Few-shot Steerable Alignment: Adapting Rewards and LLM Policies with Neural Processes

Katarzyna Kobalczyk [*1] **Claudio Fanconi** [*1] **Hao Sun** [1] **Mihaela van der Schaar** [1]

Abstract

As large language models (LLMs) become increasingly embedded in everyday applications, ensuring their alignment with the diverse preferences of individual users has become a critical challenge. Currently deployed approaches typically assume homogeneous user objectives and rely on single-objective fine-tuning. However, human preferences are inherently heterogeneous, influenced by various unobservable factors, leading to conflicting signals in preference data. Existing solutions addressing this diversity often require costly datasets labelled for specific objectives and involve training multiple reward models or LLM policies, which is computationally expensive and impractical. In this work, we present a novel framework for few-shot steerable alignment, where users' underlying preferences are inferred from a small sample of their choices. To achieve this, we extend the Bradley-Terry-Luce model to handle heterogeneous preferences with unobserved variability factors and propose its practical implementation for reward modelling and LLM fine-tuning. Thanks to our proposed approach of functional parameter-space conditioning, LLMs trained with our framework can be adapted to individual preferences at inference time, generating outputs over a continuum of behavioural modes. We empirically validate the effectiveness of methods, demonstrating their ability to capture and align with diverse human preferences in a data-efficient manner. Our code is made available at: https://github.com/kasia-kobalczyk/few-shot-steerable-alignment.

[*]Equal contribution [1]Department of Applied Mathematics and Theoretical Physics, University of Cambridge, Cambridge, United Kingdom. Correspondence to: Katarzyna Kobalczyk <knk25@cam.ac.uk>, Claudio Fanconi <caf83@cam.ac.uk>.

ICML 2025 Workshop on Models of Human Feedback for AI Alignment

1. Introduction

Motivation: The need for personalisation. As large language models (LLMs) become more integrated into daily life, it is critical that these models align with diverse human values and preferences. The majority of current methods for LLM alignment via Reinforcement Learning from Human Feedback (RLHF) (Christiano et al., 2017; Ouyang et al., 2022) or Direct Preference optimisation (DPO) (Rafailov et al., 2024), rely on single-objective fine-tuning, which assumes uniformity in human preferences. In reality, users of LLM-based systems often have diverse or even conflicting goals, influenced by factors like demographic and cultural backgrounds, cognitive processes, or implicit ambiguity in their prompts. Such user-specific factors are often unobservable, leading to collected preference data containing conflicting reward signals. When such data is aggregated under a homogeneous Bradley-Terry-Luce (BTL) model (Bradley & Terry, 1952), the result is equivalent to a Borda count (Siththaranjan et al., 2024), failing to capture the underlying diversity and learning to counterintuitive results.

The Challenge: Unknown sources of heterogeneity require new methods. To overcome the limitations of single-objective models, researchers have explored multi-objective alignment wherein the outputs of the LLM are guided by a weighted mixtures of reward functions (Guo et al., 2024b; Wang et al., 2024b; Ramé et al., 2023; Jang et al., 2023; Wang et al., 2024c). However, these methods typically require multiple datasets, each labelled with respect to a specific objective (e.g., honesty, helpfulness, or safety), and expect users to represent their preferences as a vector of weights corresponding to the predefined set of objectives. In cases where the sources of preference heterogeneity are unobservable, such approaches become impractical. Additionally, existing approaches require training multiple independent reward models or even multiple LLM policies (Chidambaram et al., 2024), one for each objective, which can be prohibitively expensive regarding computational resources and memory requirements.

Our goal: Few-shot steerable alignment. The aim of this paper is to address the following questions: Can we infer the underlying preferences of a user based on a small *few-shot* sample of their historical preference choices? Can we train

LLMs to align with these preferences at *inference time*? Our goal is to achieve this desired form of alignment, which we refer to as *few-shot steerable*, by only having access to a dataset of preference choices collected from multiple users acting according to diverse, but not explicitly stated objectives while avoiding the computational costs of having to train multiple independent reward or LLM policies.

Contributions: ▶ We extend the standard BTL model to account for the heterogeneous preferences across a population of users, enabling few-shot adaptation to individualised reward functions. ▶ By modelling user-level reward functions as samples from a stochastic process, we propose a practical implementation of this model based on Neural Processes (NPs) (Garnelo et al., 2018a;b). We demonstrate its applicability in both reward modelling (**NP-BTL**) and direct preference optimisation (**NP-DPO**). ▶ By introducing functional, parameter-space conditioning, NP-DPO opens up the possibility for training LLMs that can be adapted to preferences of individual users at inference time and can generate outputs across a continuum of behavioural modes, reflecting diverse human preferences without constraining the underlying factors of variability to a finite set of fixed, pre-defined objectives nor requiring training and storage of multiple models. ▶ We evaluate our propose models, demonstrating and carefully analysing their effectiveness in capturing and aligning with heterogeneous human preferences.

2. Background & Problem Definition

2.1. Background

Standard setup. Throughout this paper, we let $\mathcal{Y}$ denote the space of candidate options that users rank according to their preferences. For any pair of options $y_1, y_2 \in \mathcal{Y}$ we use the notation $y_1 \succ y_2$ to denote that y_1 is preferred over y_2. Conventional approaches to preference learning start by collecting a dataset of user preference choices $\mathcal{D}$, represented as tuples, $\mathcal{D} = \{(y_j^w, y_j^\ell)\}$ with $y_j^w, y_j^\ell \in \mathcal{Y}$ such that $y_j^w \succ y_j^\ell$. In the context of natural language modelling, $\mathcal{Y}$ is the space of natural language. Users express their preferences over pairs $y_1, y_2 \in \mathcal{Y}$, with $y_1 = [x, \tilde{y}_1]$, $y_2 = [x, \tilde{y}_2]$, where x is a prompt common for both options and $\tilde{y}_1, \tilde{y}_2$ two distinct completions sampled from an LLM, denoted by $\tilde{y}_1, \tilde{y}_2 \sim \pi_\theta(\cdot|x)$, where $\pi_\theta(\cdot|x)$ is the generative distribution of the LLM, aka the LLM policy.

A standard RLHF procedure starts by learning a reward function $r : \mathcal{Y} \to \mathbb{R}$ aiming to capture the utility of individual options. The higher the utility of a candidate option y, the more likely it is that it is preferred over the alternatives. After fitting the reward function to the dataset $\mathcal{D}$, it is then used to optimise the LLM policy π_θ via reinforcement learning. Alternatively, in DPO, these two steps can be combined into one optimisation step as (Rafailov et al., 2024).

RLHF under the BTL model. The BTL model (Bradley & Terry, 1952) is by now a standard approach to modelling user preferences, with the likelihood of $y_1 \succ y_2$ defined as:

$$\begin{aligned} p(y_1 \succ y_2) &= \sigma(r(y_1) - r(y_2)) \\ &:= \frac{\exp(r(y_1))}{\exp(r(y_1)) + \exp(r(y_2))}, \end{aligned}$$

where $r : \mathcal{Y} \to \mathbb{R}$ is a reward function assumed common for all users. Typically, r is modelled by a neural network r_ϕ whose parameters ϕ are fitted by maximising the likelihood of the preference dataset $\mathcal{D}$. The learned reward function r_ϕ is then used to train the LLM policy π_θ by maximising:

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

where β is a parameter controlling the deviation from the base reference policy π_{ref}.

DPO under the BTL model. Thanks to a change of variables trick, DPO (Rafailov et al., 2024) enables direct optimisation of the policy parameters θ according to the following objective:

$$\max_\theta \mathop{\mathbb{E}}_{(x,\tilde{y}^w,\tilde{y}^\ell)\sim\mathcal{D}} \left[\log\sigma\left(\beta\log\frac{\pi_\theta(\tilde{y}^w|x)}{\pi_{\mathrm{ref}}(\tilde{y}^w|x)} - \beta\log\frac{\pi_\theta(\tilde{y}^\ell|x)}{\pi_{\mathrm{ref}}(\tilde{y}^\ell|x)}\right)\right].$$

Distributional Preference Learning. We note that both methods are based on the BTL model which assumes that the internal preferences of individual users can be summarised well with a single reward function r. As discussed in detail by (Siththaranjan et al., 2024), this approach does not cater to diverse, potentially conflicting user preferences that may stem from unobservable effects of the so-called "hidden context". To uncover the impact of the hidden context, Siththaranjan et al. (2024) propose distributional preference learning (DPL)–a class of models that estimate a distribution of possible reward values for each option rather than returning simple point estimates. In particular, their proposed mean-variance model assumes that:

$$p(y_1 \succ y_2) = \mathbb{E}\left[\mathbb{1}\{r(y_1) - r(y_2) > 0\}\right],$$

where $r(y) \sim \mathcal{N}(\mu(y), \sigma(y))$, with both μ and σ modelled as outputs of a neural network. While this model enables the measurement of the overall variability of utilities that users assign to a single option y, it does not solve the problem of identifying the expected value of the utility assigned to a given option for a given individual.

2.2. Problem Definition

Our Setup. With the focus on modelling preferences of individual users, our setup assumes access to a pre-collected dataset $\mathcal{D} = \{\mathcal{D}_i\}_{i\in\mathcal{I}}$ of preference choices collected from multiple users indexed by $i \in \mathcal{I}$. Each user-specific dataset,

$\mathcal{D}_i$ consists of a number of preference pairs represented as tuples $\mathcal{D}_i = \{(y^w_{i,j}, y^\ell_{i,j})\}_{j=1}^{N_i}$, where $y^w_{i,j}$ is the option chosen by user i and $y^\ell_{i,j}$ the rejected option.

The goal. Given a small, few-shot sample of preferences expressed in a context dataset $\mathcal{D}_i^C = \{(y^w_{i,j}, y^\ell_{i,j})\}_{j=1}^{N^C}$, we wish to: ▶ **1)** For any $y_1, y_2 \in \mathcal{Y}$ not part of $\mathcal{D}_i^C$, predict the probability of the user preferring y_1 over y_2. ▶ **2)** Align the LLM policy π_θ with the user-specific preferences so that for any new prompt x, with a high likelihood, the LLM generates content aligned with preferences of the given user.

3. Method

3.1. Stochastic Reward Modelling

We assume that users, indexed by i, express their preferences according to their internal reward function $r_i : \mathcal{Y} \to \mathbb{R}$. We model these functions as samples from a stochastic process. That is, each $r_i = r(\cdot; z_i^*)$ for a particular value of $z_i^* \in \mathcal{Z}$, where $\mathcal{Z}$ is an unknown latent space. Given a particular value of z_i^* and two options $y_1, y_2 \in \mathcal{Y}$, the distribution of preferences is modelled as:

$$p(y_1 \succ y_2 | z_i^*) = \frac{\exp(r(y_1; z_i^*))}{\exp(r(y_1; z_i^*)) + \exp(r(y_2; z_i^*))} \tag{1}$$

$$= \sigma\left(r(y_1; z_i^*) - r(y_2; z_i^*)\right). \tag{2}$$

To address the problem of predicting user preferences for any two previously unobserved options y_1, y_2, given a contextual dataset $\mathcal{D}_i^C$, we draw inspiration from the line of work on Neural Processes (NPs) (Garnelo et al., 2018a;b). An NP is a meta-learning model that learns to map contextual observations to the posterior predictive distribution. In our context, we will aim to approximate:

$$p(y_1 \succ y_2 | \mathcal{D}_i^C) = \int p(y_1 \succ y_2 | z_i^*) p(z_i^* | \mathcal{D}_i^C) dz_i^*, \tag{3}$$

where $p(z_i^* | \mathcal{D}_i^C)$ is the true conditional of z_i^* given the observed context dataset $\mathcal{D}_i^C$ which is unknown to us. NPs approximate it with either a variational distribution $q(\cdot|\mathcal{D}_i^C)$ or a deterministic map from the context dataset $\mathcal{D}_i^C$ to a fixed embedding $z \in \mathbb{R}^d$. For simplicity, we choose to present here the latter option and model the value of the individual reward function, conditioned on the contextual dataset $\mathcal{D}_i^C$ as:

$$r_\phi(y|\mathcal{D}_i^C) = g_{\phi_d}(y, z_i^C), \tag{4}$$

$$z_i^C = h_{\phi_e}(\mathcal{D}_i^C) = h_{\phi_{e,2}}\left(\frac{1}{N^C}\sum_{j=1}^{N^C} h_{\phi_{e,1}}(y^w_{i,j}, y^\ell_{i,j})\right), \tag{5}$$

where g_{ϕ_d} and h_{ϕ_e} are decoder and encoder networks, respectively. As is standard in NPs, the network h_{ϕ_e} is implemented as a Deep Set (Zaheer et al., 2018) enabling conditioning the reward function on an arbitrary number of observed data points N^C. Replacing $p(z_i^*|\mathcal{D}_i^C)$ with a delta-mass centered at $z_i^C = h_{\phi_e}(\mathcal{D}_i^C)$ gives:

$$p(y_1 \succ y_2 | \mathcal{D}_i^C) \approx p(y_1 \succ y_2 | z_i^C) \tag{6}$$

$$= \sigma(r_\phi(y_1|z_i^C) - r_\phi(y_2|z_i^C)) \tag{7}$$

We refer to this model as the NP-BTL reward model.

Algorithm 1 NP-BTL reward model training

```
Require: The dataset $\mathcal{D} = \{\mathcal{D}_i\}_{i\in\mathcal{I}}$, Max iter $T$, Initial parameters $\phi$, Learning rate $\eta$
  $t = 0$
  while $t < T$ do
    Randomly sample a user $i$
    Subsample $\mathcal{D}_i^C$, $\mathcal{D}_i^T$ from $\mathcal{D}_i$
    $\mathcal{L}(\phi) \leftarrow -\sum_{(y^w, y^l)\in\mathcal{D}_i^T} \log p_\phi(y^w \succ y^l | \mathcal{D}_i^C)$
    Update parameters with SGD: $\phi \leftarrow \phi - \eta\nabla\mathcal{L}(\phi)$
    $t \leftarrow t + 1$
  end while
  return $\phi$
```

Training. The NP-BTL model is trained in an episodic fashion. The parameters ϕ are found by maximising the likelihood on tasks consisting of pairs of so-called context and target datasets $(\mathcal{D}_i^C, \mathcal{D}_i^T)$, with $\mathcal{D}_i^C$ and $\mathcal{D}_i^T$ subsampled from $\mathcal{D}_i$, where $\mathcal{D}_i^C$ acts as the observable subset of context data points and $\mathcal{D}_i^T = \{(y^w_{i,j}, y^\ell_{i,j})\}_{j=1}^{N_T}$ contains pairs of options whose preference order we aim to predict. The simplified training procedure (assuming a batch size of 1) is presented in Algorithm 1.

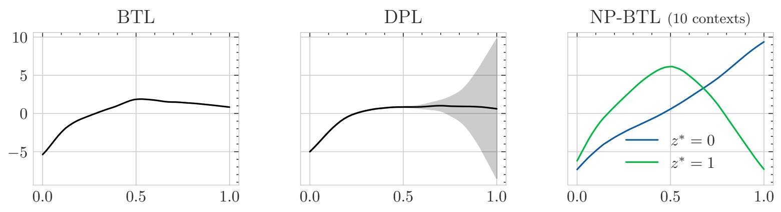
3.2. Conditional Direct Preference Optimisation

Our next step is to consider the training of personalisable LLM policies π_θ that can be conditioned on the contextual datasets $\mathcal{D}_i^C$. To do this, we need to find a way of modulating the output of a language model based on the observed context set $\mathcal{D}_i^C$, and crucially, be able to do so at inference time. To enable this, we consider a prototype method based on modulating hypernetworks inspired by (Perez et al., 2017). We model conditional policies as:

$$\pi_\theta(\tilde{y}|x, \mathcal{D}_i^C) = \pi_{\theta'}(\tilde{y}|x), \quad \theta' = \theta \odot \tau_i, \quad \tau_i = h_\phi(\mathcal{D}_i^C).$$

In the above, θ are the original parameters of the LLM and τ_i is the set of modulating parameters output by a hypernetwork h_ϕ build on top of the context encoder h_{ϕ_e}, i.e. $h_\phi(\mathcal{D}_i^C) = h_{\phi_{mod}}(h_{\phi_e}(\mathcal{D}_i^C)) \equiv h_{\phi_{mod}}(z_i^C)$, so that $\phi = \{\phi_e, \phi_{mod}\}$. Details on implementing the modulating operation $\theta \odot \tau_i$ are presented in the Appendix D.2.3. Extending the conventional DPO objective to our conditional formulation, we optimise the parameters θ and ϕ with respect to:

$$\max_{\theta,\phi} \mathop{\mathbb{E}}_{\substack{(\mathcal{D}_i^C, \mathcal{D}_i^T), \\ (x, \tilde{y}^w, \tilde{y}^\ell)\in\mathcal{D}_i^T}} \left[\log\sigma\left(r_\theta(y^w) - r_\theta(y^\ell)\right)\right], \tag{8}$$



(a) Unconditional reward models: BTL and DPL fitted to the synthetic data vs. the conditional NP-BTL model, given ten randomly sampled contextual data points with $z^* = 0$ and $z^* = 1$.

N^C	Accuracy [%]
0	75.2 ± 0.6
1	80.0 ± 0.4
3	88.2 ± 0.3
5	93.6 ± 0.2
10	97.2 ± 0.2

(b) Accuracy of the NP-BTL model vs. the number of contextual data points. Baseline accuracy is 75%.

Figure 1: *Synthetic data results.* Conditioning on user-provided preferences enables the conditional reward model to recover the correct mode of behaviour. As the number of context data points increases

where $r_\theta(y) := \beta \log \frac{\pi_\theta(\tilde{y}|x,\mathcal{D}_i^C)}{\pi_{\text{ref}}(\tilde{y}|x)}$ is the implicit DPO reward (Rafailov et al., 2024). As in NP-BTL, during training, the subsets $(\mathcal{D}_i^C, \mathcal{D}_i^T)$ are subsampled from $\mathcal{D}_i$. The model learns to condition its outputs on the observed contexts by training over multiple pairs of context and target datasets. The introduced parameter-space conditioning enables flexible modulation of the behaviour of the LLM policy.

4. Empirical Studies and Insights

In this section, we demonstrate the performance of both NP-BTL and NP-DPO on a series of illustrative experiments with synthetic data and larger-scale experiments on real data. We also provide further insights into the efficacy of conditioning with respect to the conflict rate within the training data and the number of contextual examples. The error bars in all plots represent the standard errors across testing tasks. Details on the experimental setups can be found in Appendix D.2. For presentation clarity, results in this sections are presented just for one choice of the base language model. An extensive evaluation across more LLMs can be found in Appendix A.

4.1. Illustrative comparison of existing approaches.

We begin with an illustrative example qualitatively comparing NP-BTL with the standard BTL and the DPL models.

Example 4.1. *We simulate a heterogeneous population of users expressing their preferences with respect to options represented as values $y \in [0, 1]$. We assume that our population consists of two kinds of users represented by the non-observable latent variables $z^* \sim \text{Bernoulli}(0.5)$ and we model the corresponding reward functions as:*

$$r(y, z^*) = x \text{ for } y \leq 0.5 \quad \text{and} \tag{9}$$

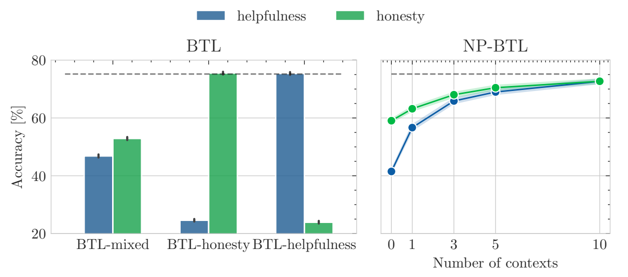
$$r(y, z^*) = \begin{cases} 2y & \text{if } z^* = 0 \\ 1 - y & \text{if } z^* = 1 \end{cases} \text{ for } y > 0.5 \tag{10}$$

We generate a synthetic dataset of options $y_1, y_2 \in [0, 1]$ and user choices made according to $p(y_1 \succ y_2) = \mathbb{1}\{(y_1, z^*) > r(y_2, z^*)\}$ and fit all three models: NP-BTL, BTL, and DPL. During training, the number of contexts for the NP-DPL models varies between 0 and 10, and the number of targets is set to 20. Figure 1a illustrates the fitted reward functions for all three models. We observe that, as expected, the BTL model averages out the rewards across the entire population, resulting in a flat reward function for $y > 0.5$ and failing to align with the preferences of the individual subgroups of users. While the DPL model can appropriately capture the variation in user preferences for options $y > 0.5$, it cannot adapt to user-specific preferences. Our conditional model successfully recovers the shape of the ground truth rewards. Table 1b shows the efficacy of the adaptation based on the number of preference choices provided as contexts. We observe that for $N^C = 0$, the conditional model matches the baseline performance of the non-conditional models, ≈75%. As the number of contextual data points increases, so does the alignment of the predicted rewards and the resulting predictions, reaching near-perfect accuracy at $N^C = 10$.

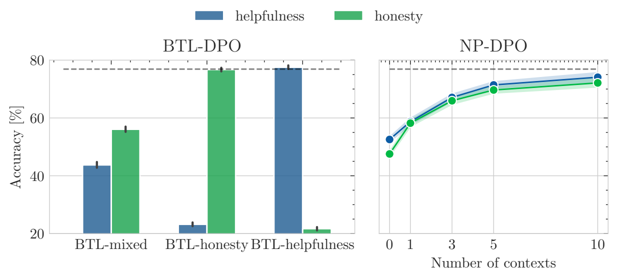
4.2. Real-world example: Helpfulness vs. Honesty

We move beyond synthetic data to evaluate NP-BTL and NP-DPO on the real-world UltraFeedback dataset[1](Cui et al., 2024) commonly used to evaluate RLHF and DPO pipelines. Each sample within this dataset consists of user prompts x, two candidate LLM responses $\tilde{y}_1$ and $\tilde{y}_2$ and scores assessing both options with respect to the following categories: helpfulness, honesty, truthfulness and instruction following. To simulate users with varying preferences, we define two reward functions: $r(y; 0) = \text{helpfulness}(\tilde{y})$ and $r(y; 1) = \text{honesty}(\tilde{y})$ according to which we generate the preference choices. During training and testing, the reward models are only presented with the two options in a text format and the user choice. The underlying scores of helpfulness and honesty are part of the hidden context. This experiment aims to investigate the ability of NP-BTL and NP-DPO to identify the two distinct modes of user preferences and quickly adapt to one of them, given a small sample of contextual preference choices. Thus, we filter our training and testing preference dataset to consist of only

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



(a) Accuracies of the baseline BTL models (left) and the NP-BTL model (right).



(b) Accuracies of the implicit rewards of LLM policies trained with BTL-DPO (left) and NP-DPO (right).

Figure 2: *Performance of a) NP-BTL and b) NP-DPO vs. their non-conditional counterparts.*

pairs which are *conflicting* with respect to $r(\cdot;0)$ and $r(\cdot;1)$. Such pairs maximise the information gain with respect to the true value of z^*.

Definition 4.2 (Conflicting pairs)**.** Given two reward functions: r_0, r_1, we say that two options y, y' are conflicting if the decision made according to r_0 differs from the decision made with r_1, i.e.

$$\mathbb{1}\{r_0(y) > r_0(y')\} \neq \mathbb{1}\{r_1(y) > r_1(y')\}.$$

We fit the NP-BTL reward model with Llama-3-8B[2] text embeddings and fine-tune the gemma-2b[3] language model with NP-DPO on our generated dataset. We compare the accuracies of the NP-BTL reward model and the accuracies of the implicit rewards of NP-DPO with respect to the BTL reward models' accuracies and the implicit rewards of the LM policy trained with standard BTL-DPO. We train two kinds of baseline models: 1) trained on the same heterogeneous dataset of preferences as the NP-based models (BTL-mixed) and 2) trained on each of the two homogeneous subsets of data generated according to the honest and helpful reward functions (BTL-honesty and BTL-helpfulness). Note that the BTL-honesty and BTL-helpfulness models require privileged access to the value of the hidden variables z_i^*. Their performance is reported for comparative purposes as the theoretically achievable upper bound.

Figure 2 shows the results with models evaluated separately on the honesty valuing and helpfulness valuing subsets of users. We observe the following. Due to the nature of this dataset with 100% conflicting pairs, the standard BTL-mixed models fail to align with the heterogeneous preferences of the population, resulting in an average performance of 50% across the entire population. In contrast, the NP-based rewards and policies successfully adapt to the preferences of individual subgroups as the number of contextual data points increases. At $N^C = 10$, the conditional NP-BTL rewards nearly match the accuracies of the dedicated BTL-honesty and BTL-helpfulness models (see dashed lines). Similar observations can be drawn for the NP-DPO implicit rewards.

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

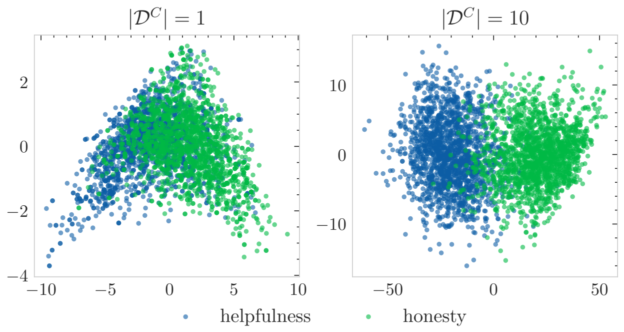


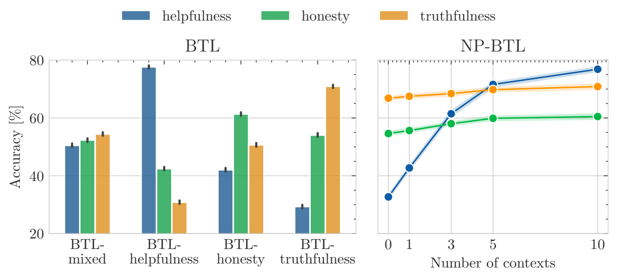
Figure 3: *PCA of context embeddings*. Points are labelled by the true value of z_i^*. We observe a clear clustering effect with increasing contextual data points.

We investigate the latent embeddings z_i^C of the context datasets $\mathcal{D}_i^C$ used to condition the reward functions. In Figure 3, we display the plots of the embeddings reduced to 2 dimensions via principle component analysis (PCA). Points are labelled according to the ground truth value of z_i^* according to which the context dataset has been generated. Two clusters are visible in the graphic, indicating that the model learned to separate the two types of users.

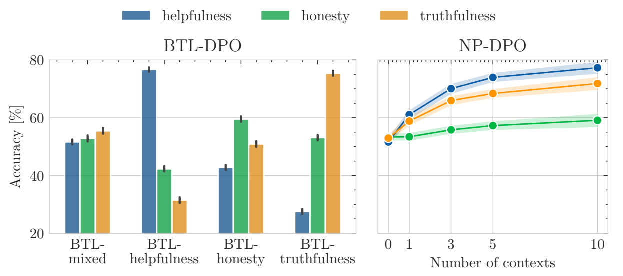
4.3. Beyond just two behavioural modes

4.3.1. Helpfulness vs Honesty vs Truthfulness

We adopt an analogous setup as before, but this time with the reward functions representing three different modes of behaviour. We let $z^* \sim \text{Multinomial}([1/3, 1/3, 1/3])$, with $z^* \in \{0, 1, 2\}$ corresponding to the honesty, helpfulness, and truthfulness objectives, respectively. To ensure high informativeness of contextual samples, we filter the dataset to pairs y_1, y_2 for which one choice made according to $r(\cdot\,;0)$, $r(\cdot\,;1)$, $r(\cdot\,;2)$ is distinct from the others. As previously, we train NP-BTL and NP-DPO models and compare them against baselines analogous to our previous setup.

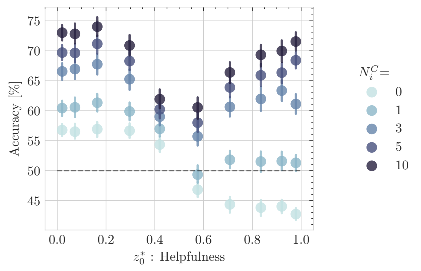


(a) Accuracies of the baseline BTL models (left) and the NP-BTL model (right).

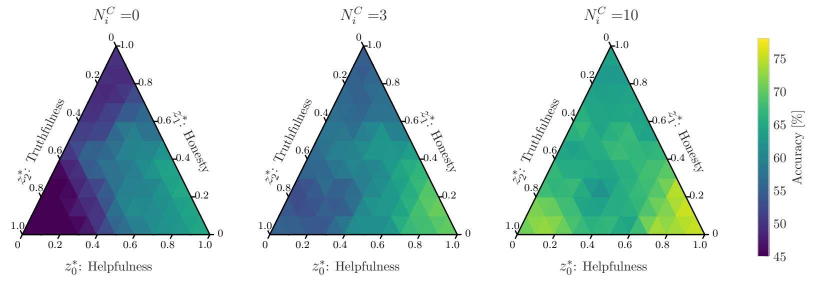


(b) Accuracies of the implicit rewards of LLM policies trained with BTL-DPO (left) and NP-DPO (right).

Figure 4: *Performance of a) NP-BTL and b) NP-DPO vs. their non-conditional counterparts.*



(a) 2-dimensional setup.



(b) 3-dimensional setup.

Figure 5: *Accuracies of the NP-BTL models.* a) y-axis represents the accuracies of the NP-BTL-2D rewards evaluated on test-set preference pairs with choices made according to $r_{2D}(\cdot\,; z^*)$, where $z_0^* = 1 - z_1^*$ varies between 0.0 and 1.0. Values of the x-axis are binned into ten equally-sized bins in the range $[0, 1]$. b) Colour corresponds to the accuracy of the NP-BTL-3D rewards evaluated on test-set preference pairs with choices made according to $r_{3D}(\cdot\,; z^*)$, varying the value of z^* inside the 3D simplex. The area of the simplex is divided into triangular bins with edge lengths of 0.1.

Figure 4 shows the test-time accuracies of the learned rewards evaluated separately on preference choices made with the helpfulness- ($r(\cdot\,; 0)$ in blue), honesty- ($r(\cdot\,; 1)$ in green), and truthfulness-preferring ($r(\cdot\,; 2)$ in orange) rewards. We make analogous conclusions to experiments on the 2-mode datset. The BTL-mixed reward model fails to align with any of the three objectives. The accuracy of the NP-BTL model increases with the number of contextual preference pairs across all three modes of behaviour. At $N_i^C = 10$, its performance matches the performance of the dedicated BTL-helpfulness, -honesty and -truthfulness models.

4.3.2. A continuous spectrum of behaviours

One of the appealing properties of the NP-based models is that they are not constrained to modelling just a fixed set of rewards. Instead, they can represent an entire *spectrum* of behaviours. We illustrate this with the UltraFeedback dataset in a two- and three-dimensional setup.

2D: We let $z^* = [z_0^*, z_1^*]^T \sim \text{Dirichlet}([0.5, 0.5])$ and we set $r_{2D}(y; z^*) = z_0^* \cdot \text{helpfulness}(\tilde{y}) + z_1^* \cdot \text{honesty}(\tilde{y})$

3D: We let $z^* = [z_0^*, z_1^*, z_2^*]^T \sim \text{Dirichlet}([0.5, 0.5, 0.5])$ and we $r_{3D}(y; z^*) = z_0^* \cdot \text{helpfulness}(\tilde{y}) + z_1^* \cdot \text{honesty}(\tilde{y}) + z_2^* \cdot \text{truthfulness}(\tilde{y})$

We train one NP-BTL model for the 2D setup (NP-BTL-2D) and one for the 3D setup (NP-BTL-3D). Figure 5 shows the accuracies of the NP-BTL-2D and NP-BTL-3D rewards on the test-set pairs, with choices made according to r_{2D} and r_{3D}, varying the value of the ground-truth z^* inside the 2- and 3- dimensional simplex, respectively. Both in the 2D and 3D setups, with increasing contextual data points, the accuracy improves across the entire spectrum of behavioural modes represented by z^*. This observation implies that our NP-BTL model successfully adapts its predictions to the observable contexts, enabling not only discrete, group-level alignment, but a truly personalisable, user-level alignment.

4.3.3. Generalisation to new users

Appendix A.3 presents results showing the ability of NP-based reward modelling in generalising to novel users.

4.4. Correlations in data and identifiability of the hidden context.

4.4.1. Analytic study

In the experiments of section 4.2, training and testing datasets were constructed such that the competing options were perfectly conflicting. In reality, the underlying factors driving human decisions may exhibit strong correlations. For instance, if an answer is helpful, it is also likely that it is honest. As a result, real-world datasets do not consist of just conflicting examples, despite being generated by distinct reward functions. In this section, we investigate how correlations within the collected datasets of preference pairs affect the model's ability to identify a user's underlying reward function correctly. We begin with an analytics example inspired by the example of helpfulness vs. honesty.

Example 4.3. *Suppose all options $y \in \mathcal{Y}$ can be represented with two numbers $y = [h_0, h_1]$ (e.g. helpful and honest scores). We assume that our population of users consists of two equally sized groups of users represented by ground-truth latent variables $z^* \sim \text{Bernoulli}(0.5)$; with the corresponding reward functions $r(y, z^* = 0) = h_0$ and $r(y, z^* = 1) = h_1$. Decisions are taken according to:*

$$p(y^w \succ y^\ell | z^*) = \mathbb{1}\left\{r(y^w, z^*) > r(y^\ell, z^*)\right\}$$

This setup simulates a scenario where half of the users always prefer the more helpful while the half always choose the more honest answer. To investigate the impact of correlations in honest vs. helpful scores on the performance of our model, we will assume that scores of all options are i.i.d and normally distributed so that $[h_1, h_2] \sim \mathcal{N}(\mu, \Sigma)$.

Suppose we observe a contextual dataset $\mathcal{D}^C$ provided by a single user according to the abovementioned process. We wish to investigate the performance of a Bayes optimal classifier in determining the underlying value of z^*. We let

$$\hat{Z}(\mathcal{D}^C) = \arg\max_{k \in \{0,1\}} \mathbb{P}(z^* = k | \mathcal{D}^C), \tag{11}$$

and we look at the error rate of $\hat{Z}$ defined as:

$$\mathcal{E} := \mathbb{E}\left[\hat{Z}(\mathcal{D}^C) \neq z^*\right]. \tag{12}$$

In the above, the expectation is taken over all possible observable contexts $\mathcal{D}^C$. In case when we restrict $\mathcal{D}^C$ to observations consisting of only a single preference choice $\mathcal{D}^C = \{y^w \succ y^\ell\}$, and assuming that $\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$, we can show that $\mathcal{E} = \frac{1}{4} + \frac{1}{2\pi}\arcsin(\rho)$ (Appendix B). Note, at $\rho = -1.0$, the dataset consists of 100% conflicting pairs, resulting in the theoretical 0% error rate, meaning that a single preference choice is sufficient to determine the underlying value of z^* unambiguously. On the other hand, at $\rho = 1.0$, none of the preference pairs are conflicting. If a response is helpful, it must also be honest. Such data does not allow us to differentiate between the underlying reward functions of individuals, resulting in an expected error rate of 50%.

In light of the above, we now assess the performance of our NP-BTL model on synthetic data generated according to the

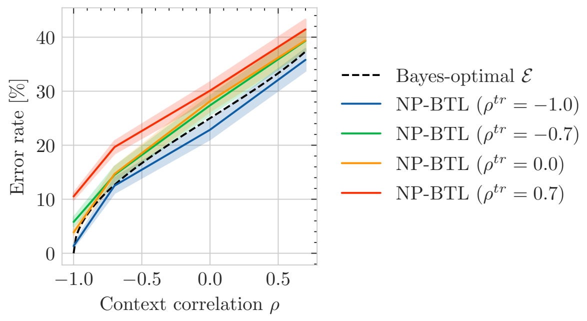


Figure 6: *Error rate of NP-BTL vs Bayes optimal.*

process described in Example 4.3. We fit four independent NP-BTL models where, at training time, the correlation ρ is set to $\rho^{tr} \in \{-1.0, -0.7, 0.0, 0.7\}$. This corresponds to a rate of conflicting pairs of 100%, 75%, 50% and 25%, respectively. Figure 6 shows the error rate of all four NP-BTL models where the two options forming the single preference pair provided in context are sampled from a normal distribution with a varying parameter ρ value. We find that all models achieve near-optimal error rates, with the error rate increasing as the correlation in the training set increases. This confirms our hypothesis that datasets exhibiting strong correlations provide weaker training signals for the NP-BTL model to learn how to differentiate between conflicting reward functions of individuals.

4.4.2. Helpfulness vs. Honesty

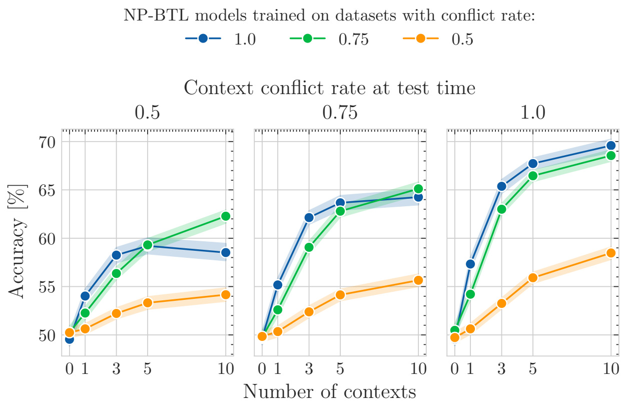


Figure 7: *Accuracy of NP-BTL rewards vs. conflict rate on the HH dataset.*

We now go back to the Helpfulness vs. Honesty (HH) dataset introduced in section 4.2 to investigate the impact of the conflict rate within the training dataset and in-context examples on the performance of our NP-BTL rewards. We train three NP-BTL models on three versions of the HH dataset, with rates of conflicting pairs of 0.5, 0.75 and 1.0, both in the context and target pairs. We evaluated these models on a held-out test set where the rate of conflicting pairs in $\mathcal{D}^C$ varies, and the rate of conflict in $\mathcal{D}^T$ is set to 1.0. 7 shows the results. We observe that model performance decreases for all models as the in-context conflict rate decreases. Contextual pairs which are not conflicting do not provide the necessary information to identify the hidden intentions of the user. We also observe that the performance of the NP-BTL model trained on data with a 0.5 conflict rate is significantly worse than that of the NP-BTL models trained on datasets with conflict rates of 1.0 and 0.75. This confirms our hypothesis from earlier–data with low conflict rates may not provide sufficiently strong training signals to learn to identify distinct human preferences.

5. Related Work

RLHF and DPO. Reinforcement Learning from Human Feedback (RLHF) has been foundational to aligning AI systems with human preferences. This technique typically involves learning a reward function from binary human preference data, commonly modelled using the BTL framework (Bradley & Terry, 1952). The applications of RLHF to language models has been a significant focus of recent research, aiming to align LLMs with human values and preferences. Notable works in this area include those focusing on fine-tuning models (Bai et al., 2022; Stiennon et al.; Kim et al., 2022; Ouyang et al., 2022). However, aligning LLMs through RLHF remains challenging due to training instability, which typically requires learning a reward model and the subsequent fine-tuning of the LLM. To address these inefficiencies, (Rafailov et al., 2024) propose DPO, a method that bypasses the need for explicit reward model training by optimising preference data directly in a supervised fashion.

Pluralistic Alignment. Most RLHF and DPO methods are based on the BTL model, which assumes a single reward function for all users. This approach can fail to capture the diversity of user preferences. Sorensen et al. (2024) identify the need to develop methods that enable pluralistic alignment, serving humans with varying objectives. An important contribution in this field has been in recognising the "hidden context effect"–factors that influence user preferences in a not directly observable manner. To address these effects, Siththaranjan et al. (2024) propose DPL, modelling a distribution over plausible reward values for each competing option. While DPL enables the identification of potential hidden context effects, it does not provide a solution for adapting the reward functions, and thus the final LLM policies, to individual users. Our approach builds on the concept of steerable pluralism (Sorensen et al., 2024), wherein a model is steered towards responses that faithfully represent a given user's preferences or a subgroup of users.

Multi-objective Approaches. Multi-objective learning frameworks address AI alignment with diverse or conflicting user preferences. Pareto-optimal optimisation balances multiple objectives by finding optimal trade-offs (Boldi et al., 2024; Zhong et al., 2024). In contrast, our focus is on steerable LLM adaptation to individual preferences. A limited number of related works attempts to extend preference optimisation with such flexibility. Rewarded Soups (Ramé et al., 2023) fine-tune one LLM per reward function and interpolates the LLM's parameters for conditional generation. Wang et al. (2024c) learns adapting parameters to enable inference-time control. Others (Wang et al., 2024b; Guo et al., 2024b) use prompt-based methods, steering LMs via reward weightings in text. Unlike these approaches, our method neither relies on a predefined set of objectives nor requires users to express they preferences as a vector of weights. Instead, we infer user-specific preferences based on a few-shot sample of their historical choices. We learn to condition reward functions and LLM policies by meta-learning from a heterogeneous dataset of user preferences with unknown factors of variability. The presented NP-BTL reward model can be seen as a generalisation of the work presented by (Poddar et al., 2024) enabling conditioning on a variable number of contextual preference pairs.

Appendix E presents a summary table of the related work.

6. Limitations

Due to computational limitations, our empirical evaluation is limited to relatively small LLMs. We presents additional results with three other LLMs in Appendix A.1 and A.2. To ensure safe and reliable deployment of few-shot steerable LLM policies with NP-DPO, experiments on larger datasets and LLMs with more parameters are essential. We also note that the proposed modulation operation based on FiLM layers is just a prototype solution which may be further improved by considering more parameter-efficient approaches.

7. Conclusions

This work introduces a novel framework for few-shot steerable alignment, addressing the challenge of personalising AI systems. By extending the BTL model to handle heterogeneous preferences with unobserved variability factors, we enable the inference of user preferences from small samples of their choices. Our conditional reward modelling approach (NP-BTL) and its extension to DPO (NP-DPO) allow for training steerable LLM policies that can adapt to individual user preferences with minimal data. Experiments on synthetic and real-world datasets demonstrate the effectiveness of our methods in efficiently capturing and aligning with diverse human preferences. We also analyse the value of contextual data in determining the underlying reward function of a user. Noting that not all preference pairs are equally informative, future research directions include exploring active learning approaches for heterogeneous preference data collection, enabling maximum information gain about the internal preferences of users. We believe that the methods and insights presented in this paper mark a significant step towards developing truly personalisable AI agents.

8. Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

9. Acknowledgements

This work was supported by Azure sponsorship credits granted by Microsoft's AI for Good Research Lab. Katarzyna Kobalczyk is supported by funding from Eedi, Claudio Fanconi is sponsored by Canon Medical Systems Corporation, and Hao Sun is sponsored by the Office of Naval Research.

References

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., Chen, C., Olsson, C., Olah, C., Hernandez, D., Drain, D., Ganguli, D., Li, D., Tran-Johnson, E., Perez, E., Kerr, J., Mueller, J., Ladish, J., Landau, J., Ndousse, K., Lukosuite, K., Lovitt, L., Sellitto, M., Elhage, N., Schiefer, N., Mercado, N., DasSarma, N., Lasenby, R., Larson, R., Ringer, S., Johnston, S., Kravec, S., Showk, S. E., Fort, S., Lanham, T., Telleen-Lawton, T., Conerly, T., Henighan, T., Hume, T., Bowman, S. R., Hatfield-Dodds, Z., Mann, B., Amodei, D., Joseph, N., McCandlish, S., Brown, T., and Kaplan, J. Constitutional AI: Harmlessness from AI Feedback, December 2022. URL http://arxiv.org/abs/2212.08073. arXiv:2212.08073 [cs].

Boldi, R., Ding, L., Spector, L., and Niekum, S. Pareto-Optimal Learning from Preferences with Hidden Context. 2024. doi: 10.48550/ARXIV.2406.15599. URL https://arxiv.org/abs/2406.15599. Publisher: arXiv Version Number: 1.

Bradley, R. A. and Terry, M. E. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2334029. Publisher: [Oxford University Press, Biometrika Trust].

Chen, D., Chen, Y., Rege, A., and Vinayak, R. K. PAL: Pluralistic alignment framework for learning from heterogeneous preferences. In *NeurIPS 2024 Workshop on Behavioral Machine Learning*, 2024. URL https://openreview.net/forum?id=BpfxrR8tYs.

Chidambaram, K., Seetharaman, K. V., and Syrgkanis, V. Direct Preference Optimization With Unobserved Preference Heterogeneity, May 2024. URL http://arxiv.org/abs/2405.15065. arXiv:2405.15065 [cs].

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/d5e2c0adad503c91f91df240d0cd4e49-Abstract.html.

Cui, G., Yuan, L., Ding, N., Yao, G., He, B., Zhu, W., Ni, Y., Xie, G., Xie, R., Lin, Y., Liu, Z., and Sun, M. UltraFeedback: Boosting Language Models with Scaled AI Feedback, 2024. URL https://arxiv.org/abs/2310.01377. _eprint: 2310.01377.

Garnelo, M., Rosenbaum, D., Maddison, C., Ramalho, T., Saxton, D., Shanahan, M., Teh, Y. W., Rezende, D., and Eslami, S. M. A. Conditional Neural Processes. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1704–1713. PMLR, July 2018a. URL https://proceedings.mlr.press/v80/garnelo18a.html.

Garnelo, M., Schwarz, J., Rosenbaum, D., Viola, F., Rezende, D. J., Eslami, S. M. A., and Teh, Y. W. Neural Processes, 2018b. _eprint: 1807.01622.

Guo, Y., Cui, G., Yuan, L., Ding, N., Sun, Z., Sun, B., Chen, H., Xie, R., Zhou, J., Lin, Y., Liu, Z., and Sun, M. Controllable preference optimization: Toward controllable multi-objective alignment. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1437–1454, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.85. URL https://aclanthology.org/2024.emnlp-main.85/.

Guo, Y., Cui, G., Yuan, L., Ding, N., Wang, J., Chen, H., Sun, B., Xie, R., Zhou, J., Lin, Y., Liu, Z., and Sun, M. Controllable Preference Optimization: Toward Controllable Multi-Objective Alignment, 2024b. URL https://arxiv.org/abs/2402.19085. _eprint: 2402.19085.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-Rank Adaptation of Large Language Models, October 2021. URL http://arxiv.org/abs/2106.09685. arXiv:2106.09685 [cs].

Jang, J., Kim, S., Lin, B. Y., Wang, Y., Hessel, J., Zettlemoyer, L., Hajishirzi, H., Choi, Y., and Ammanabrolu, P. Personalized Soups: Personalized Large Language Model Alignment via Post-hoc Parameter Merging, October 2023. URL http://arxiv.org/abs/2310.11564. arXiv:2310.11564 [cs].

Kim, C., Park, J., Shin, J., Lee, H., Abbeel, P., and Lee, K. Preference Transformer: Modeling Human Preferences using Transformers for RL. September 2022. URL https://openreview.net/forum?id=Peot1SFDX0.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. URL https://api.semanticscholar.org/CorpusID:6628106.

Li, X., Lipton, Z. C., and Leqi, L. Personalized language modeling from personalized human feedback. In *ICLR 2024 Workshop on Reliable and Responsible Foundation Models*, 2024. URL https://openreview.net/forum?id=8qavdRGl7J.

Liu, J., Zhu, Y., Wang, S., Wei, X., Min, E., Lu, Y., Wang, S., Yin, D., and Dou, Z. Llms + persona-plug = personalized llms, 2024. URL https://arxiv.org/abs/2409.11901.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. Training language models to follow instructions with human feedback, March 2022. URL http://arxiv.org/abs/2203.02155. arXiv:2203.02155 [cs].

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

Perez, E., Strub, F., Vries, H. d., Dumoulin, V., and Courville, A. FiLM: Visual Reasoning with a General Conditioning Layer, 2017. URL https://arxiv.org/abs/1709.07871. _eprint: 1709.07871.

Poddar, S., Wan, Y., Ivison, H., Gupta, A., and Jaques, N. Personalizing Reinforcement Learning from Human Feedback with Variational Preference Learning, August 2024. URL http://arxiv.org/abs/2408.10075. arXiv:2408.10075 [cs].

Rafailov, R., Sharma, A., Mitchell, E., Ermon, S., Manning, C. D., and Finn, C. Direct Preference Optimization: Your Language Model is Secretly a Reward Model, July 2024. URL http://arxiv.org/abs/2305.18290. arXiv:2305.18290 [cs].

Ramé, A., Couairon, G., Shukor, M., Dancette, C., Gaya, J.-B., Soulier, L., and Cord, M. Rewarded soups: towards Pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards, October 2023. URL http://arxiv.org/abs/2306.04488. arXiv:2306.04488 [cs].

Siththaranjan, A., Laidlaw, C., and Hadfield-Menell, D. DISTRIBUTIONAL PREFERENCE LEARNING: UNDERSTANDING AND ACCOUNTING FOR HIDDEN CONTEXT IN RLHF. 2024.

Sorensen, T., Moore, J., Fisher, J., Gordon, M., Mireshghallah, N., Rytting, C. M., Ye, A., Jiang, L., Lu, X., Dziri, N., Althoff, T., and Choi, Y. A Roadmap to Pluralistic Alignment, July 2024. URL http://arxiv.org/abs/2402.05070. arXiv:2402.05070 [cs].

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D. M., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. Learning to summarize from human feedback.

Wang, H., Lin, Y., Xiong, W., Yang, R., Diao, S., Qiu, S., Zhao, H., and Zhang, T. Arithmetic control of LLMs for diverse user preferences: Directional preference alignment with multi-objective rewards. In Ku, L.-W., Martins, A., and Srikumar, V. (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8642–8655, Bangkok, Thailand, August 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.468. URL https://aclanthology.org/2024.acl-long.468/.

Wang, H., Lin, Y., Xiong, W., Yang, R., Diao, S., Qiu, S., Zhao, H., and Zhang, T. Arithmetic Control of LLMs for Diverse User Preferences: Directional Preference Alignment with Multi-Objective Rewards, 2024b. URL https://arxiv.org/abs/2402.18571. _eprint: 2402.18571.

Wang, K., Kidambi, R., Sullivan, R., Agarwal, A., Dann, C., Michi, A., Gelmi, M., Li, Y., Gupta, R., Dubey, A., Ramé, A., Ferret, J., Cideron, G., Hou, L., Yu, H., Ahmed, A., Mehta, A., Hussenot, L., Bachem, O., and Leurent, E. Conditioned Language Policy: A General Framework for Steerable Multi-Objective Finetuning, 2024c. URL https://arxiv.org/abs/2407.15762. _eprint: 2407.15762.

Wang, K., Kidambi, R., Sullivan, R., Agarwal, A., Dann, C., Michi, A., Gelmi, M., Li, Y., Gupta, R., Dubey, K. A., Rame, A., Ferret, J., Cideron, G., Hou, L., Yu, H., Ahmed, A., Mehta, A., Hussenot, L., Bachem, O., and Leurent, E. Conditional language policy: A general framework for steerable multi-objective finetuning. In Al-Onaizan, Y., Bansal, M., and Chen, Y.-N. (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 2153–2186, Miami, Florida, USA, November 2024d. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.118. URL https://aclanthology.org/2024.findings-emnlp.118/.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Scao, T. L., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. M. Huggingface's transformers: State-of-the-art natural language processing, 2020. URL https://arxiv.org/abs/1910.03771.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R., and Smola, A. Deep Sets, 2018. URL https://arxiv.org/abs/1703.06114. _eprint: 1703.06114.

Zhao, S., Dang, J., and Grover, A. Group preference optimization: Few-shot alignment of large language models. In *R0-FoMo:Robustness of Few-shot and Zero-shot Learning in Large Foundation Models*, 2023. URL https://openreview.net/forum?id=p6hzUHjQn1.

Zhong, Y., Ma, C., Zhang, X., Yang, Z., Chen, H., Zhang, Q., Qi, S., and Yang, Y. Panacea: Pareto Alignment via Preference Adaptation for LLMs, May 2024. URL http://arxiv.org/abs/2402.02030. arXiv:2402.02030 [cs].

Appendix

Code: Code implementing the NP-BTL and NP-DPO models, alongside the instructions to reproduce the key experiments presented in this paper are made available at: https://github.com/fanconic/few-shot-steerable-alignment.

A. Additional results

In this section, we expand our results on real-world data to multiple LLMs, demonstrating that our proposed method generalises to different backbones. We observe that the conclusions from our original experiments generalise across different LLMs. With increasing number of contextual observations, the accuracy of the conditional NP rewards and policies increases, nearly matching the performance of the BTL models trained on the respective subsets of the data and significantly outperforming the baseline BTL-mixed policies.

A.1. NP-BTL with different LLM Backbones for Encoding

We run the BTL and NP-BTL reward models using `gemma-2b` embeddings of responses instead of the originally used `Llama-3-8B` embeddings. The size of the `gemma-2b` embeddings is 2048 (half of the size of the `Llama-3-8B` embeddings. We obtained the following results presented in an analogous format to Figures 2 and 4 of the main manuscript in Table 2 for the HH dataset and Table 2 for the HHT dataset.

Model	Helpfulness	Honesty
BTL-helpfulness	68.7 ± 0.3	32.1 ± 0.3
BTL-honesty	30.1 ± 0.2	69.7 ± 0.3
BTL-mixed	51.1 ± 0.3	48.5 ± 0.3

(a) Accuracy [%] of the BTL reward models

N_i^C	Helpfulness	Honesty
0	39.9 ± 0.3	60.3 ± 0.3
1	50.9 ± 0.4	62.0 ± 0.4
3	59.3 ± 0.4	66.4 ± 0.4
5	63.9 ± 0.4	67.5 ± 0.4
10	68.5 ± 0.4	69.7 ± 0.4

(b) Accuracy [%] of the NP-BTL model

Table 1: Comparison of BTL reward models (sub-table 1a) and NP-BTL model (sub-table 1b) using `gemma-2b` as backbone for encoding, instead of `Llama-3-8b` on the HH dataset with two behavioural modes.

Model	Helpfulness	Honesty	Truthfulness
BTL-helpfulness	70.8 ± 0.3	45.2 ± 0.3	34.1 ± 0.3
BTL-honesty	43.1 ± 0.3	56.2 ± 0.3	52.2 ± 0.3
BTL-truthfulness	32.2 ± 0.3	50.9 ± 0.3	69.5 ± 0.3
BTL-mixed	50.5 ± 0.3	55.0 ± 0.3	54.0 ± 0.3

(a) Accuracy [%] of the BTL reward models

N_i^C	Helpfulness	Honesty	Truthfulness
0	45.1 ± 0.3	52.0 ± 0.3	57.5 ± 0.3
1	51.6 ± 0.4	53.3 ± 0.3	59.8 ± 0.4
3	63.0 ± 0.4	57.9 ± 0.3	64.5 ± 0.4
5	68.6 ± 0.4	61.3 ± 0.4	68.0 ± 0.4
10	73.5 ± 0.4	62.7 ± 0.4	70.6 ± 0.4

(b) Accuracy [%] of the NP-BTL model

Table 2: Comparison of BTL reward models (sub-table 2a) and NP-BTL model (sub-table 2b) using `gemma-2b` as backbone for encoding, instead of `Llama-3-8b` on the HHT dataset with three behavioural modes.

We observe that the standard BTL-mixed models fail to align with the heterogeneous preferences of the population, resulting in an average performance of 50% across the entire population. In contrast, the NP-based rewards again successfully adapt to the preferences of individual subgroups as the number of contextual data points increases. This presents empirical evidence that NP-BTL can learn to condition on a few preference samples encoded by different backbone LLMs.

A.2. NP-DPO on different LLM Policy Models

In the same spirit as in the previous subsection, We repeat our experiments with NP-DPO and BTL-DPO using three different LLMs as the base models. While in the main paper we used `gemma-2b` to perform DPO and NP-DPO on, we change the policy model to `SmolLM2-1.7B`[4], `Qwen2.5-1.5B`[5] and `Llama-3.2-3B`[6]

We report the results obtained for the HHT dataset in Table 3. We observe that the results are consistent with what we reported in Figure 4b. Across all three policy models, the implicit reward accuracy increases with more available context samples N_i^C when trained with NP-DPO. If we apply a traditional BTL DPO learning model to a dataset of conflicting pairs, we observe that implicit reward accuracy stays around $\sim 50\%$. This provides empirical evidence that the NP-DPO and the FiLM layer injection method generalise well across different LLM policies.

LLM	**Model**	N_i^C	**Helpfulness**	**Honesty**	**Truthfulness**
`SmolLM2-1.7B`	BTL-helpfulness	0	80.8 ± 0.4	40.8 ± 0.5	27.1 ± 0.5
	BTL-honesty	0	33.6 ± 0.5	63.9 ± 0.6	56.2 ± 0.6
	BTL-truthfulness	0	28.4 ± 0.5	52.6 ± 0.5	76.1 ± 0.5
	BTL-mixed	0	59.0 ± 0.5	49.5 ± 0.5	45.3 ± 0.6
	NP-DPO	0	53.3 ± 0.5	51.9 ± 0.6	51.5 ± 0.6
		1	58.9 ± 0.7	51.7 ± 0.6	55.9 ± 0.7
		3	66.3 ± 0.8	53.0 ± 0.6	60.4 ± 0.8
		5	71.0 ± 0.8	55.0 ± 0.7	65.8 ± 0.8
		10	72.0 ± 1.0	58.2 ± 1.1	67.5 ± 1.1
`Qwen2.5-1.5B`	BTL-helpfulness	0	78.8 ± 0.4	39.4 ± 0.5	29.2 ± 0.5
	BTL-honesty	0	35.8 ± 0.5	65.1 ± 0.5	53.5 ± 0.5
	BTL-truthfulness	0	28.2 ± 0.5	52.5 ± 0.5	74.9 ± 0.4
	BTL-mixed	0	45.1 ± 0.5	54.9 ± 0.5	56.9 ± 0.5
	NP-DPO	0	41.0 ± 0.5	54.1 ± 0.5	61.9 ± 0.5
		1	49.4 ± 0.7	55.2 ± 0.5	64.2 ± 0.6
		3	58.1 ± 0.8	56.3 ± 0.6	68.8 ± 0.6
		5	64.0 ± 0.8	57.4 ± 0.7	71.0 ± 0.6
		10	72.6 ± 1.0	57.8 ± 1.0	72.7 ± 1.0
`Llama-3.2-3B`	BTL-helpfulness	0	79.6 ± 0.4	39.3 ± 0.5	31.3 ± 0.5
	BTL-honesty	0	41.7 ± 0.5	65.7 ± 0.5	50.7 ± 0.5
	BTL-truthfulness	0	26.7 ± 0.4	52.5 ± 0.5	78.5 ± 0.4
	BTL-mixed	0	56.6 ± 0.5	49.6 ± 0.5	52.0 ± 0.5
	NP-DPO	0	44.4 ± 0.5	55.1 ± 0.5	57.3 ± 0.5
		1	57.0 ± 0.8	55.4 ± 0.6	59.9 ± 0.7
		3	67.9 ± 0.7	56.8 ± 0.6	65.2 ± 0.7
		5	71.1 ± 0.7	58.3 ± 0.7	68.4 ± 0.7
		10	76.3 ± 0.9	60.6 ± 1.0	70.3 ± 1.0

Table 3: Accuracy [%] of BTL-DPO and NP-DPO on the implicit reward on the HHT dataset with three behavioural modes across different LLMs.

A.3. The Persona Dataset

To further demonstrate the flexibility of our method in adapting to diverse user preferences, we present an additional set of experiments with the Anthropic Persona dataset[7].

The Persona dataset consists of preference data of 135 simulated personas acting according to their underlying political or religious views, personality, and moral beliefs. To test the ability of the NP-based rewards models in learning across heterogenous preference data with unobserved factors of variability, in our experiments, we do not use any descriptions of the underlying behaviours of each persona—we assume it to be part of the hidden context. The only contextual information

[4]https://huggingface.co/HuggingFaceTB/SmolLM-1.7B-Instruct

[5]https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct

[6]https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct

[7]https://github.com/anthropics/evals/tree/main/persona

model	N_i^C	**Q1**	**Mean**	**Median**	**Q3**
BTL	-	47.8	49.5	50.1	51.5
NP-BTL	0	16.1	44.6	43.8	71.9
	1	20.5	50.9	51.8	78.3
	3	52.9	69.6	77.8	92.1
	5	67.2	78.5	86.6	96.3
	10	72.2	81.8	89.7	97.2

Table 4: Distribution of the average reward accuracy [%] among the testing personas. Results averaged across all 5 testing folds.

about a given persona is the few-shot sample of their preference choices.

Our main focus in using the Persona dataset is to test the ability of our NP reward models to generalise to previously unseen users (i.e. personas). To ensure that the reward functions of different personas are indeed conflicting, we introduce anti personas. Let $\mathcal{D}_i$ denote the original dataset corresponding to the persona i. For each i we randomly split $\mathcal{D}_i$ into two halves: $\mathcal{D}_{i'}$ and $\mathcal{D}_{i''}$ and we reverse the preference choices in $\mathcal{D}_{i''}$—creating the anti-persona of i. This procedure ensures that no pairs (y_1, y_2) are shared between the conflicting users i' and i''. The resulting collection of training data can be represented as $\{\mathcal{D}_{i'}\}_{i=1}^{135} \cup \{\mathcal{D}_{i''}\}_{i=1}^{135}$.

To minimise the impact of variance from the choice of which personas are in the testing dataset, we perform a 5-fold meta-training-testing procedure. We split all 270 personas (135 original + 135 anit-personas) into five folds and train five independent models. For each model, we use a different fold for testing and the remaining folds for meta-training. At test time, we always evaluate our models on unseen preference pairs belonging to unseen personas. Each training, validation, and testing meta-split consists of preference data of 177, 39, and 54 personas, respectively (with around 500 data points per persona). As in the experiments on the UF datasets, we use between 0 and 10 contextual data points and 20 target data points during both training and testing.

Results. We present the results of the NP-BTL model in Table 4. This table shows the mean, upper and lower quartiles of the mean accuracy of the learned rewards across all test-time, unseen personas. I.e. the score of 72.2% for the NP-BTL model at $N_i^C = 10$, means that for 75% of the personas, the mean accuracy of the learned reward is 72.2% or higher. We note that for the non-conditional BTL model and the NP model at $N_i^C = 0$ the alignment is poor and highly variable due to the high heterogeneity in preferences between different personas. This alignment improves substantially upon conditioning on the contextual preference data $\mathcal{D}_i^C$. Even with just 5 examples, for half of the test-time personas the accuracy is 86.6% or higher, and for 75% of personas the accuracy is 67.2% or higher. This confirms that our few-shot conditional reward model can successfully generalise to users acting according to previously unseen reward functions.

A.4. NP-BTL vs. VPL

As noted in the related work section, the NP-BTL model presented in this paper bears close resemblance with the VPL model proposed by Poddar et al. (2024). If VPL's encoder is extended to handle a varying number of in-context samples and if the NP-BTL's deterministic encoder is replaced with a variational encoder, we arrive at essentially equivalent architectures.

For completeness, we include a comparison of the NP-BTL reward model implemented with a deterministic encoder (as in the main body of this paper) and with two versions of a variational encoder:

The first implementation follows the training objective of neural process with variational encoding as introduced by Garnelo et al. (2018b). If we let $q(\cdot|\mathcal{D})$ correspond to the latent distribution inferred based on samples (x, y^w, y^ℓ) included in a dataset $\mathcal{D}$, the the loss of the NP-BTL model with variational encoding can be represented as:

$$\mathcal{L}(\phi) = \mathop{\mathbb{E}}_{(\mathcal{D}_i^C, \mathcal{D}_i^T)} \left\{ \mathbb{E}_{z_i^C \sim q(z|\mathcal{D}_i^C)} \left[\log p_\phi(\mathcal{D}_i^T | z_i^C) \right] - \beta D_{\mathrm{KL}} \left(q(z|\mathcal{D}_i^C) || q(z|\mathcal{D}_i^T) \right) \right\} \tag{13}$$

We let $q(\cdot|\mathcal{D})$ be a multivariate normal distribution whose mean and standard deviation are the outputs of the context encoding network.

The second implementation follows the setup of Poddar et al. (2024). Similarly as above, we let $q(\cdot|\mathcal{D})$ correspond to the latent distribution inferred based on samples (x, y^w, y^ℓ) included in a dataset $\mathcal{D}$ and we let $p(z)$ denote the variational prior. The loss under this model is defined as:

$$\mathcal{L}(\phi) = \mathop{\mathbb{E}}_{(\mathcal{D}_i^C, \mathcal{D}_i^T)} \left\{ \mathbb{E}_{z_i^C \sim q(z|\mathcal{D}_i^C)} \left[\log p_\phi(\mathcal{D}_i^T | z_i^C) \right] - \beta D_{\mathrm{KL}} \left(q(z|\mathcal{D}_i^C) || p(z) \right) \right\} \tag{14}$$

We let $q(z|\mathcal{D})$ be a multivariate normal distribution whose mean and standard deviation are the outputs of the context encoding network and $p(z)$ a standard normal.

Note, the two versions differ in the Kullback-Leibler regularisers. Refer to Garnelo et al. (2018b) for the derivation of the loss presented in 13.

N_i^C	deterministic	variational (Garnelo et al., 2018b)	variational (Poddar et al., 2024)
0	49.9 ± 0.3	50.0 ± 0.1	50.3 ± 0.2
1	52.1 ± 0.3	54.6 ± 0.2	55.5 ± 0.2
3	60.2 ± 0.3	61.4 ± 0.3	61.1 ± 0.3
5	65.6 ± 0.3	65.0 ± 0.3	63.9 ± 0.2
10	68.7 ± 0.2	69.4 ± 0.2	67.1 ± 0.2

(a) The HH dataset

N_i^C	deterministic	variational (Garnelo et al., 2018b)	variational (Poddar et al., 2024)
0	51.6 ± 0.2	50.9 ± 0.1	52.1 ± 0.2
1	55.1 ± 0.2	54.5 ± 0.1	55.5 ± 0.2
3	61.4 ± 0.2	59.7 ± 0.2	60.2 ± 0.2
5	65.4 ± 0.2	64.0 ± 0.2	63.3 ± 0.2
10	68.5 ± 0.2	69.9 ± 0.2	67.1 ± 0.2

(b) The HHT dataset.

Table 5: Deterministic vs. variational encoders.

As illustrated in Table 5, all three versions of the encoders perform similarly, with a slight advantage of the Neural Process version of variational encoding over the one suggested by Poddar et al. (2024). Given these results and the simplicity in training of the deterministic encoder, we decided to employ it as default for our NP-BTL model.

A.5. Few-shot learning with noisy data

The data for the experiment presented in the main body of this paper has been generated with no aleatoric noise in the preference choices given the knowledge of the ground truth value of z_i^*. To test our method's ability to learn from noisy preference choices, we have conducted additional experiments where the preference choices are sampled from the BTL model, with the level of noise controlled by the temperature parameter T in the sigmoid function, namely:

$$p(y_1 \succ y_2 | z_i^*) = \frac{\exp(\frac{1}{T} r(y_1; z_i^*))}{\exp(\frac{1}{T} r(y_1; z_i^*)) + \exp(\frac{1}{T} r(y_2; z_i^*))} \tag{15}$$

Table 6 demonstrates that, as anticipated, with increasing levels of noise in preference choices (i.e. increasing value of T), the accuracy of the NP-BTL model decreases. Nevertheless, the model does retain its ability to perform few-shot preference learning over the noisily generated data: its performance improves significantly with increasing the number of example provided in context. For reference, the BTL (mixed) models obtain accuracy equivalent to a random guess in all settings.

T N_i^C	0.0	0.01	0.1	0.7	1.0
0	49.6 ± 0.4	50.1 ± 0.2	50.2 ± 0.2	49.4 ± 0.2	49.7 ± 0.2
1	51.2 ± 0.3	55.8 ± 0.3	55.4 ± 0.3	51.6 ± 0.2	51.5 ± 0.2
3	60.7 ± 0.3	62.5 ± 0.3	61.7 ± 0.3	56.6 ± 0.2	55.9 ± 0.2
5	65.9 ± 0.2	65.6 ± 0.2	65.0 ± 0.3	60.6 ± 0.2	58.5 ± 0.2
10	69.0 ± 0.2	67.9 ± 0.2	67.5 ± 0.2	63.6 ± 0.2	61.0 ± 0.2

(a) The HH dataset.

T N_i^C	0.0	0.01	0.1	0.7	1.0
0	51.5 ± 0.2	51.2 ± 0.2	52.4 ± 0.2	50.6 ± 0.2	51.3 ± 0.2
1	55.0 ± 0.2	55.1 ± 0.2	56.1 ± 0.2	52.7 ± 0.2	52.9 ± 0.2
3	61.1 ± 0.2	61.1 ± 0.2	62.0 ± 0.2	57.6 ± 0.2	56.7 ± 0.2
5	65.4 ± 0.2	64.1 ± 0.2	65.6 ± 0.2	61.6 ± 0.2	60.6 ± 0.2
10	69.0 ± 0.2	67.6 ± 0.2	68.7 ± 0.2	65.5 ± 0.2	64.0 ± 0.2

(b) The HHT dataset.

Table 6: Results on the HH and HHT datasets generated with noisy preference choices. $T = 0.0$ corresponds to the deterministic choices as in the main experiments.

A.6. The HelpSteer Dataset

To demonstrate our method's generalisability to latent factors other than those considered by the UltraFeedback dataset, we also conducted experiments with a new dataset obtained by combining the HelpSteer[8] and HelpSteer2[9] datasets. These datasets support aligning models to become more helpful, factually correct and coherent while being adjustable regarding the complexity and verbosity of their responses. To reflect this objective, we consider a setup where users behave with respect to one of the four objectives described by the following reward function:

$$r(y, z_i^*) = \begin{cases} \text{overall}(y) + \text{complexity}(y) + \text{verbosity}(y), & \text{if } z_i^* = 0 \\ \text{overall}(y) + \text{complexity}(y) - \text{verbosity}(y), & \text{if } z_i^* = 1 \\ \text{overall}(y) - \text{complexity}(y) + \text{verbosity}(y), & \text{if } z_i^* = 2 \\ \text{overall}(y) - \text{complexity}(y) - \text{verbosity}(y), & \text{if } z_i^* = 3 \end{cases}, \tag{16}$$

where $\text{overall}(y) = \text{helpfulness}(y) + \text{correctnes}(y) + \text{coherence}(y)$, and $z_i^* \sim \text{Multinomial}([0.25, 0.25, 0.25, 0.25])$. Preference choices are generated with a deterministic choice function based on the reward difference. We have created two versions of this dataset:

- High conflict: a dataset in which, for every sample, at least one reward function assigns a different preference choice than the others.
- Low conflict: A dataset in which, for 50% of samples, at least one reward function assigns a different preference choice than the others, while for the remaining 50%, all four reward functions yield the same preference choice.

We report the results for the NP-BTL and the BTL (mixed) models under the two variations in the conflict rates in Table 7.

model N_i^C	BTL (mixed)	NP-BTL
0	70.2 ± 0.8	69.0 ± 0.6
1	-	74.4 ± 0.5
3	-	84.2 ± 0.4
5	-	86.1 ± 0.4
10	-	89.0 ± 0.3

(a) The high conflict version.

model N_i^C	BTL (mixed)	NP-BTL
0	71.0 ± 0.2	71.0 ± 0.3
1	-	72.1 ± 0.3
3	-	74.6 ± 0.2
5	-	75.8 ± 0.2
10	-	76.1 ± 0.2

(b) The low conflict version.

Table 7: Accuracy of the NP-BTL vs. BTL (mixed) on the HelpSteer datasets.

[8] https://huggingface.co/datasets/nvidia/HelpSteer
[9] https://huggingface.co/datasets/nvidia/HelpSteer2

B. Analytic study: proof

Suppose $y = [h_0, h_1] \sim \mathcal{N}(\mu, \Sigma)$, where $\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$. Let $Z^* \sim \text{Bernoulli}(0.5)$, and

$$r(y; Z^*) = \begin{cases} h_0 & \text{if } Z^* = 0 \\ h_1 & \text{if } Z^* = 1 \end{cases} \tag{17}$$

We assume that decisions are made according to

$$\mathbb{P}(y_1 \succ y_2) = \begin{cases} 1 & \text{if } r(y_1; Z^*) > r(y_2; Z^*) \\ 0 & \text{if } r(y_1; Z^*) < r(y_2; Z^*) \end{cases} \tag{18}$$

We define a Bayes optimal classifier predicting the value of Z^*, given an observed dataset $\mathcal{D}^C$ as :

$$\hat{Z}(\mathcal{D}^C) = \arg\max_{k \in \{0,1\}} \mathbb{P}(Z^* = k | \mathcal{D}^C), \tag{19}$$

where in case $\mathbb{P}(Z^* = 0|\mathcal{D}^C) = \mathbb{P}(Z^* = 1|\mathcal{D}^C)$, we let $\hat{Z}(\mathcal{D}^C) \sim \text{Bernouli}(0.5)$.

We look at the expected error rate of $\hat{Z}(\mathcal{D}^C)$. We have that:

$$\begin{aligned}
\mathbb{E}\left[\hat{Z}(\mathcal{D}^C) \neq Z^*\right] &= \mathbb{P}\left(\hat{Z}(\mathcal{D}^C) \neq Z^*\right) \\
&= \mathbb{P}\left(\hat{Z}(\mathcal{D}^C) = 1, Z^* = 0\right) + \mathbb{P}\left(\hat{Z}(\mathcal{D}^C) = 0, Z^* = 1\right) \\
&= 2\mathbb{P}\left(\hat{Z}(\mathcal{D}^C) = 1, Z^* = 0\right) && \text{(by symmetry)} \\
&= \mathbb{P}\left(\hat{Z}(\mathcal{D}^C) = 1 | Z^* = 0\right)
\end{aligned}$$

Let us now consider a special case of a single contextual sample, such that $\mathcal{D}^C = \{(y^w, y^\ell)\}$, where $y^w = [h_0^w, h_1^w]$, $y^\ell = [h_0^\ell, h_1^\ell]$ are sampled i.i.d from $\mathcal{N}(\mu, \Sigma)$. We then have, for $k \in \{0, 1\}$

$$\mathbb{P}\left(Z^* = k|\mathcal{D}^C\right) \propto \mathbb{P}(y^w \succ y^\ell | Z^* = k)\mathbb{P}(Z^* = k) \tag{20}$$

Therefore,

$$\mathbb{P}\left(Z^* = 0|\mathcal{D}^C\right) = \begin{cases} 0.5 & \text{if } h_0^w > h_0^\ell \text{ and } h_1^w > h_1^\ell \\ 1.0 & \text{if } h_0^w > h_0^\ell \text{ and } h_1^w < h_1^\ell \\ 0 & \text{otherwise} \end{cases} \quad \mathbb{P}\left(Z^* = 1|\mathcal{D}^C\right) = \begin{cases} 0.5 & \text{if } h_1^w > h_1^\ell \text{ and } h_0^w > h_0^\ell \\ 1.0 & \text{if } h_1^w > h_1^\ell \text{ and } h_0^w < h_0^\ell \\ 0 & \text{otherwise} \end{cases}$$

The event $\hat{Z}(\mathcal{D}^C) = 1$ can only happen in two cases: a) it happens with a probability of 1 if $\mathbb{P}(Z^* = 1|\mathcal{D}^C) > \mathbb{P}(Z^* = 0|\mathcal{D}^C)$ b) it happens with a probability of 0.5 when $\mathbb{P}(Z^* = 1|\mathcal{D}^C) = \mathbb{P}(Z^* = 0|\mathcal{D}^C)$. Conditioned on $Z^* = 0$, we must necessarily have $h_0^w > h_0^\ell$. This in turn implies that $\hat{Z}(\mathcal{D}^C) = 1$ can only happen in case b), i.e. when $\mathbb{P}(Z^* = 1|\mathcal{D}^C) = \mathbb{P}(Z^* = 0|\mathcal{D}^C) = 0.5$, which happens if and only if $h_1^w > h_1^\ell$ and $h_0^w > h_0^\ell$. Thus,

$$\mathbb{P}\left(\hat{Z}(\mathcal{D}^C) \neq Z^*\right) = \frac{1}{2}\mathbb{P}(h_1^w > h_1^\ell | h_0^w > h_0^\ell) \tag{21}$$

To compute this probability we define $X = [X_0, X_1] = [h_0^w - h_0^\ell, h_1^w - h_1^\ell]$. We have that $X \sim \mathcal{N}(\mathbf{0}, \Sigma)$ and our probability of interest is equal to

$$\mathbb{P}\left(\hat{Z}(\mathcal{D}^C) \neq Z^*\right) = \frac{1}{2}\mathbb{P}(X_1 > 0 | X_0 > 0) = \mathbb{P}(X_1 > 0, X_0 > 0), \tag{22}$$

which by a standard result based on the geometry of the Gaussian distribution is equal $\frac{1}{4} + \frac{1}{2\pi}\arcsin(\rho)$.

□

C. Complexity Analysis of NP-BTL and NP-DPO

Training. Several operations influence the time complexity per iteration of the algorithm. First, sampling a user i from the dataset is an $O(1)$ operation. Sampling the context set $\mathcal{D}_i^C$ and the target set $\mathcal{D}_i^T$ from the user-specific dataset $\mathcal{D}_i$ requires again $O(1)$. During training, the number of context data points N_i^C for each task varies between $N^{C,min}$ and $N^{C,max}$. The number of target data points N^T is constant for all tasks.

At each forward pass, the algorithm computes the conditional latent variable z_i^C for the sampled context datasets $\mathcal{D}_i^C$. This operation takes $O(N_i^C \cdot f_{enc})$ where f_{enc} is the complexity of a forward pass of the encoder network. The loss, i.e. negative log-likelihood of the target pairs $\mathcal{D}_i^T$, is computed with a forward pass of all target pairs $(y_{i,j}^w, y_{i,j}^\ell) \in \mathcal{D}_i^T$ and the latent variable z_i^C through the decoder network. This takes $O(N^T \cdot f_{dec})$, where f_{dec} is the complexity of one forward pass through the decoder network.

Parameters of the model are optimised with a version of stochastic gradient descent run for a maximum of T steps. Thus, the overall time complexity of training an NP-BTL or NP-DPO model is:

$$O(T \cdot (N^{C,max} \cdot f_{enc} + N^T \cdot f_{dec})),$$

which is linear in the number of iterations T, the number of target pairs N^T, and the maximum number of context pairs $N^{C,max}$.

Inference. To infer the reward for a single option $y \in \mathcal{Y}$, given a contextual sample $\mathcal{D}_i^C = \{(y_{i,j}^w, y_{i,j}^\ell)\}_{j=1}^{N_i^C}$, the model first computes the contextual embedding z_i^C and then passes it through the decoder network together with the option y. Thus, time complexity of inference is $O(N_i^C \cdot f_{enc} + f_{dec})$.

D. Implementation and experimental details

All experiments are implemented in Python, using the PyTorch (Paszke et al., 2017) and HuggingFace libraries (Wolf et al., 2020). All experiments can be run on a machine with a single A100 Nvidia GPU. We provide the code implementing the NP-BTL and NP-DPO models alongside the instructions needed to reproduce the key experiments in this repository: anonymous.4open.science/r/few-shot-preference-learning-5CA1/.

All reward and LLM policies are trained using the Adam optimiser (Kingma & Ba, 2014).

Unless otherwise stated, all mentions of MLPs are implemented as a two-layer neural networks with a hidden dimension of 256 and GELU activation.

Following the convention of NP training (Garnelo et al., 2018a;b), all tasks $\tau = (\mathcal{D}^C, \mathcal{D}^T)$ are constructed such that context pairs are a subset of the target pairs, i.e. $\mathcal{D}^C \subseteq \mathcal{D}^T$. We only report the unseen pairs' accuracies during model evaluation: $\mathcal{D}^T \setminus \mathcal{D}^C$.

D.1. Illustrative comparison of existing approaches

Dataset. In this experiment all options $y \in \mathcal{Y}$ are represented in $\mathcal{Y} = [-1, 1]$. The dataset is constructed by first generating 20k pairs sampled uniformly from $[-1, 1]^2$. This dataset is split into 10k training, 5k validation and 5k testing pairs. To construct a single task $\tau_i = (\mathcal{D}_i^C, \mathcal{D}_i^T)$, we first sample the unobservable $z_i^* \sim \text{Bernoulli}(0.5)$. Then, we sample N_i^C context and N^T target pairs from the training, validation, or testing split. For each pair in the context and target datasets, the preference choices are determined deterministically so that $p(y_1 \succ y_2) = \mathbb{1}\{r(y_1, z^*) > r(y_2, z^*)\}$, where

$$r(y, z^*) = x \ \text{ for } \ y \leq 0.5 \quad \text{and} \tag{23}$$

$$r(y, z^*) = \begin{cases} 2y & \text{if } z^* = 0 \\ 1 - y & \text{if } z^* = 1 \end{cases} \ \text{for } y > 0.5 \tag{24}$$

The number of context pairs varies uniformly between $N^{C,min} = 0$ and $N^{C,max} = 10$. The number of target pairs is set to $N^T = 20$.

NP-BTL implementation. The encoder network h_{ϕ_e} is implemented as a DeepSet (Zaheer et al., 2018), with the inner and outer encoders implemented as MLPs: $h_{\phi_{e,1}} : \mathbb{R}^2 \to \mathbb{R}^{256}$, $h_{\phi_{e,2}} : \mathbb{R}^{256} \to \mathbb{R}^{256}$. The inputs to the inner network are preference pairs from the context dataset $(y^w, y^\ell) \in \mathcal{D}^C$ concatenated together. The decoder $g_{\phi_d} : \mathbb{R}^{256+1} \to \mathbb{R}$, first maps a single target option $y \in \mathcal{D}^T$ to a 256-dimensional embedding e_y with an MLP. Then, e_y is concatenated with the embedding of the context dataset, $z^C = h_\phi(\mathcal{D}^C)$, and passed through another MLP to obtain the value of the conditional reward $r_\phi(y|\mathcal{D}_i^C) \in \mathbb{R}$, i.e.:

$$r_\phi(y|\mathcal{D}_i^C) = g_{\phi_d}(y, z_i^C) = \mathrm{MLP}_2\left(\left[\mathrm{MLP}_1(y)||z_i^C\right]\right)$$

Baseline implementation. Both the BTL and DPL models are implemented as MLPs. For the BTL model, the 1-dimensional output represents the reward value of a single option $r_\phi(y)$. For the DPL model, the output is 2-dimensional, representing the mean and the logarithm of the standard deviation of a normal distribution $\mathcal{N}(\mu(y), \sigma(y))$, describing the distribution of the likely values that $r_\phi(y)$ takes.

Training. Each batch consists of 64 tasks containing the same number of context data points $(\mathcal{D}_i^C, \mathcal{D}_i^T)$ (for the non-conditional baselines we have $\mathcal{D}_i^C = \varnothing$). Models are trained for a maximum of 50 SGD updates, retaining the model parameters with the lowest loss on the validation split. The learning rate is set to 1e-4.

D.2. Experiments with the UltraFeedback dataset

D.2.1. The datasets

Helpful-Honest dataset. The basis of this experiment is the UltraFeedback[10] dataset. Each option $y = [x, \tilde{y}]$ within this dataset is represented with a user prompt x and a candidate LLM response $\tilde{y}$. Alongside the set of prompts and candidate responses, each response in this dataset is assigned a score in four categories: honesty, helpfulness, truthfulness and instruction following. To simulate users with varying preferences, we define two reward functions:

$$r(y, z^*) = \begin{cases} \mathrm{honesty}(\tilde{y}) & \text{if } z^* = 0 \\ \mathrm{helpfulness}(\tilde{y}) & \text{if } z^* = 1 \end{cases} \tag{25}$$

To construct the dataset with 100% conflicting pairs, we filter the dataset to pairs of options for which the response made with $r(\cdot; 0)$ differs from the response made with $r(\cdot; 1)$. As a result, our final dataset consists of 28763 training, 3673 validation, and 3509 testing pairs (y_1, y_2).

Helpful-Honest-Truthful dataset. This dataset is generated in an analogous manner, but instead, the reward functions are determined by three instead of two factors. We let:

$$r(y, z^*) = \begin{cases} \mathrm{honesty}(\tilde{y}) & \text{if } z^* = 0 \\ \mathrm{helpfulness}(\tilde{y}) & \text{if } z^* = 1 \\ \mathrm{truthfulness}(\tilde{y}) & \text{if } z^* = 2 \end{cases} \tag{26}$$

where $z^* \sim \mathrm{Multinomial}([1/3, 1/3, 1/3])$. To ensure high informativeness of contextual samples, we filter the dataset to pairs y_1, y_2 for which one choice is made according to $r(\cdot\,; 0)$, $r(\cdot\,; 1)$, $r(\cdot\,; 2)$ is distinct from the others The resulting dataset consists of 27475 training, 3415 validation, and 3415 testing pairs.

Training and testing. The raw representation of prompts and responses is in natural language. For computational efficiency, we represent them as frozen embeddings pre-computed with the `LLama-3-8B`[11] language model. These embeddings are obtained as the last hidden state representation of the last token in the tokenized representation of y. We denote the embeddings of $y_{i,j}^w, y_{i,j}^\ell$ as $e_{i,j}^w, e_{i,j}^\ell \in \mathbb{R}^{4096}$. During training, validation, and testing, we first sample $z_i^* \sim \mathrm{Bernoulli}(0.5)$ and then sample N_i^C context pairs and N^T target pairs with decisions made deterministically according to $p(y_1 \succ y_2) = \mathbb{1}\{r(y_1, z_i^*) > r(y_2, z_i^*)\}$.

During training, the number of context pairs for reward modelling varies uniformly between $N^{C,min} = 0$ and $N^{C,max} = 10$. The number of target pairs is set to $N^T = 20$.

For LLM policy fine-tuning, during training, the number of context pairs varies uniformly between $N^{C,min} = 0$ and $N^{C,max} = 6$. The number of target pairs is set to $N^T = 9$.

[10]https://huggingface.co/datasets/openbmb/UltraFeedback
[11]https://huggingface.co/meta-llama/Meta-Llama-3-8B

D.2.2. Reward Models

NP-BTL. Our context encoder is implemented as a DeepSet with multi-head self-attention. For $\mathcal{D}_i^C = \{(y_{i,j}^w, y_{i,j}^\ell)\}_{j=1}^{N_i^C}$ the contextual embedding z_i^C is computed as:

$$z_i^C = h_{\phi_{e,2}}\left(\sum_{j=1}^{N_i^C} \text{MultiHeadAttn}(W^T[e_{i,j}^w || e_{i,j}^\ell])\right), \tag{27}$$

where $W \in \mathbb{R}^{2\cdot 4096\times 256}$ is a linear map, $h_{\phi_{e,2}} : \mathbb{R}^{256} \to \mathbb{R}^{256}$ is an MLP, and MultiHeadAttn a multi-head attention layer with 8 heads and dropout 0.1 [12].

The decoder $g_{\phi_d} : \mathbb{R}^{256+256} \to \mathbb{R}$, first passess the embedding e of a target option $y \in \mathcal{D}^T$ through an MLP, then concatenates the output with z_i^C, and passess it through another MLP so that:

$$r_\phi(y|\mathcal{D}_i^C) = g_{\phi_d}(y, z_i^C) = \text{MLP}_2\left(\left[\text{MLP}_1(e)||z_i^C\right]\right) \tag{28}$$

BTL. The BTL model is a simple MLP mapping the embedding $e \in \mathbb{R}^{4096}$ to scalar rewards in $\mathbb{R}$.

Training. Each batch consists of 64 tasks containing the same number of context data points $(\mathcal{D}_i^C, \mathcal{D}_i^T)$ (for the non-conditional baselines we have $\mathcal{D}_i^C = \varnothing$). Models are trained for a maximum of 450 SGD updates, retaining the model parameters with the lowest loss on the validation split. The learning rate is set to 1e-4.

D.2.3. LLM Policy Models

NP-DPO. Contextual embeddings z_i^C are obtained in the same way as in reward modelling with NP-BTL. A key component in this task is modulating the LLM parameters θ with the conditional latent variable z_i^C. Our modulation operator is based on feature-wise linear modulation (FiLM) (Perez et al., 2017). We map z_i^C to a set of modulating parameters $\tau_i = \{\gamma_m(z_i^C), \beta_m(z_i^C)\}_{m=1}$, where L is equal to the number of attention blocks. The shorthand notation $\theta \odot \tau_i$ introduced in the main body of the paper denotes the operation of rescaling the output of each attention block by the parameters $\gamma_m, \beta_m \in \mathbb{R}^{d_{model}}$. If we denote by o_m the output of the attention block at the m^{th} layer, then hidden representations modulated via z_i^C are obtained as:

$$o'_m = \gamma_m \otimes o_m + \beta_m, \tag{29}$$

where $\otimes$ denotes element-wise product. The parameters θ of the base LLM and the collection of the weights $\phi = \{\phi_{mod}, \phi_e\}$ defining the modulating hypernetwork $h_{\phi_{mod}}$ and the context encoder h_{ϕ_e} are optimised jointly by maximising the conditional DPO objective:

$$\max_{\theta,\phi} \mathop{\mathbb{E}}_{\substack{(\mathcal{D}_i^C, \mathcal{D}_i^T),\\ (x,\tilde{y}^w,\tilde{y}^\ell)\in\mathcal{D}_i^T}} \left[\log\sigma\left(\beta\log\frac{\pi_\theta(\tilde{y}^w|x,\mathcal{D}_i^C)}{\pi_{\text{ref}}(\tilde{y}^w|x)} - \beta\log\frac{\pi_\theta(\tilde{y}^\ell|x,\mathcal{D}_i^C)}{\pi_{\text{ref}}(\tilde{y}^\ell|x)}\right)\right]. \tag{30}$$

BTL-DPO. We optimise the parameters θ of the chosen language model with the standard DPO objective:

$$\max_{\theta} \mathop{\mathbb{E}}_{(x,\tilde{y}^w,\tilde{y}^\ell)} \left[\log\sigma\left(\beta\log\frac{\pi_\theta(\tilde{y}^w|x)}{\pi_{\text{ref}}(\tilde{y}^w|x)} - \beta\log\frac{\pi_\theta(\tilde{y}^\ell|x)}{\pi_{\text{ref}}(\tilde{y}^\ell|x)}\right)\right]. \tag{31}$$

Training. To reduce memory requirements, parameters θ of the base LLM are optimised with Low-Rank Approximation (LoRA) (Hu et al., 2021), with a rank of 512 and a LoRA-α of 1024. The learning rate of the Adam optimizer is set to 1e-6. The parameter β is set to 0.05. During training, we use a batch size of 1.

[12] https://pytorch.org/docs/stable/generated/torch.nn.MultiheadAttention

D.3. Correlations in data and identifiability of the hidden context

D.3.1. Analytic study

Dataset. Options $y = [h_0, h_1] \in \mathbb{R}^2$ are sampled with $\mathcal{N}(\mu, \Sigma)$ with $\mu = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and $\Sigma = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$. For any two pairs $y_1, y_2 \in \mathbb{R}^2$, decisions are made according to $p(y_1 \succ y_2) = \mathbb{1}\{r(y_1, z^*) > r(y_2, z^*)\}$, where

$$r(y, z^*) = \begin{cases} h_0 & \text{if } z^* = 0 \\ h_1 & \text{if } z^* = 1 \end{cases} \tag{32}$$

The dataset is constructed by first generating 20k pairs $y_1, y_2 \in \mathbb{R}^2$. This dataset is split into 10k training, 5k validation and 5k testing pairs. To construct a single task $\tau_i = (\mathcal{D}_i^C, \mathcal{D}_i^T)$, we first sample the unobservable $z_i^* \sim \text{Bernoulli}(0.5)$ and then sample N_i^C context and N^T target pairs from the training, validation, or testing split with preference choices made according to $r(\cdot\,; z_i^*)$. The number of context pairs varies uniformly between $N^{C,min} = 0$ and $N^{C,max} = 10$. The number of target pairs is set to $N^T = 20$.

Model implementation and training. Implementation of the NP-BTL and the baseline BTL reward models is analogous to that presented in D.1, with the input dimension replaced from 1 to 2. The training setup is identical to D.1.

E. Related Work

We extend our comparison to related work in Table 8, where we investigate the works on (1) whether the source of heterogeneity is observable or not, (2) whether and how contextual information is provided, (3) whether it generalises to new and unseen users, (4) what kind of steerable model they investigate, (5) if the number of learnable rewards and policies is finite or unbounded.

Model	**Source of Heterogeneity**	**Contextual Information**	**Generalises to New Users**	**Steerable Model Type**	**Number of Learnable Rewards / Policies**
P-SOUPS (Jang et al., 2023)	Observable	Preference prompt	?	Reward	Finite, pre-determined
CLP (Wang et al., 2024d)	Observable	User-specific weight vector	✓	Policy	Finite, pre-determined
P-RM (Li et al., 2024)	Observable	User-specific ID vector	✗	Reward	Finite, pre-determined
P-DPO (Li et al., 2024)	Observable	User-specific ID vector	✗	Policy	Finite, pre-determined
DPA (Wang et al., 2024a)	Observable	User-specific weight vector	✓	Reward	Unbounded
CPO (Guo et al., 2024a)	Observable	User-specific weight vector	✓	Policy	Unbounded
PAL (Chen et al., 2024)	Semi-observable	User-specific feature vector	✓	Reward	Unbounded
VPL (Poddar et al., 2024)	Unobservable	User preference data $\mathcal{D}_i^C$ (fixed size)	✓*	Reward	Unbounded
GPO (Zhao et al., 2023)	Unobservable	User preference data $\mathcal{D}_i^C$	✓	Reward	Unbounded
Pplug (Liu et al., 2024)	Unobservable	Historical user interaction data	✓	Policy	Unbounded
NP-BTL (ours)	Unobservable	User preference data $\mathcal{D}_i^C$	✓	Reward	Unbounded
NP-DPO (ours)	Unobservable	User preference data $\mathcal{D}_i^C$	✓	Policy	Unbounded

Table 8: *Comparison of the related work.* (*) Likely yes, but not demonstrated in the paper.