is valid only when all users share the same underlying preference. In the real situation of heterogeneous tastes, it essentially aggregates opinions among the population. The resulting model tends to favor the majority while leaving the minority underrepresented, and diverse preference information may be lost due to gradient conflicts.

The above issues cannot be tackled if the learner cannot distinguish signals from different groups. In other words, one has to modify the model specification to incorporate this factor into the learning process. Naturally, we consider **conditional reward modeling**. Specifically, the objective function is

$$\max_{r_\theta} \mathop{\mathbb{E}}_{z \sim \mathcal{Z}} \left[\mathop{\mathbb{E}}_{(x^z, y_w^z, y_l^z) \sim \mathcal{D}_z} \left[\log p_\theta(y_w^z \succ y_l^z \mid x^z, z) \right] \right], \tag{3}$$

where r_θ is the reward model to be learned parametrized by θ, $\mathcal{Z}$ is the set of all possible users, $\mathcal{D}_z$ is the user-specific preference dataset, and $p_\theta(y_w^z \succ y_l^z \mid x^z, z)$ is computed as

$$p_\theta(y_w^z \succ y_l^z \mid x^z, z) = \frac{e^{r_\theta(y_w^z \mid x^z, z)}}{e^{r_\theta(y_w^z \mid x^z, z)} + e^{r_\theta(y_l^z \mid x^z, z)}}, \tag{4}$$

which is similar to the original Bradley-Terry-Luce choice model as in Equation (2). Comparing the above conditional reward modeling with the traditional one, the simple but key difference lies in its consideration of users. Instead of fitting the marginal preference distribution $p(\cdot \mid x, y_w, y_l)$, which mixes voices from all users together, it fits the **conditional** distribution $p(\cdot \mid x, y_w, y_l, z)$ given a specific client. Such conditioning enables the model to embed conflicting opinions from diverse groups in the conditional distributions and to adapt its behaviour, i.e., the preference prediction, according to the given user. This conditional reward modeling provides the key ingredient for LLM personalization.

Theoretically, the user label z can be any data that identifies one user from another, for instance, a user ID. Nevertheless, using meaningful information from which the user's preference can be best inferred, such as social attributes, life experiences, personality, and preference examples, is more effective for the learning process and helps generalization to a broader population, utilizing the semantic interpretation ability of language models. In all the experiments in this section, we only consider preference examples as user identifiers due to limited data. In Section 4, we also explore using synthetic user stories as identifiers.

3.3 Baseline: Few-Shot Prompting

Training LLMs is highly consuming in both time and memory. Hence, before training conditional reward models, we first examine the performance of a tuning-free method - few-shot prompting. Specifically, a post-trained LLM is directly used as a reward model and prompted to predict user preferences.

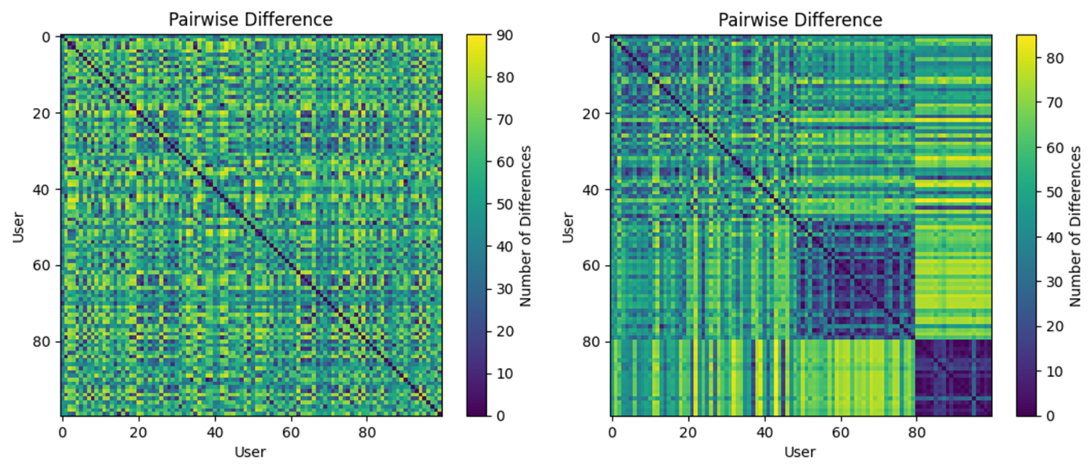
Apart from the prompt-response triplet to be annotated, which consists of a prompt and two responses, the model is additionally provided with a small number of preference examples of a specific user. Each preference example is in the same format of a query triplet, but with the ground-truth label telling which response is preferred by the user under consideration. This method expects the model to infer the underlying preference of the given client through limited demonstrations.

3.3.1 Experimental Setup

Dataset and preprocessing We consider a widely used RLHF dataset, UltraFeedback (Cui et al., 2023). Each entry of UltraFeedback consists of a query, $2 \sim 4$ responses generated by language models, and their corresponding multi-objective scores rated on four dimensions, namely instruction-following, honesty, truthfulness, and helpfulness. When training a reward model, these four scores are normally aggregated as the overall score of a response, and the accepted and rejected pairs can then be determined.

We follow the suggested procedure in the original work to filter out potentially contaminated entries. Additionally, to encourage disagreements among users, we only consider queries with 4 responses whose scores are not dominated by each other. Eventually, 300 questions are batched into the training set, and 95 questions constitute the test set. In fact, 300 questions for each user is not sufficient for effective conditional reward model learning, and due to limited computational resources, increasing the number of training samples for all users is not affordable. Therefore, we use a dataset with fewer users later to train the reward model, as in Section 3.5, and only the test set introduced here is used for evaluating the few-shot prompting method.

Simulating diverse preferences The multi-objective ratings enable us to simulate diverse user preferences. We model each user as a preference vector $w \in \Delta_4$, where w_i represents



(a) Random (b) Cluster

Figure 3: Visualization of user opinions under two settings

the importance this user attaches to a certain dimension. As such, the user-specific rating for a query is the weighted sum, instead of the uniform sum, of the sub-scores. When comparing a pair of responses, different users may choose oppositely. Specifically, we generate 100 preference vectors using two strategies:

- Random: sample 99 variables from $\mathrm{Dirichlet}([1, 1, 1, 1])$, plus 1 uniform baseline
- Cluster: sample 49, 30, and 20 variables from $\mathrm{Dirichlet}([2.5, 2.5, 2.5, 2.5])$, $\mathrm{Dirichlet}([0.7, 0.3, 6.0, 3.0])$, and $\mathrm{Dirichlet}([7.5, 0.5, 1.2, 0.8])$ respectively, plus 1 uniform baseline

Here, the uniform baseline refers to $w = [0.25, 0.25, 0.25, 0.25]$, representing the uniform aggregation used in current post-training schemes. Figure 3 visualizes the user opinions, where each entry indicates the number of differences between two users in their choice of the most preferred response for a certain query among the 95 test queries. Under the random setting, users demonstrate highly diverse opinions, as in Figure 3a; under the cluster setting, conflicts between clusters are more tense, and inter-cluster differences are mitigated, as in Figure 3b.

Question selection by information gain To facilitate the few-shot prompting method, several preference examples should be provided in the prompt to demonstrate user tastes. To find the most informative set, we adopt a greedy selection strategy based on mutual information. In particular, the question with the highest information gain is selected in each iteration. In other words, we choose question j that maximizes

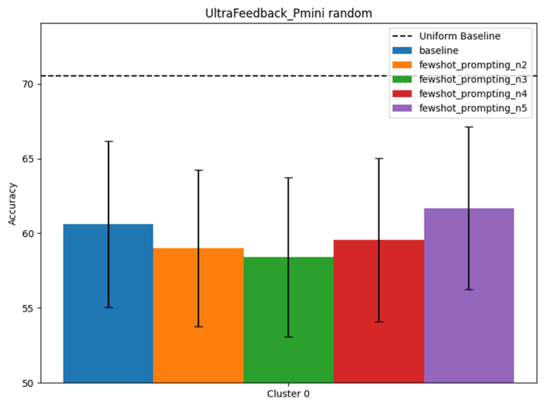
$$
\begin{aligned}
I(Z; Q_j \mid Q_S) &= I(Z; Q_{S\cup\{j\}}) \\
&= \sum_{z\in Z}\sum_{a\in A_j} p(z,a)\log\frac{p(z,a)}{p(z)p(a)} \\
&= \sum_{z\in Z}\left(p(z,a_z)\log\frac{p(z,a_z)}{p(z)p(a_z)} + 0\right) && (5)\\
&= \sum_{z\in Z} -\log p(z)p(a_z) && (6)\\
&= \sum_{z\in Z} -\log\frac{\text{number of users choosing } a_z}{100} && (7)
\end{aligned}
$$

where Z is the user set, Q_j is the response of question j, S is the set of questions already chosen, Q_S is the set of responses to all questions in S, A_j is the response set of question j, a_z is the response user z prefer the most for query j. Additionally, $p(z,a)$ is the probability of user z preferring response a and equals 1 when $a = a_z$ and 0 otherwise; $p(z)$ is the probability of selecting user z from a total of 100 users and equals $\frac{1}{100}$, $p(a_z)$ is the probability of any user preferring a_z for question j, estimated by the empirical number of users choosing a_z in the dataset. Note that this greedy approach may not eventually yield the most informative set, but is widely adopted for its computation efficiency and satisfactory performance in practice (Minoux, 2005).

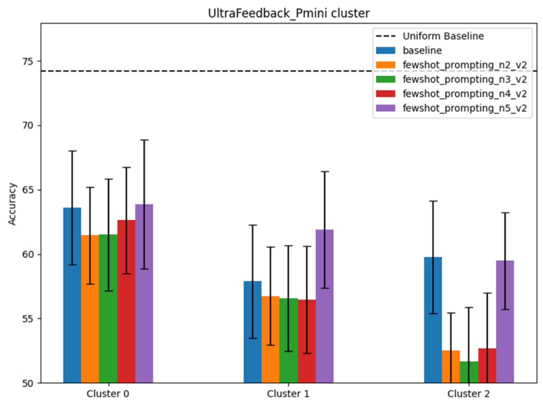
Models and machines We adopt a Gemma-2 model with 2B parameters that is post-trained on a joint set of common RLHF datasets based on the open-source SFT model provided by Google (Team, 2024). This model has a basic knowledge of human preferences learned from the mixed data. All experiments are conducted on one machine with one A100 GPU.

3.3.2 Results and Analysis

In the experiments, the model is directly prompted with 2 $\sim$ 5 preference demonstrations from a particular user, a query, and two responses, and then answers which response is more likely to be preferred by the current user. For illustration, an example prompt is attached in Appendix A. Figure 4 presents the results using different numbers of few-shot demonstrations, evaluated by the prediction accuracy on the test set.



(a) Random



(b) Cluster

Figure 4: Prediction accuracy of the few-shot prompting method under two settings

In Figure 4, the dotted line represents the accuracy of the model with no few-shot demonstrations for the user with the uniform preference, which reflects the currently reported accuracy using the traditional user-agnostic reward modeling and training. While this accuracy is relatively high, all results represented by the bar plots, where the underlying user preferences are highly heterogeneous, are much worse, verifying that the current RLHF scheme may fail to cater to diverse clients in real-world applications.

Comparing across multiple bars, one observes that when the number of few-shot demonstrations is small, performance is worse than the baseline with no demonstrations, which may be caused by misinterpretation from limited data. Only when the number of examples reaches 5 is the accuracy improved to a small extent. In the clustered setting, improvements among different groups are not consistent, with cluster 1 witnessing the largest increase, while the other two nearly none. This indicates that the selected demonstration set may be somewhat biased, providing more information for certain groups.

From these experiments, we conclude that the training-free few-shot prompting method is quite limited in dealing with diverse user preferences. In the next subsection, we explore a Variational Auto-Encoder-based reward model structure from recent literature(Poddar et al., 2024) that is effective in personalization but suffers from unstable training.

3.4 Realizing Conditional Reward Modeling with Variational Auto-Encoder

In the previous subsection, the few-shot prompting method utilizing solely the semantic understanding and inference ability of language models is shown to be not powerful enough for preference personalization. Instead, we consider **training** a conditional reward model with preference demonstrations as user identifiers.

3.4.1 Challenges of Direct Finetuning

One straightforward approach is to append all examples into prompts with the querying prompt-response triplet and directly fine-tune the language model to fit the correct choice. However, this method, despite its simplicity, is challenged by several factors. The first and main problem stems from the limited context window size of language models. In other words, the length of the querying prompt a model can process is restricted. Some preference demonstrations are lengthy, and the number of demonstrations is also strictly upper-bounded by this window size. For example, the GPT-2 model (Radford et al., 2019) has a window size of 1024 tokens and can accommodate approximately two triplets in the UltraFeedback dataset, leaving space for only one demonstration.

Another challenge lies in the ability of language models to automatically infer the underlying user preferences from the demonstrations and project them into the decision-making process for the querying triplet. Although language models are trained to interpret contextual meaning and may indeed have the potential to achieve it, there is no explicit objective in the said simple finetuning approach that encourages the model to do it, and it is hard to find such an objective. With the direct goal of choosing the correct response, whether the model can establish a good connection between the demonstrations and the decision it is asked to make remains unknown.

3.4.2 VAE-based Model Structure

Considering the above two drawbacks, we search for a proper model structure that can deal with multiple, potentially long preference examples and has specialized designs to infer user tastes before choosing a favored response. Recently, Poddar et al. (2024) proposed a Variational Auto-Encoder (VAE) -based backbone that can be used to realize conditional reward model training, and was reported to achieve the state-of-the-art performance. Here, we first introduce the architecture in detail for completeness, and later point out some training failure issues encountered in practice in Section 3.5, as well as our solutions in Section 4.

Figure 5 illustrates the structure of the VAE-based conditional reward model. To better explain

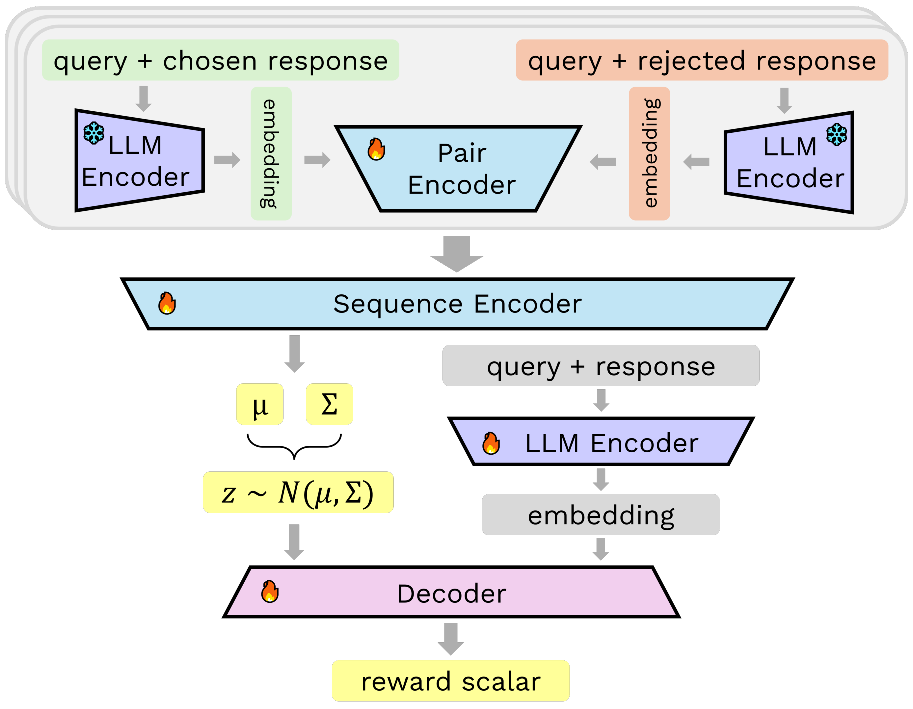


Figure 5: VAE-based conditional reward model

how data is passed through this pipeline, we first consider the inference process of a sample consisting of several preference demonstrations and the querying triplet.

1. For each demonstration, the prompt is concatenated with the chosen and rejected responses, respectively, and separately passed into the same LLM encoder to obtain the corresponding embedding vectors. Note that the parameters of this LLM encoder are frozen, meaning that this module is not updated during training. Next, the embeddings of the chosen and rejected conversations are input into a pair encoder to be learned, which outputs a joint embedding for the entire demonstration.

2. The embeddings of all demonstrations are then processed by a sequence encoder to be learned, which integrates information and infers the latent user taste, reflected by the generated mean and variance of the preference distribution. Next, a variable representing the current user's taste is sampled from a Gaussian distribution parametrized by the output mean and variance. This step follows the typical VAE design.

3. In parallel, the prompt and one response of the querying triplet are concatenated and passed into an LLM encoder, which shares the same structure as the one mentioned in step 1 but is updated during training. The output embedding, together with the sampled user variable in step 2, is input into the decoder to be learned. Finally, the decoder outputs

the reward score of the given response. Out of the two responses in the querying triplet, the one with a higher reward score will be chosen.

This structure resolves the two challenges confronted by the naive finetuning approach. Demonstrations are processed independently by the pair encoder, which produces embeddings with a controlled length, and integrated by the sequence encoder, bypassing the restriction of the LLM context window size. For another thing, the VAE structure that infers latent user preference distributions explicitly forces the model to follow the learn-and-decide step.

3.4.3 Objective Interpretation

To end this subsection, we consider the objective function used to train this VAE-based conditional reward model. Following the classical VAE framework, the objective also consists of a maximum likelihood term, corresponding to the reconstruction loss in usual VAE training, and a regularization term concerning the prior of the latent space. Specifically, we have

$$\max_{p_\theta, q_\psi} \mathop{\mathbb{E}}_{\substack{h \sim \mathcal{H}, \\ \{(x^h, y_w^h, y_l^h)_c\}_{i=1}^N \sim \mathcal{D}_h, \\ (x^h, y_w^h, y_l^h) \sim \mathcal{D}_h}} \Bigg[\mathop{\mathbb{E}}_{z \sim q_\psi(\cdot | \{(x^h, y_w^h, y_l^h)_c\}_{i=1}^N)} \left[\log p_\theta(y_w^h \succ y_l^h \mid x^h, z) \right] \\ - \beta \, \mathrm{KL}\left(q_\psi(\cdot \mid \{(x^h, y_w^h, y_l^h)_c\}_{i=1}^N) \parallel p_z(\cdot) \right) \Bigg], \tag{8}$$

where $p_\theta(y_w^h \succ y_l^h \mid x^h, z)$ is computed using Equation (4), where $r_\theta(\cdot)$ denotes the output reward score of the model for a certain response. Here, p_θ covers the LLM under-training and the decoder, q_ψ covers the pair and sequence encoders, $\mathcal{H}$ is the user set, $\mathcal{D}_h$ is the user-specific preference dataset, $\{(x^h, y_w^h, y_l^h)_c\}_{i=1}^N$ are the preference examples, (x^h, y_w^h, y_l^h) is the querying triplet, $q_\psi(\cdot \mid \{(x^h, y_w^h, y_l^h)_c\}_{i=1}^N)$ denotes the posterior distribution of the underlying user preference given the examples, $p_z(\cdot)$ denotes its prior distribution, $\mathrm{KL}(\cdot \| \cdot)$ computes the KL-divergence between two distributions, and β is a hyperparameter.

The first term in Equation (8), the maximum likelihood term, corresponds to the vanilla conditional reward modeling objective in Equation (3), which aims to predict the correct response favored by a specific user. The KL-divergence term regularizes the posterior to not deviate too much from a given prior, which is usually a standard Gaussian distribution. In practice, this regularization is relatively less useful for reward learning, as the true population may be highly multimodal. Hence, the hyperparameter β can be simply set to 0 or a very small number.

3.5 Experimental Results and Major Limitations

This subsection presents the experimental results showing that the VAE-based conditional reward model training is *sometimes* effective in preserving diverse user preferences, yet suffers from unstable training, with many runs failing to converge.

3.5.1 Experimental Setup

Dataset and preprocessing We still consider the UltraFeedback dataset (Cui et al., 2023) and mainly use its binarized version as our testbed. In the binarized version, each entry consists of a prompt and only two candidate responses, which are extracted from the candidate pool of the original dataset. Again, following the documented protocol from the source, entries that are potentially contaminated are filtered out. Different from the experiments for few-shot prompting in Section 3.2, we did not focus on questions with 4 non-dominating responses only. Instead, such selection is left to be done after diverse user simulation.

Simulating diverse preferences To have a sufficient number of training samples for each user for successful latent preference learning, and limited by time costs and access to GPU resources, only two different users were simulated in the following experiments. Similarly to the few-shot prompting setup, we utilize the fine-grained scores for four aspects, namely helpfulness, honesty, instruction-following, and truthfulness, of each response provided by the UltraFeedback dataset to construct different preference profiles. Specifically, one user prioritizes solely “helpfulness” and the other favors only “honesty”, i.e., their preference weights are [1,0,0,0] and [0,1,0,0], respectively. Given the two responses to the same prompt, these two users may choose differently according to the subscore. To maximize the disagreement, triplets where the two users prefer the same response or attach equal rates to the two candidate responses are filtered out during training. As a result, 30896 and 423 triplets are selected for the training and test sets, respectively, of each user. Note that samples for different users are not necessarily identical.

Table 1: Hyperparameters used for the VAE-based model

Hyperparameter	**Value**
Output embedding dimension of the LLM encoders	1024
Hidden layer dimension of the pair encoder	512
Output dimension of the pair encoder	512
Dimension of the latent space	512
Hidden layer dimension of the decoder	512
Learning rate	0.0001
Optimizer	AdamW with weight decay = 0.001
Scheduler	Cosine with 3% warmup steps
Per device batch size	4
Gradient accumulation step size	8
Number of training epochs	2

Preference demonstration selection Recall that preference demonstrations should be provided for each querying triplet as user identifiers. A total of 8 examples are randomly sampled from a subset of 100 entries of the original dataset for each querying triplet. Here, we did not perform greedy selection based on information gain as in Section 3.2, but the performance is satisfactory. Note that this selection is completed in advance - the training and test sets are fixed across all runs.

Sub-modules for the VAE-based model As shown in Figure 5, there are several sub-modules in the backbone. Following the settings from the paper (Poddar et al., 2024), we use a two-layer Multi-Layer Perceptron as the pair encoder, a self-attention layer as the sequence encoder, and a two-layer Multi-Layer Perceptron as the decoder. Due to limitations on computational resources, we adopt a relatively small language model - the open-sourced GPT-2 model by OpenAI (Radford et al., 2019) with 124M parameters as the LLM encoders. Note that only the LLM encoders for the querying prompt-response pair participate in training.

Hyperparameters and machines The hyperparameters used for the VAE-based model are reported in Table 1. All experiments are conducted on one machine with 1 A6000 GPU.

Baselines Apart from the VAE-based conditional reward model, we also include the following baselines for comparison.

- Few-shot prompting: In Section 3.2, we evaluate the few-shot prompting method using the average accuracy on 100 users with different preference weights across the four aspects, either generated randomly or clustered. In the current experimental setting, we only consider two users. To directly compare the few-shot prompting method against others, one shall also evaluate on the new test sets described above. However, as mentioned in Section 3.4.1, the GPT-2 model suffers from a very small context window size, and thus is incapable of handling multiple preference demonstrations. Instead, we still adopt the Gemma-2b model (Team, 2024) as in Section 3.2, whose context length can cover up to 8 demonstrations. Since few-shot prompting only involves inference, the time consumption is acceptable.

- Non-conditional/traditional reward model: To verify the effectiveness of the conditional reward modeling formulation on personalized preference learning, we also train a GPT-2 model using the non-conditional/traditional formulation introduced in Section 2.1. No preference demonstrations are provided as user identifiers, and the training objective is to fit the mixed, aggregated, user-agnostic preference distribution.

3.5.2 Results and Limitations of Failed Training

Figure 6 and Table 2 illustrates the results of the few-shot prompting baselines with 0, 2, 4, 6, and 8 preference demonstrations, the traditional reward modeling baseline, and the VAE-based conditional reward modeling. The accuracy refers to the mean prediction accuracy of the two users favoring "helpfulness" and "honesty", and std. refers to the standard deviation, computed across three runs with different random seeds.

Although using the Gemma-2b model with many more parameters, all the few-shot prompting

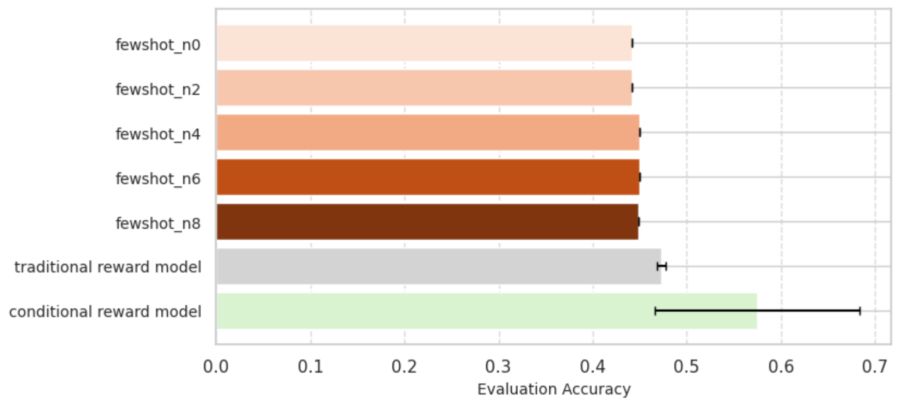


Figure 6: Test results of multiple methods on the personalized UltraFeedback dataset

Table 2: Numerical test results of multiple methods

Method	Fewshot n0	Fewshot n2	Fewshot n4	Fewshot n6	Fewshot n8
Accuracy±Std.	0.442±0.000	0.442±0.000	0.450±0.000	0.449±0.000	0.448±0.000

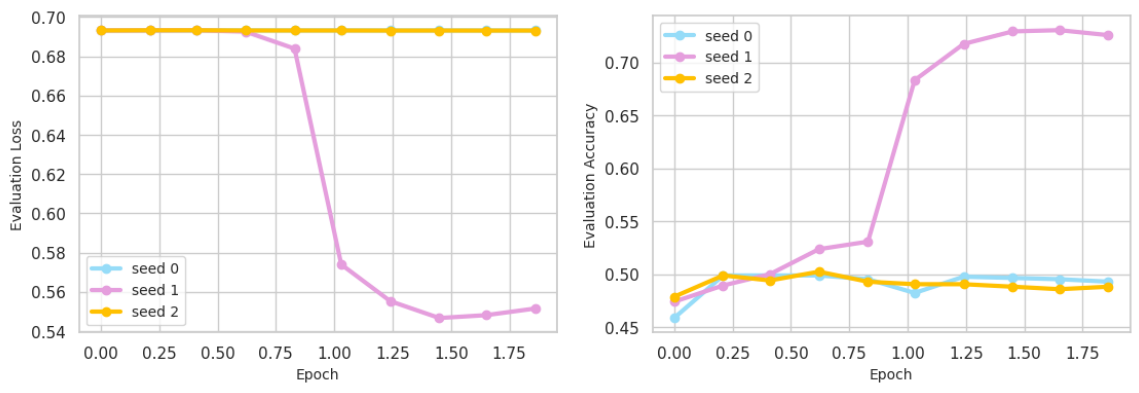
Method	Traditional reward model	Conditional reward model
Accuracy±Std.	0.473±0.005	**0.574±0.109**

baselines fall behind the two methods involving finetuning on the smaller GPT-2 model. This emphasizes the necessity of a learning process in the human preference alignment for language agents. Similarly to the results in Section 3.2, increasing the number of demonstrations in few-shot prompting does not improve the performance significantly, also reflecting that the model itself may not be able to correctly interpret the underlying user taste and use them to predict the preferred response, although such guidelines are included in the prompt.

The VAE-based conditional reward model greatly outperforms the traditional baseline. This superiority stems from the fundamentally different reward model specification - distinguishing, preserving, and learning diverse user preferences by conditioning the output on not only the querying triplet, but also the user identifiers.

However, as shown by the standard deviation in Table 2 and the error bar in Figure 6, the performance of the VAE-based conditional reward model is highly unstable. Careful examination reveals that, surprisingly, two out of the three runs of this method **failed to converge**, with evaluation accuracy stagnating below 0.5 throughout the training process. Metrics on the test set, including loss and accuracy, are plotted against the number of epochs trained, as shown in Figure 7. More sets of experiments using the same settings demonstrate similar results.

Although the VAE-based conditional reward model proves effective, the observed unstable training limits its reproducibility and practical applicability. In the next section, we analyze the training behaviour in detail and adopt various methods from different perspectives to resolve the issue, with a simple adaptive sampling strategy being the most helpful solution.



(a) Test loss (b) Test accuracy

Figure 7: Evaluation curves of the conditional reward model across three runs

4 Techniques for Stabilizing and Accelerating Training

4.1 Section Overview

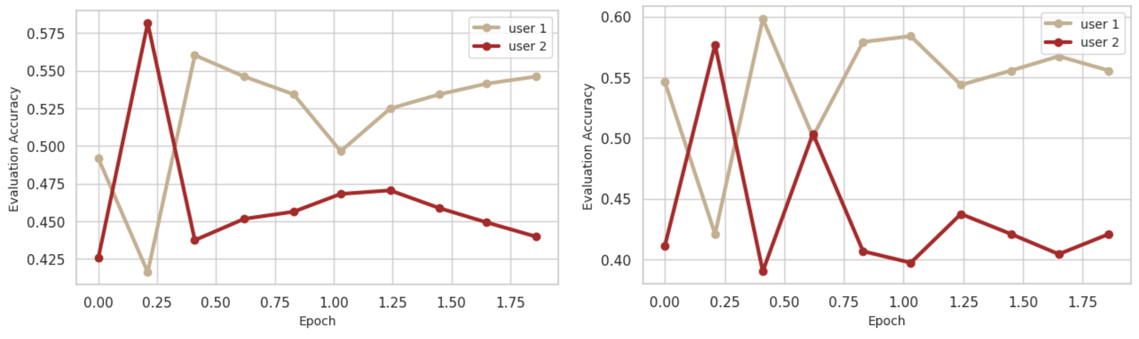
This section documents the explored methods aiming to resolve the training failure issue mentioned in Section 3.5, including gradient manipulation techniques - Multiple Gradient Descent Algorithm and Projecting Conflicting Gradients - from the Multi-Objective Optimization literature, an additional contrastive learning paradigm, and the adaptive sampling strategy. The first two approaches, although clearly motivated, did not yield desired outcomes. The simple adaptive sampling strategy eventually proves to stabilize and accelerate training effectively. To keep a smooth logic flow of this report, the working adaptive sampling strategy is described first in Section 4.2, followed by the other two attempts in Sections 4.3 and 4.4.

4.2 Adaptive Sampling and Improved Results

In Section 4.2.1, we first discuss the possible reason for the stagnation of the training process, followed by the specification on the adaptive sampling strategy in Section 4.2.2 improved experimental results, and analysis on how the strategy works in Section 4.2.3.

4.2.1 Conflicts Between Different Users

Recall that although on average, the VAE-based conditional reward model achieves an apparently higher accuracy compared to the traditional reward model due to improved model speci-



(a) Failed run with seed 0 (b) Failed run with seed 2

Figure 8: Evaluation accuracy of individual users in the two failed runs

fication recognizing diverse preferences, there is a large standard deviation among independent runs. This is because out of the three runs with different seeds, two of them failed to witness any decrease in the loss function. In previous works that aim to obtain an unconditional reward model achieving similar, fair performances across multiple social groups (Chakraborty et al., 2024; Ramesh et al., 2024), as mentioned in Section 2.2, conflicts among the different groups are the major bottleneck that needs to be properly resolved. Inspired by this insight, we suspect that the training stagnation problem is also caused by user conflicts.

To verify this guess, we examine the variation trend of the evaluation accuracy of individual users during training for the two failed runs. As shown in Figure 8, a clear adversarial relation is observed between the two users. At nearly every step, the accuracy of one user increases, while that of the other decreases. Potentially due to random batch sampling, where the proportion of data of the two users varies across steps, together with other stochastic factors, such as unstable gradient norms, the model focuses on different users alternatingly. As a consequence, only one user is improved for a small number of steps, and both users are seldom jointly considered, leading to a catastrophic training stagnation phenomenon.

Unlike the traditional reward model formulation, user conflicts in conditional reward modeling were expected to be inherently handled by the encoder that produces different latent embeddings for individual users, distinguishing their opposite preference selections. However, our observation reveals that such conflicts are still hard to alleviate. This may be because when the encoder is undertrained, it cannot map embeddings that are sufficiently distinguishable for the decoder. Hence, conflicts persist for the decoder and adversely affect the training of the encoder, forming a deadlock.

4.2.2 A Simple Adaptive Sampling Strategy

After investigating various possible solutions, a simple adaptive sampling strategy stands out as effective. On a high level, this strategy follows a continual learning mindset. The detailed description is as below.

1. Divide all users into three groups: to-be-trained (A), training (B), finished (C).
2. At each phase, select one user to group B and train for a fixed number of steps, during which draw samples proportionately from each user such that $N_A : N_B : N_C = 1 : 2 : 4$, where N_A, N_B, and N_C are the total number of samples from users in group A, B, C, respectively.
3. Put this user in group C; repeat steps 2 and 3 until all users are in group C.

Specifically, the adaptive sampling strategy divides the entire training into n phases, each consisting of a sufficient number of steps, where n is the number of users. At each phase, it focuses on a single user, i.e., the one in group B, to train both the encoder to learn his/her latent preference distribution and the decoder to adapt its prediction given the latent embeddings. At this phase, signals from users that have not been processed yet, i.e., those in group A, are restrained due to the smaller amount of data. On the other hand, signals from users that have already been processed and whose preferences have been learned earlier, i.e., those in group C, are amplified. Note that this amplification is designed deliberately because the gradient norms become relatively small after the learning tasks for these users converge, and a strong enough gradient signal is still required to prevent forgetting.

Intuitively, this strategy prevents the model from altering between different users by solving them sequentially. It also prevents the forgetting problem in continual learning, which refers to cases where the model degrades on earlier tasks as it fits new ones, by sampling more data, amplifying signals, and thus keeping the model activated for processed users.

It is worth pointing out the difference between sampling proportionately and gradient weighting. Indeed, attaching the corresponding weight to gradient vectors calculated from data of different users is theoretically equivalent to sampling proportionately. However, to obtain an individual gradient for each client requires a separate forward and backward pass on the entire model. In other words, n bi-directional passes are needed when there are n users. Given the

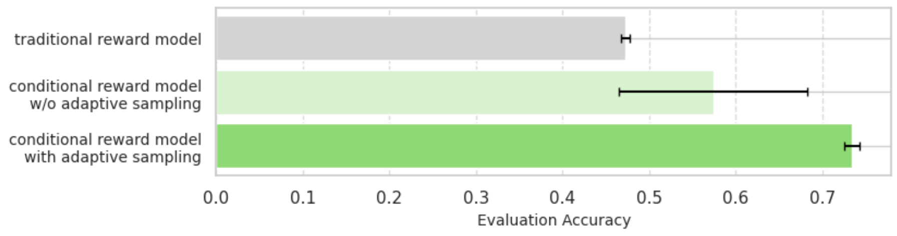


Figure 9: Evaluation accuracy of the conditional reward model with adaptive sampling

Table 3: Numerical test results of the conditional reward model with adaptive sampling

Method	Traditional reward model	Conditional reward model w/o adaptive sampling	Conditional reward model with adaptive sampling
Accuracy±Std.	0.473±0.005	0.574±0.109	**0.734±0.009**

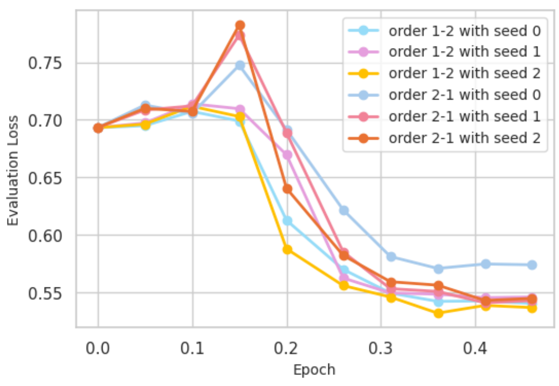
massive scale of language models, such operations are highly inefficient, and the training time would be drastically inflated. On the contrary, sampling proportionately only requires drawing examples from multiple user datasets according to a weighted probability distribution. Examples can still be combined into one batch and passed into the model as a whole. Here, a single gradient obtained from one forward and backward pass is sufficient, as weighting is already handled on the data level.

4.2.3 Improved Results and Analysis

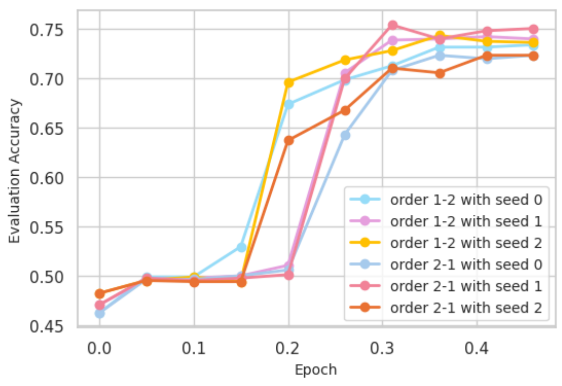
We now present the improved results after adopting the adaptive sampling strategy. Given the two simulated users, there are two possible sequential orders for moving them into the "Training" group B, i.e., 1-2 and 2-1. We experiment with both orders, each three times with different seeds, resulting in a total of six runs. We consider all six runs to calculate the metrics for the adaptive sampling strategy reported below, unless otherwise specified.

As shown in Figure 9 and Table 3, compared to the existing vanilla VAE-based conditional reward model (Poddar et al., 2024), our adaptive sampling strategy greatly increases the average accuracy and decreases the standard deviation across multiple runs. Figure 10 verifies that all six independent runs are successful with good loss and accuracy curves.

Notably, our strategy also significantly accelerates convergence and reduces the number of training epochs needed. Instead of 2 epochs, it now takes only 0.25 epochs for the model to converge, with curves for comparison plotted together in Figure 11 and numerical training times

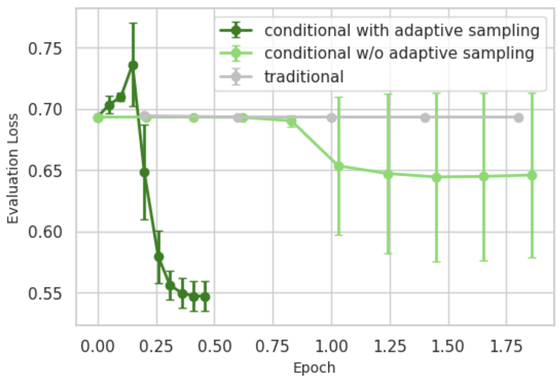


(a) Test loss

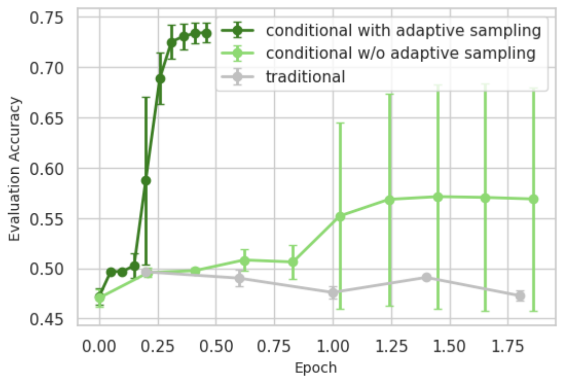


(b) Test accuracy

Figure 10: Evaluation curves of the conditional reward model with adaptive sampling



(a) Test loss



(b) Test accuracy

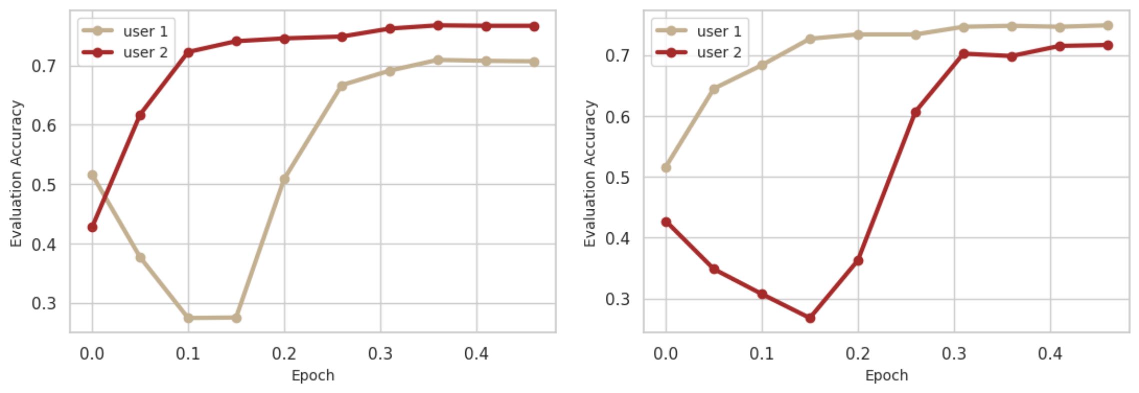
Figure 11: Comparison of convergence speed

listed in Table 4. This again verifies that our strategy prevents the model from "pondering", and the effectiveness of the continual learning-style design.

Finally, we examine whether the adversarial trend in the evaluation accuracies of individual clients is alleviated. Figure 12 demonstrates that, as expected, the accuracy of the user who is selected to be first focused on smoothly increases without fluctuation. When the training enters the second phase, the accuracy of the other user is also improved, while that of the previous client stays high without degrading. This provides an intuitive visualization of how the adaptive

Table 4: Training epochs and times

Method	Traditional reward model	Conditional reward model w/o adaptive sampling	Conditional reward model with adaptive sampling
Number of Epochs Needed	2	2	0.25
Avg. Training Time	2h 32m 22s	4h 24m 16s	1h 9m 16s



(a) Order 1-2 (b) Order 2-1

Figure 12: Evaluation accuracy of individual users with adaptive sampling

sampling strategy works. Results in Figure 12a and Figure 12b are averaged across three seeds for the corresponding training order.

4.3 Attempt: Multi-Objective Gradient Manipulation

As plotted in Figure 8, the evaluation accuracies of different users often change in opposite directions at each update step, which is possibly caused by multiple gradient vectors conflicting with each other. Conflicting gradients have been a prevalent challenge in the field of Multi-Objective Optimization (MOO), and several algorithms have been developed to resolve it through gradient manipulation. In this section, we experiment with these methods. Section 4.3.1 provides an overview of MOO, Section 4.3.2 specifies the two algorithms we apply - Multiple Gradient Descent Algorithm (MGDA) and Projecting Conflicting Gradients (PCGrad), and Section 4.3.3 reports the results.

4.3.1 Introduction to Multi-Objective Optimization

MOO aims to learn under multiple, potentially conflicting, objectives. Many practical problems in various domains, such as federated learning (McMahan et al., 2017) and drug design (Xie et al., 2021), can all be formulated as an MOO task. For example, clients participating in federated learning often possess heterogeneous local data, and multiple criteria must be satisfied when designing new medicines. These objectives often conflict with each other, making it impossible to find one solution that is better than the others in all objectives. Instead, MOO methods aim to locate solutions on the Pareto Front, which is the set of Pareto optimal solutions that are not dominated by any other (Miettinen, 1999). In other words, on the Pareto Front, any improvement in one objective requires degradation in another. Finding Pareto optimal solutions is not the goal in our problem, and therefore, we do not elaborate on the detailed concepts here and refer interested readers to some recent papers (Liu et al., 2024; Chen et al., 2025).

When dealing with conflicting objectives, conflicting gradients naturally emerge during training. One important line of MOO methods is the gradient manipulation methods, which aim to resolve such conflicts and find a common descent direction for all objectives. Given the adversarial trend between the two users in our experiments, perhaps finding a common update gradient using MOO methods helps.

4.3.2 MGDA and PCGrad

We experiment with two gradient manipulation methods, namely Multiple Gradient Descent Algorithm (MGDA) (Désidéri, 2012) and Projecting Conflicting Gradients (PCGrad) (Yu et al., 2020). At each step, MGDA computes the update vector $\mathbf{d}$ that optimizes the minimal improvement among all objectives:

$$\max_{\mathbf{d}} \min_{i \in \{1,\ldots,m\}} (f_i(\boldsymbol{\theta}) - f_i(\boldsymbol{\theta} - \eta \mathbf{d})), \tag{9}$$

where m is the number of objectives, $f_i(\cdot)$'s are the objective functions, θ is the optimization variable (e.g., model parameters), and η is the step size. The dual problem of this task is to find a convex combination of all gradients that has the minimum norm, namely

$$\min_{\boldsymbol{\alpha} \in \Delta_m} \| \sum_{i=1}^{m} \alpha_i \boldsymbol{g}_i \|_2^2 \tag{10}$$

where Δ_m is the $(m-1)$-dimensional simplex and $\boldsymbol{g}_i$ is the gradient of the i-th objective, i.e., $\boldsymbol{g}_i = \nabla_{\boldsymbol{\theta}} f_i(\boldsymbol{\theta})$. This problem is usually solved by the Frank-Wolfe algorithm (Jaggi, 2013).

While MGDA is widely used to resolve gradient conflicts, PCGrad appears more intuitive as it explicitly eliminates components in gradients that conflict with others. PGCrad considers two gradients $\boldsymbol{g}_1$ and $\boldsymbol{g}_2$ "conflicting" if their inner product is negative. For two conflicting gradients, it corrects each of them by projecting them onto the normal plane of the other, reducing

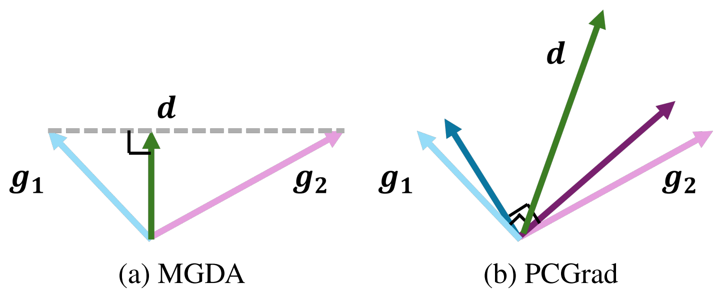


Figure 13: Illustration of MGDA and PCGrad

forces in opposite directions. For instance, the corrected $\boldsymbol{g}_1$ with respect to $\boldsymbol{g}_2$ is

$$\hat{\boldsymbol{g}}_1 = \boldsymbol{g}_1 - \frac{\boldsymbol{g}_1^T \boldsymbol{g}_2}{||\boldsymbol{g}_2||_2^2} \boldsymbol{g}_2 \tag{11}$$

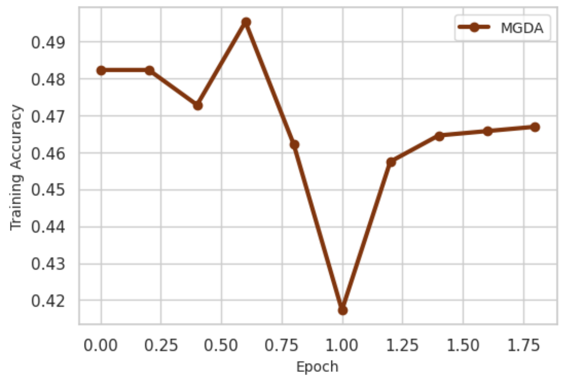
The final composite gradient is the summation of all corrected sub-gradients, i.e., $\mathbf{d} = \sum_{i=1}^{m} \hat{\boldsymbol{g}}_i$. Figure 13 illustrates how the update direction of MGDA and PCGrad is computed.

4.3.3 Results and Analysis

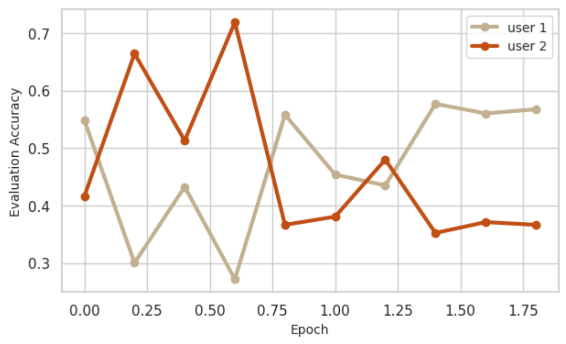
In our experiments, we treat the losses calculated from the data of different users as objectives. At each step, we pass the data of different users into the model separately to obtain individual gradients of each objective, which are then combined into a composite update direction by applying MGDA or PCGrad. After mitigating gradient conflicts, we expected the performance for both users to be simultaneously improved most of the time. However, as shown in Figure 14, the previously observed adversarial trend persists, and the training process still fails to converge. We suppose the reason lies in the probabilistic nature of VAE, where a sampling step is performed when generating user embeddings from the latent space. Given the non-deterministic behaviour, gradient surgery may not be effective. Moreover, obtaining individual gradients requires separate forward and backward passes for data from different users, significantly inducing computational overhead, which is also an advantage of the adaptive sampling method as mentioned in Section 4.2.

4.4 Attempt: Contrastive Learning

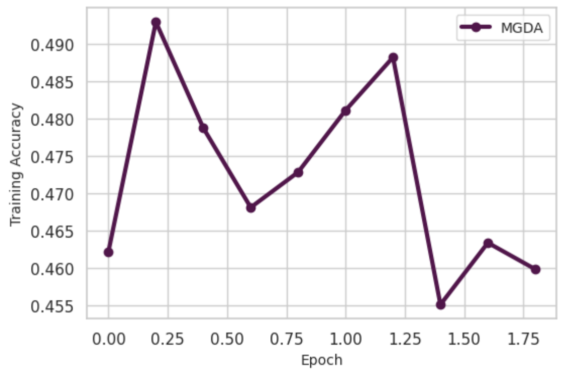
This section reports a contrastive learning method we tried that aims to resolve the training stagnation issue from another perspective. We discuss the motivation and methodology in Sec-



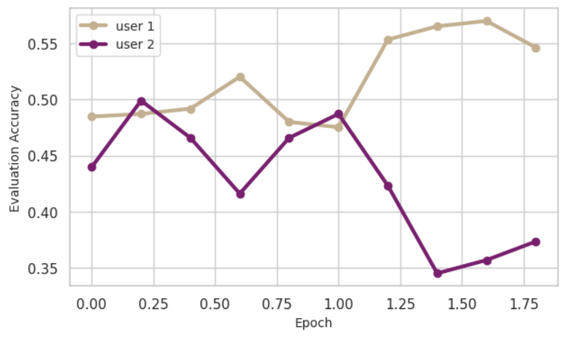
(a) Evaluation accuracy of MGDA



(b) Individual accuracies of MGDA



(c) Evaluation accuracy of PCGrad



(d) Individual accuracies of PCGrad

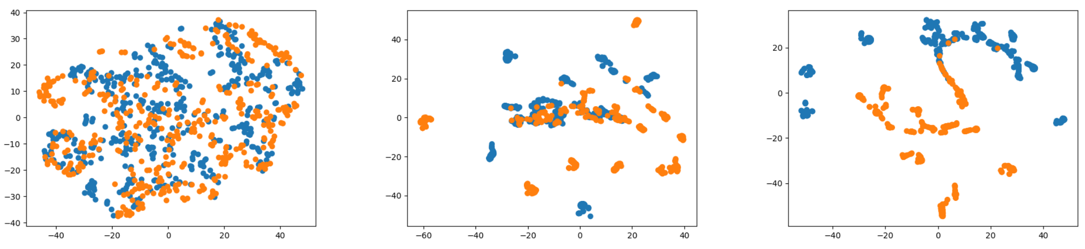
Figure 14: Results using MGDA and PCGrad

tion 4.4.1, and present the results in Section 4.4.2. Although the performance of this approach is not satisfactory, we still report our exploration for completeness.

4.4.1 Motivation and Contrastive Loss

As discussed in Section 4.2.1, one possible reason for some runs to fail in converging might come from an inferior encoder. Specifically, when the encoder is undertrained, it is incapable of inferring the underlying user preference from the given demonstrations and generating effective latent embeddings for different users. As a result, the latent user variables passed into the decoder are hard to distinguish, and hence, conflicting preferences cannot be separately modeled as conditional distributions as designed, inducing the adversarial and fluctuating trend in the evaluation accuracy for individual clients.

Figure 15 serves as a shred of supporting evidence for this analysis. Using the t-SNE (Van der Maaten and Hinton, 2008) method, it visualizes the mean vectors of the latent space for the two users generated by the encoder during the only successful training. Each point corresponds to



(a) Epoch 0.0 (start) (b) Epoch 1.0 (c) Epoch 2.0 (end)

Figure 15: t-SNE visualization of latent user embeddings in a successful run

the mean vector inferred from one set of preference demonstrations, and the color indicates the user identity. At the beginning, embeddings for different users are mixed, and towards the end, they are clearly separated into clusters. In Figure 15b, embeddings start to show a tendency of separation, and the corresponding epoch falls into the period when loss decreases drastically, as in Figure 7a. This correspondence verifies a strong relation between the distinguishability of latent embeddings and the model performance.

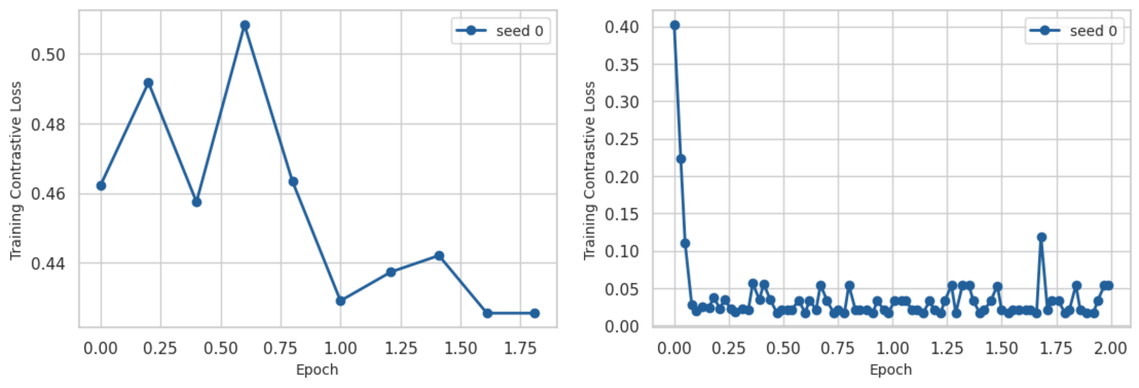
In the above analysis, whether the encoder can produce distinguishable user embeddings seems to be a key issue. One possible method to stimulate this is to use an additional contrastive loss, which encourages latent embeddings of the same user to be more similar and those of different users to be more different. In other words, the contrastive loss promotes a more clustered latent space output by the encoder, which would help the decoder to tell diverse preferences from individual clients apart. We adopt a contrastive loss based on cosine similarity. Specifically, on all samples $\left\{\left(h, \{(x^h, y_w^h, y_l^h)_c\}_{k=1}^N, (x^h, y_w^h, y_l^h)\right)\right\}_{i=1}^B$ in each batch, we compute

$$\mathrm{CL} = \sum_{i,j\in\mathcal{H}} -\mathbf{1}(h_i = h_j)\,\mathrm{CosSim}(\mu_i, \mu_j) + \mathbf{1}(h_i \neq h_j)\,\mathrm{CosSim}(\mu_i, \mu_j) \tag{12}$$

where B is the batch size, $\mathbf{1}(\cdot)$ is the indicator function, h_i and h_j denote the user identity of samples i and j, μ_i and μ_j are their mean vectors output by the encoder, and $\mathrm{CosSim}(\cdot,\cdot)$ refers to the cosine similarity function. This term can be directly added to the original VAE loss in practice with a proper coefficient to control the tradeoff among other objective terms.

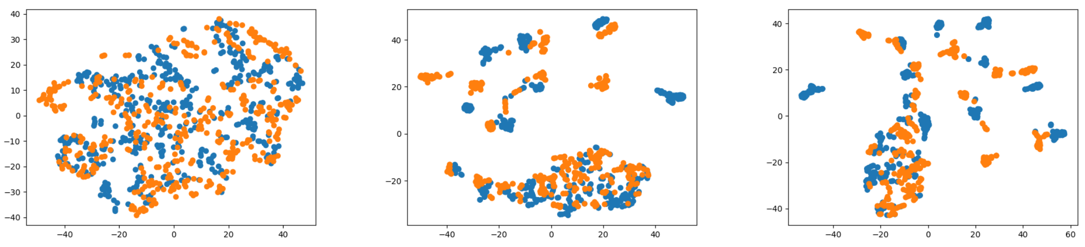
4.4.2 Results and Analysis

We experiment with the contrastive loss using one of the seeds on which the vanilla VAE-based conditional reward model fails to converge. A coefficient of 0.5 is adopted. To increase the



(a) Evaluation accuracy (b) Training contrastive loss

Figure 16: Training contrastive loss and evaluation accuracy curves



(a) Epoch 0.0 (start) (b) Epoch 0.1 (c) Step 2.0 (end)

Figure 17: t-SNE visualization of latent user embeddings with contrastive loss

number of samples in each batch and thus make the computed contrastive loss more informative, we use a larger per-device batch size of 8 and a gradient accumulation step of 4, still resulting in an effective batch size of 32.

Figure 16 plots the evaluation accuracy curve and the training contrastive loss curve. Although the contrastive loss quickly decreases to a small value, it converges and starts to oscillate, and the training still fails with nearly no improvement in the evaluation accuracy. We also examine the t-SNE visualization of latent embeddings during training, as in Figure 17. However, the effectiveness of this mechanism in stimulating the encoder to produce clustered embeddings appears very limited. After the contrastive loss converges, points corresponding to different users are only shifted apart to a small extent. Given the accuracy performance, such shifts still seem insufficient for the decoder to distinguish different users.

5 Prompting Larger Models with Fine-Grained User Stories

5.1 Section Overview

In previous sections, only preference demonstrations are used as user identifications for the model to infer the underlying user taste. Beyond this, user preferences are also highly correlated with many sociological attributes and factors, such as cultural background, life experiences, personality, occupation, etc. In this section, we explore the possibility of LLM personalization with fine-grained user stories. Unlike the few-shot prompting method with demonstrations in Section 3.2, we found that directly prompting a larger language model with such stories simulates preferences that reflect reasonable structures related to user occupation. Section 5.2 introduces the experimental setups, Section 5.3 elaborates the technique for mitigating positional bias, which refers to the phenomenon of LLMs overly choosing the first option, and finally, in Section 5.4, results, analysis, and interpretation are presented.

5.2 Experimental Setup

User stories We use 36 synthetic user stories that document in detail a client's profile, including name, age, race, current social position, family situation, interests, hobbies, and personality. We label the users according to their position for the convenience of further analysis. A sample profile is as below.

> Sociology college student: Imagine that you are Klaus Mueller, a 20-year-old student at Oak Hill College, studying sociology. You are passionate about social justice and exploring different perspectives. You have a strong sense of empathy, allowing you to connect with individuals from diverse backgrounds. Your analytical skills shine not only in your academic work but also in these extracurricular activities, where you help organize community events and awareness campaigns. For the sociology research paper, you have chosen to explore the effects of gentrification, focusing on how it impacts community cohesion and access to essential services in low-income neighborhoods. You plan to incorporate firsthand interviews and community surveys to add depth to the analysis.

Questions and models We still adopt the binarized UltraFeedback dataset (Cui et al., 2023) as our source for extracting the prompt-responses triplets for querying the model. To reduce the computational overhead, only 1000 randomly sampled entries are considered. We use the open-source LLM Llama-3 with 8B parameters provided by Meta (Grattafiori et al., 2024) as a reward model. Specifically, user stories are injected as the system prompt, querying triplets are appended in the user prompt, and the model is asked to predict which of the two responses to the given question the given user is more likely to prefer. However, we find that letting the model directly choose from two options leads to severe bias towards the first one, even if we randomly swap the order. In the next subsection, we analyze this phenomenon in detail and provide a scoring technique to resolve it.

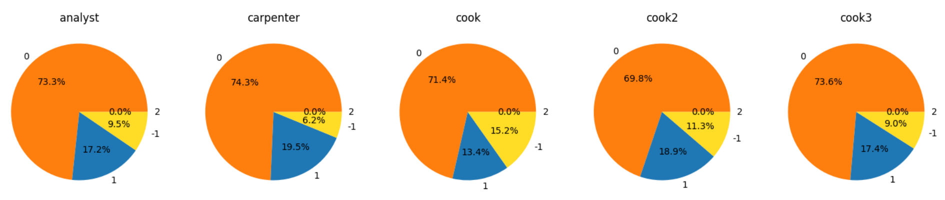
5.3 Mitigating Positional Bias

When using profile stories as user identifications and requesting the model to directly choose from the two given candidate responses, a salient positional bias is observed, which refers to the scenario where LLMs lean towards the option that is in a certain position. In our case, the first option is overfrequently preferred. The bias persisted even if we used common prompting techniques, such as chain-of-thought and explicit requirement, as listed below.

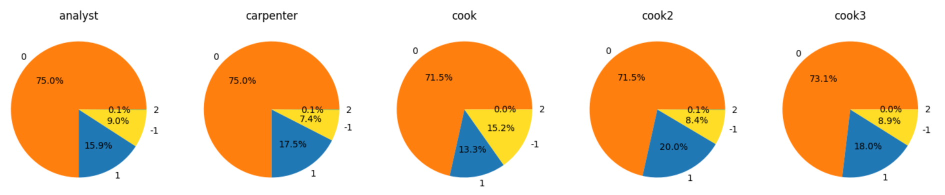
- Chain-of-thought: Please begin your evaluation by "<My analysis>" (without quotes) and provide a short analysis on the two responses.
- Explicit guideline: Ensure that the order in which the responses were presented does not influence your decision.

5.3.1 Inconsistency in User Choices and Homogeneous Preferences

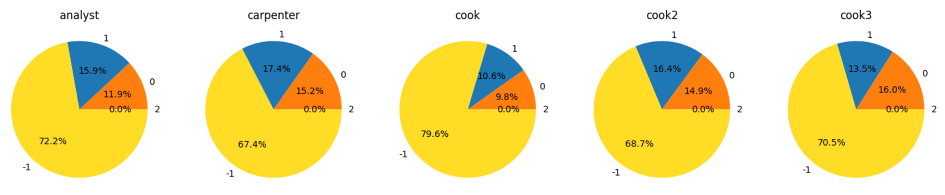
Figure 18 shows the percentage of the preferred response of some sampled users, where 0 indicates a preference for the first option, 1 for the second option, and 2 and -1 indicate invalid answers due to both or none of the options mentioned in the model response. The complete figures are attached in Appendix B. Figure 18a and Figure 18b denote two independent runs with the order of responses in all queries reversed. Note that in both runs, the responses are shuffled so that the first option is not always the preferred one in the original dataset. One can see that in both subfigures, the first option is preferred significantly more than the second, indicating that after swapping the positions of the two responses, most users also switched their choice to the opposite. Figure 18c plots the results combining these two runs, and cases where the model responses are not consistent are counted towards the -1 category. Again, a majority of choices are invalid due to inconsistency.



(a) Random order of options



(b) Above order reversed



(c) Two orders combined

Figure 18: Per user preference distribution (partial)

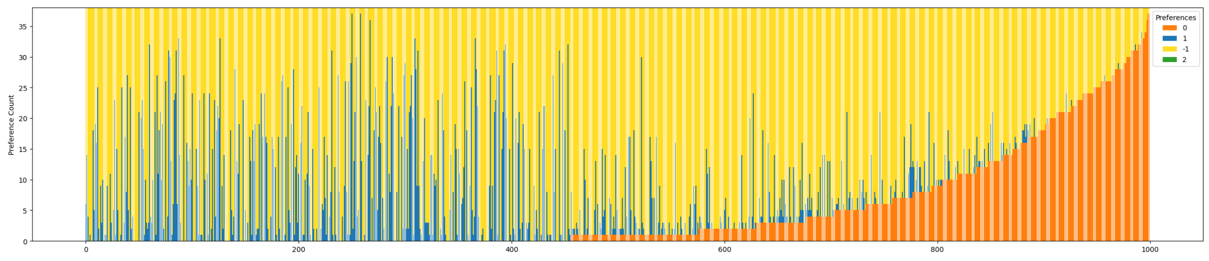
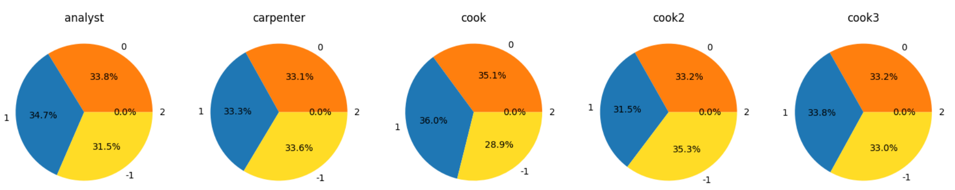


Figure 19: Per question preference distribution

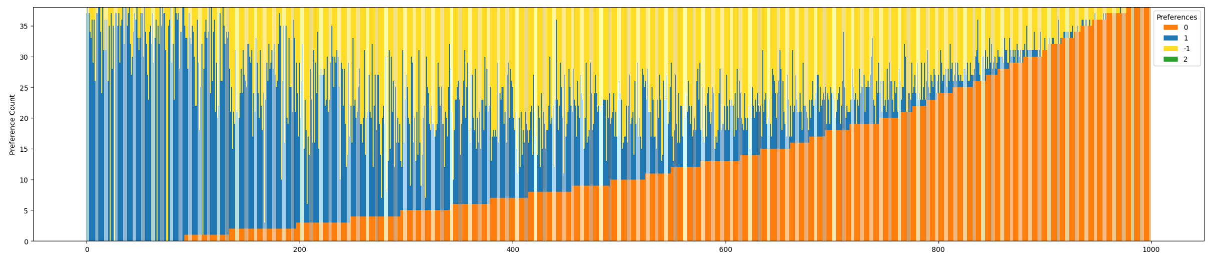
Moreover, even among the valid choices, most users are simulated to have selected the same response for a certain prompt. Figure 19 shows the percentage of preference choices for each of the 1000 questions, with bars sorted in ascending order of the number of votes for the first and second response, respectively. It is clear that more than half of the questions receive universal answers, where the bar contains either blue or orange, but not both, excluding the yellow part for invalid answers. This indicates that the heterogeneity of user preferences may not have been sufficiently simulated.

5.3.2 Scoring Querying Mode

To overcome the issue of positional bias, we introduce another querying mode using scoring. Instead of a direct pairwise comparison, we ask the model to rate a single response to a par-



(a) Per user preference distribution (two orders combined) (partial)



(b) Per question preference distribution

Figure 20: Simulation results with scoring mode

ticular prompt and select the response with a higher score to be the predicted preferred one. In fact, this mechanism is inspired by the original composition of the UltraFeedback dataset, where each response is also rated instead of directly compared.

Similarly to the UltraFeedback dataset, to guide to model to evaluate a response comprehensively and calibrate to the described user more accurately, we prompt the model to give subgrades on four aspects, namely truthfulness, instruction-following, helpfulness, and personal preference, with descriptions of each dimension included in the prompt for clarity. Each dimension will be assigned a score of -2, -1, 0, 1, or 2, where negative scores are more intuitive in expressing incorrectness, irrelevance, or personal dislike, and 0 represents a neutral perspective. The descriptions and complete prompts are attached in Appendix A.

It is noteworthy that we exclude the "honesty" dimension used in the UltraFeedback dataset. For one thing, "honesty" requests the model to be honest about its knowledge and express its uncertainty using weakeners, such as "I guess" and "perhaps". However, we observe that the LLM might follow this instruction too rigidly, penalizing responses simply because it does not include weakeners, even when it is not necessary. For another thing, when referring to the response being factually correct, "honesty" overlaps with "truthfulness". Therefore, we decided to exclude it from the evaluation criteria. In addition, we include a dimension of "personal preference", which explicitly requires the model to consider the given user profile when making the prediction.

The scoring querying mode successfully resolves the bias issue. Figure 20a illustrates the per-

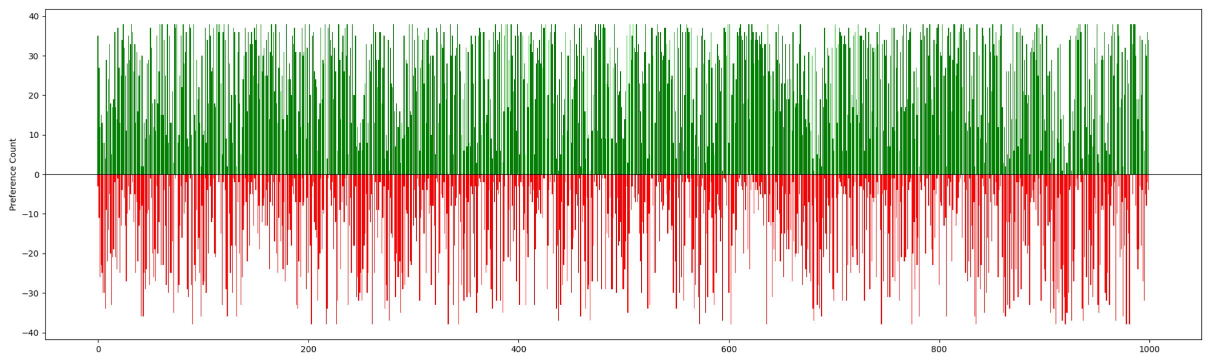


Figure 21: Frequencies of the responses being selected

centage of the selected option of the same sampled users after combining the two independent runs with reversed response positions. The frequencies of the two options being selected by each user are approximately the same, and the number of invalid model replies due to inconsistent choices or fatal responses is significantly reduced. Additionally, as shown in Figure 20b, the simulated preference is much more diverse, with most questions receiving different opinions from the population, demonstrating a sharp contrast to Figure 19.

In summary, the scoring mode greatly mitigates positional biases by improving the consistency of user choices given different orders of candidate responses and simulating heterogeneous preferences that are more similar to real-world scenarios. In the next subsection, all results are derived from experiments using the scoring mode.

5.4 Results and Interpretation

This subsection presents the analysis and interpretation of the simulated preferences. In particular, we found that users with related occupations show similar preferences.

5.4.1 Diverse Preferences of Different Users

The first question we investigate is whether the chosen response in the original dataset, which is labeled by a single annotator, is still preferred in our simulation, where multiple users are involved. Figure 21 shows the number of users choosing the originally preferred and rejected responses, respectively, for all the 1000 questions. Users counted towards the green bar select the originally preferred one, while those in the red bars select the opposite. It is apparent that although the former is slightly larger in number, there are still quite many users in the

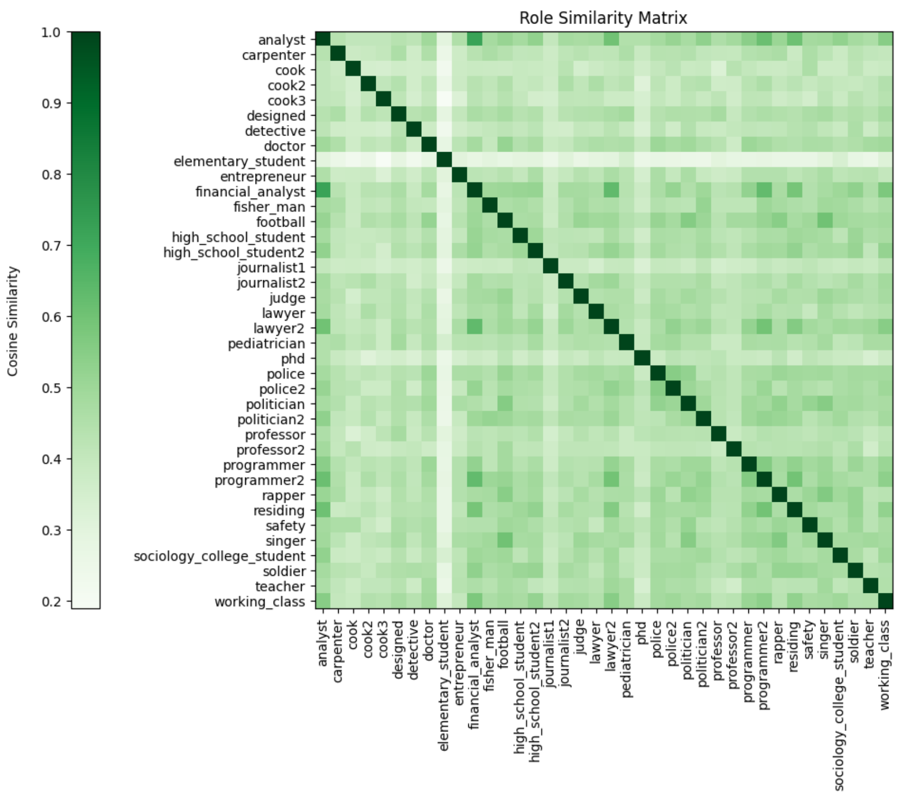


Figure 22: Pairwise similarity of user preferences

latter. The preferences of these users are underrepresented in the original dataset and thus not modeled in the RLHF. Through this simulation, we re-emphasize that human preferences are highly heterogeneous, which should be considered in the LLM alignment process.

5.4.2 Similar Preferences among Similar Occupations

For each user, their preferences for all questions are integrated into a one-dimensional vector. By calculating pair-wise cosine similarities of such vectors, a similarity matrix is obtained, as illustrated in Figure 22. An entry with a darker color represents higher similarity. One can observe that the rows and columns for "elementary student" are consistently light, indicating low similarity with all other professional occupations. This highly aligns with our intuition that young children process information and make decisions significantly differently from adults.

By linking each user with his or her mutual top five similar neighbors, we construct connected graphs to further reflect the structure. In other words, two users are connected if and only if they both ranked top five in each other's similarity list. These connected graphs, as shown in Figure 23, reveal some expected structures. For instance, singers and rappers, safety positions

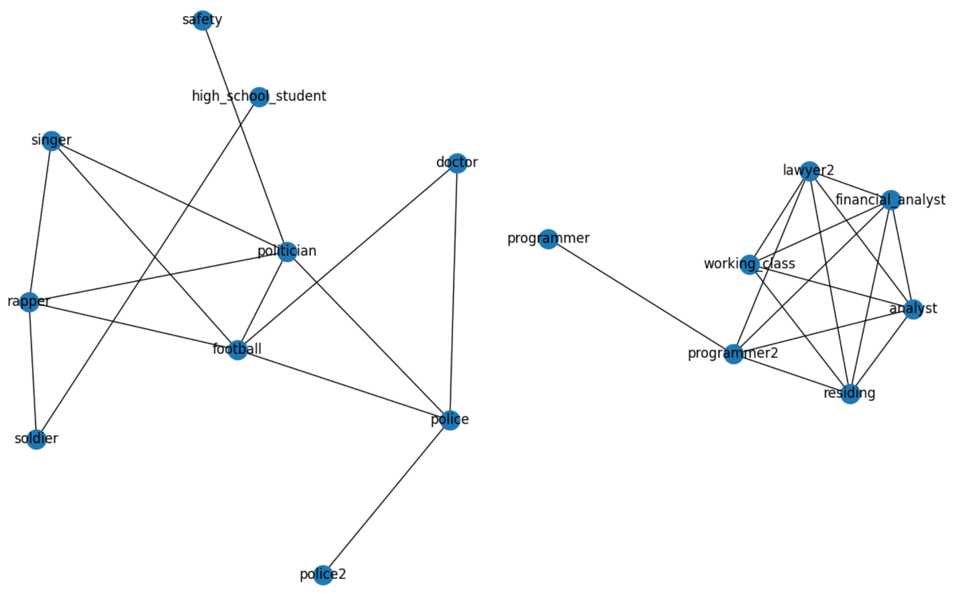


Figure 23: Connected graphs of mutual top five similar users

and politicians, two policemen, and two programmers share highly similar tastes.

5.4.3 Discussion

The above findings reflect the correlation between preferences and user occupations, and also verify the rationality and validity of the simulation results, which are obtained by simply prompting a relatively larger LLM with fine-grained user stories without any parameter fine-tuning. Nevertheless, obtaining user stories may be difficult under privacy considerations. On the other hand, providing preference demonstrations, as in experiments reported in previous sections, requires far less sensitive data.

6 Conclusion, Limitation, and Future Work

This project investigates methodologies for personalizing LLMs to diverse user preferences. We propose a conditional reward modelling framework that serves as the foundation for preserving and adapting to diverse preferences. We show that compared to the training-free few-shot prompting method and traditional reward modeling, a VAE-based structure is effective in realizing this formulation but suffers from failed training. The adaptive sampling strategy we developed resolves this issue, resulting in successful training across all runs and significantly accelerating convergence. While not working in practice, we also experimented with other methods, including gradient manipulation techniques and contrastive learning. Finally, we find that prompting larger models with fine-grained user stories simulates diverse and meaningful preferences with reasonable structures related to user occupations detected.

Several limitations merit consideration and future work. First, our experiments were constrained to a relatively small number of simulated users (two) due to computational resource limitations, and the scalability of our findings to more complex preference landscapes requires further verification. Second, the reward modeling stage addresses only the first phase of the RLHF pipeline; a complete implementation would require policy training, yielding LLMs capable of generating personalized content given user identifiers. Additionally, investigating efficient user identification mechanisms that balance informativeness (e.g., personal stories) and privacy (e.g., preference demonstrations) could enhance practical deployability.

This project contributes to the personalization of LLMs, advancing the goal of creating more adaptive and inclusive AI systems that cater to diverse human preferences. By addressing the heterogeneity of user tastes, this work takes an important step toward ensuring equitable access to high-quality AI tools for users across different backgrounds and preferences.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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

Louis Castricato, Nathan Lile, Rafael Rafailov, Jan-Philipp Fränken, and Chelsea Finn. Persona: A reproducible testbed for pluralistic alignment. *arXiv preprint arXiv:2407.17387*, 2024.

Souradip Chakraborty, Jiahao Qiu, Hui Yuan, Alec Koppel, Furong Huang, Dinesh Manocha, Amrit Singh Bedi, and Mengdi Wang. Maxmin-rlhf: Towards equitable alignment of large language models with diverse human preferences. *arXiv preprint arXiv:2402.08925*, 2024.

Yupeng Chang, Xu Wang, Jindong Wang, Yuan Wu, Linyi Yang, Kaijie Zhu, Hao Chen, Xiaoyuan Yi, Cunxiang Wang, Yidong Wang, et al. A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45, 2024.

Weiyu Chen, Xiaoyuan Zhang, Baijiong Lin, Xi Lin, Han Zhao, Qingfu Zhang, and James T Kwok. Gradient-based multi-objective deep learning: Algorithms, theories, applications, and beyond. *arXiv preprint arXiv:2501.10945*, 2025.

Ganqu Cui, Lifan Yuan, Ning Ding, Guanming Yao, Wei Zhu, Yuan Ni, Guotong Xie, Zhiyuan Liu, and Maosong Sun. Ultrafeedback: Boosting language models with high-quality feedback. 2023.

Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6):313–318, 2012.

Hanze Dong, Wei Xiong, Bo Pang, Haoxiang Wang, Han Zhao, Yingbo Zhou, Nan Jiang, Doyen Sahoo, Caiming Xiong, and Tong Zhang. Rlhf workflow: From reward modeling to online rlhf. *arXiv preprint arXiv:2405.07863*, 2024.

Esin Durmus, Karina Nguyen, Thomas I Liao, Nicholas Schiefer, Amanda Askell, Anton Bakhtin, Carol Chen, Zac Hatfield-Dodds, Danny Hernandez, Nicholas Joseph, et al. Towards measuring the representation of subjective global opinions in language models. *arXiv preprint arXiv:2306.16388*, 2023.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

Martin Jaggi. Revisiting frank-wolfe: Projection-free sparse convex optimization. In *International conference on machine learning*, pages 427–435. PMLR, 2013.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Meitong Liu, Xiaoyuan Zhang, Chulin Xie, Kate Donahue, and Han Zhao. Online mirror descent for tchebycheff scalarization in multi-objective optimization. *arXiv preprint arXiv:2410.21764*, 2024.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

Kaisa Miettinen. *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media, 1999.

Michel Minoux. Accelerated greedy algorithms for maximizing submodular set functions. In *Optimization Techniques: Proceedings of the 8th IFIP Conference on Optimization Techniques Würzburg, September 5–9, 1977*, pages 234–243. Springer, 2005.

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

Sriyash Poddar, Yanming Wan, Hamish Ivison, Abhishek Gupta, and Natasha Jaques. Personalizing reinforcement learning from human feedback with variational preference learning. *arXiv preprint arXiv:2408.10075*, 2024.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

Shyam Sundhar Ramesh, Yifan Hu, Iason Chaimalas, Viraj Mehta, Pier Giuseppe Sessa, Haitham Bou Ammar, and Ilija Bogunovic. Group robust preference optimization in reward-free rlhf. *arXiv preprint arXiv:2405.20304*, 2024.

Shibani Santurkar, Esin Durmus, Faisal Ladhak, Cinoo Lee, Percy Liang, and Tatsunori Hashimoto. Whose opinions do language models reflect? In *International Conference on Machine Learning*, pages 29971–30004. PMLR, 2023.

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

Gemma Team. Gemma. 2024. doi: 10.34740/KAGGLE/M/3301. URL `https://www.kaggle.com/m/3301`.

Marcel Torne Villasevil, Balsells I Pamies, Zihan Wang, Samedh Desai, Tao Chen, Pulkit Agrawal, Abhishek Gupta, et al. Breadcrumbs to the goal: goal-conditioned exploration from human-in-the-loop feedback. *Advances in Neural Information Processing Systems*, 36: 63222–63258, 2023.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Yiheng Wang. Large language models evaluate machine translation via polishing. In *Proceedings of the 2023 6th International Conference on Algorithms, Computing and Artificial Intelligence*, pages 158–163, 2023.

Yutong Xie, Chence Shi, Hao Zhou, Yuwei Yang, Weinan Zhang, Yong Yu, and Lei Li. Mars: Markov molecular sampling for multi-objective drug discovery. *arXiv preprint arXiv:2103.10432*, 2021.

Shunyu Yao, Howard Chen, John Yang, and Karthik Narasimhan. Webshop: Towards scalable real-world web interaction with grounded language agents. *Advances in Neural Information Processing Systems*, 35:20744–20757, 2022.

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *Advances in neural information processing systems*, 33:5824–5836, 2020.

A Sample Prompts

A Few-shot prompting prompts

We provide an example prompt used in the few-shot prompting method with preference demonstrations in Section 3.2 as below.

```
[Context]
[My preference] Here are several prompt-response examples reflecting my preferred
type of responses. You should learn from these examples and curate your response
to my preference.
<Example 1>
[Prompt] (omitted)
[Better response] (omitted)
[Worse response] (omitted)
(7 examples omitted)
Remember: your response to the following task should cater to my demonstrated
preference.
Describe a memorable holiday you had.
[RESPONSE A]
I'm just an AI, I don't have personal experiences or memories, but I can certainly
help you plan a memorable holiday! Have you considered a tropical island get-
away? (omitted)
[RESPONSE B]
My most memorable holiday was a two-week road trip across the United States
with my closest friends from college. We called ourselves the "Traveling
Troubadours" and embarked on an unforgettable journey from Boston to Los An-
geles, exploring the country's natural wonders, historic sites, and quirky roadside
attractions. (omitted)
Which one is better? A or B?
```

B Scoring criteria and roleplay prompts

The detailed descriptions of the four scoring dimensions and the prompt templates used in the roleplay prompting experiments in Section 5 are as below.

Hi, there! Thank you for helping us in the following task. Consider the following response provided by an AI assistant to a user instruction. Please rate the response based on your personal experience and preference on four dimensions: truthfulness, instruction-following, helpfulness, and personal preference. For each dimension, you can rate -2, -1, 0, 1, or 2. Here is the scoring standard for each dimension:
Truthfulness: The response should be faithful to factual knowledge as well as given contexts, never including any fictional facts that aren't true or cannot be grounded in the instruction. If you are not sure if the response is truthful, just rate 0.
Instruction-following: The response should conform to the user instructions. For example, if there are any instructions on the length, format, tone, or style of the response, the response should satisfy them.
Helpfulness: The response should be relevant and useful, either directly solving the user's doubt or offering information that can further help the user to proceed.
Personal preference: Consider whether there is anything in the response that is particularly to or against your personal taste.
You should start with a short analysis, and end with the final ratings strictly using this format: [[A, B, C, D]], where A, B, C, D are your ratings for truthfulness, instruction-following, helpfulness, and personal preference, respectively.
Here is the user instruction and the AI response:
[User Instruction] (omitted)
[AI response] (omitted)
Remember: If you are not sure of the truthfulness of the response, rate 0 for the dimension of truthfulness. Strictly follow the format [[A, B, C, D]] when stating your final ratings.

B Complete Version of Partial Figures

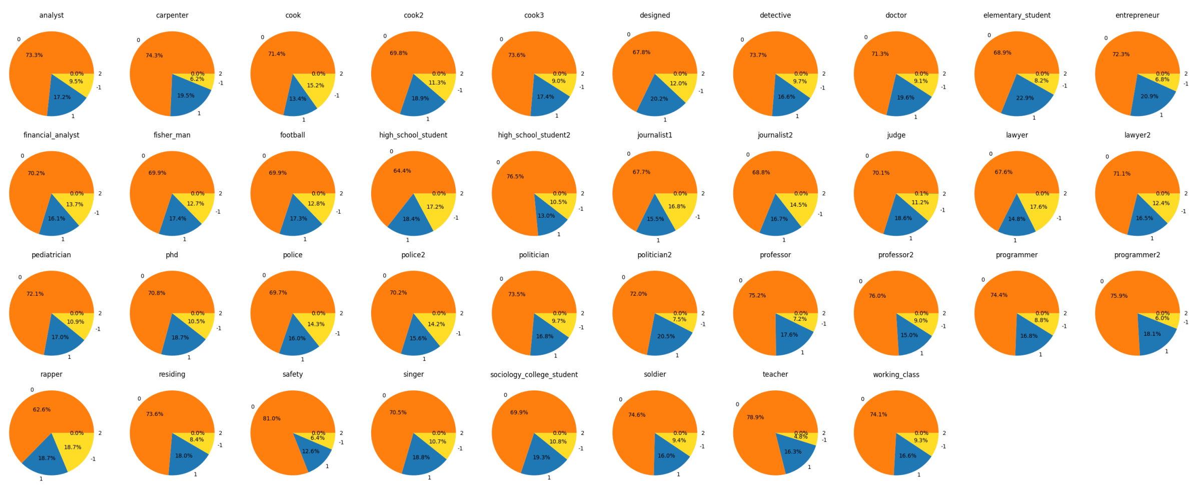


Figure 24: Full version of Figure 18a

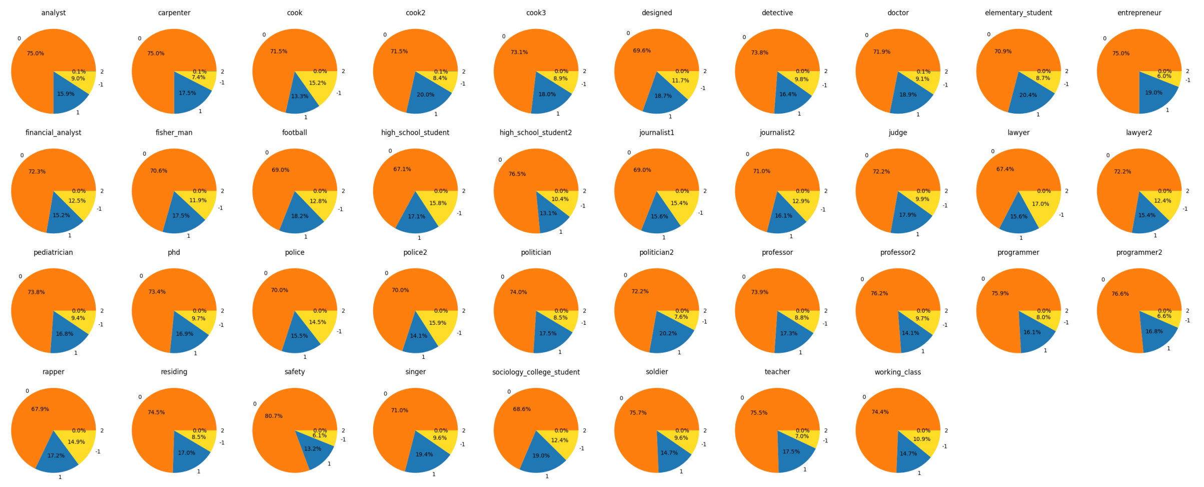


Figure 25: Full version of Figure 18b

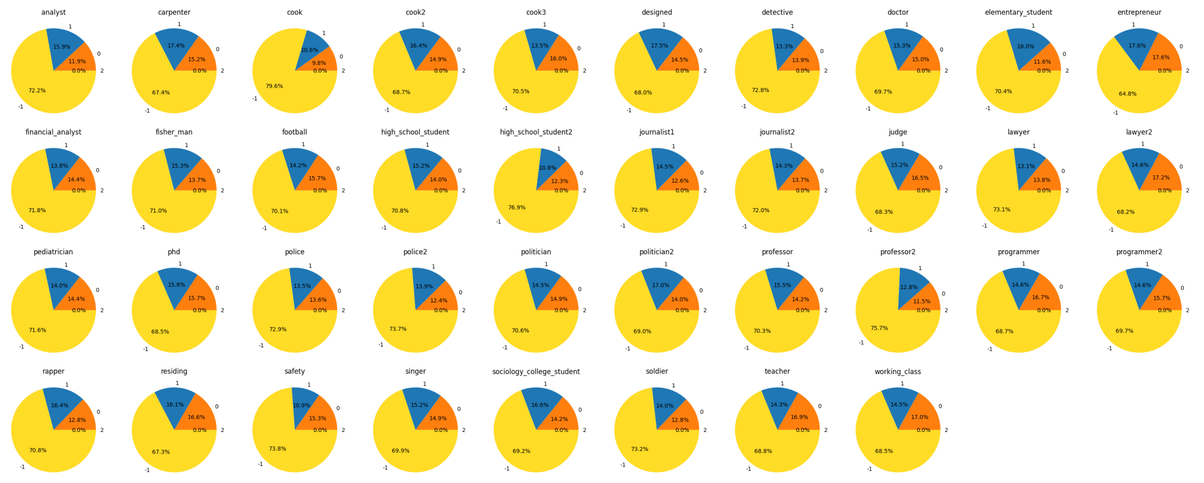


Figure 26: Full version of Figure 18c

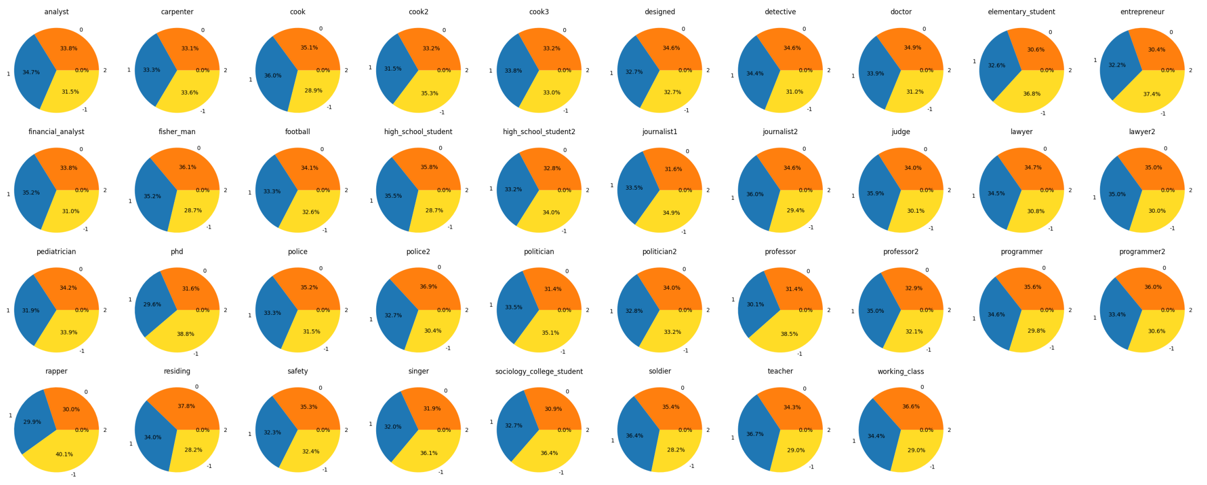


Figure 27: Full version of Figure 20a