PAL: Sample-Efficient Personalized Reward Modeling for Pluralistic Alignment

Daiwei Chen, Yi Chen, Aniket Rege, Zhi Wang, Ramya Korlakai Vinayak
University of Wisconsin-Madison
Madison, Wisconsin, USA
daiwei.chen@wisc.edu

Abstract

*Foundation models trained on internet-scale data benefit from extensive alignment to human preferences before deployment. However, existing methods typically assume a homogeneous preference shared by all individuals, overlooking the diversity inherent in human values. In this work, we propose a general reward modeling framework for pluralistic alignment (**PAL**), which incorporates diverse preferences from the ground up. PAL has a modular design that leverages commonalities across users while catering to individual personalization, enabling efficient few-shot localization of preferences for new users. Extensive empirical evaluation demonstrates that **PAL matches or outperforms state-of-the-art methods on both text-to-text and text-to-image tasks**: on Reddit TL;DR Summary, PAL is 1.7% more accurate for seen users and 36% more accurate for unseen users compared to the previous best method,* **with** *100×* **less parameters**. *On Pick-a-Pic v2, PAL is 2.5% more accurate than the best method with 156× fewer learned parameters. Finally, we provide theoretical analysis for generalization of rewards learned via PAL showcasing the reduction in number of samples needed per user.*

1. Introduction

Foundation models trained on internet data are often not readily deployable and undergo *alignment* to human preferences using large amounts of *pairwise comparison* feedback [44]. While aligning AI/ML models to human preferences, it is important to consider *whose preferences are we aligning them to*? The status quo for popular alignment frameworks is to assume a homogeneous preference shared by all humans. However, humans have diverse preferences, values and opinions [6, 20, 42, 69]. The need to capture this *plurality* in the context of AI alignment was also highlighted recently by [61]. However, the methods suggested therein and other recent works look at learning multiple rewards with a top-down approach, where the system designer decides the number and axes that one should care about [15, 33, 44, 53], e.g., helpfulness vs. harmlessness [4, 5, 24, 49]. In reality, human preferences are more complex than designer-specified axes [6], especially on subjective aspects, which leads us to propose the following goal.

Goal: Develop a sample-efficient, personalized reward modeling framework for pluralistic alignment from the ground up which learns and generalizes to heterogeneous preferences.

Our Contributions.

1. We propose **PAL**, a **novel, sample-efficient, personalizable reward modeling framework for pluralistic alignment** from the ground up (Section 2). Our modular and versatile framework achieves **superior performance to the state-of-the-art** (SoTA) in both vision (Section 3) and language (Appendix D.3) tasks, *while utilizing only a fraction of their learnable parameters*.
2. PAL reward models **achieve competitive performance with simple 2-layer MLPs** on top of frozen foundation models of varying sizes **in practical settings across modalities** (Section 3). PAL enables democratic alignment via strong accuracy-compute optimality [50].
3. PAL is complementary to existing alignment frameworks, and works seamlessly across compute budgets (see Figure 2) demonstrating the strength of its **flexible and modular design**.
4. We provide **sample complexity guarantees for generalization** towards new preference predictions for users in the dataset as well as for unseen users via few-shot learning, in the fully connected linear layer setting for one of our models (Appendix C.1), and we verify these results with extensive numerical simulations (Appendix C.2).

The learned PAL reward models can be used flexibly for personalizing downstream task of alignment through (i) train-time methods such as PPO-based RLHF [16, 70] and (ii) inference time methods via best-of-n sampling such as controlled decoding [37, 41]. In addition, the modular nature of PAL has high potential for future adaptability as it enables seamlessly updating learned reward models via switching data encoders and distance metrics, or adding

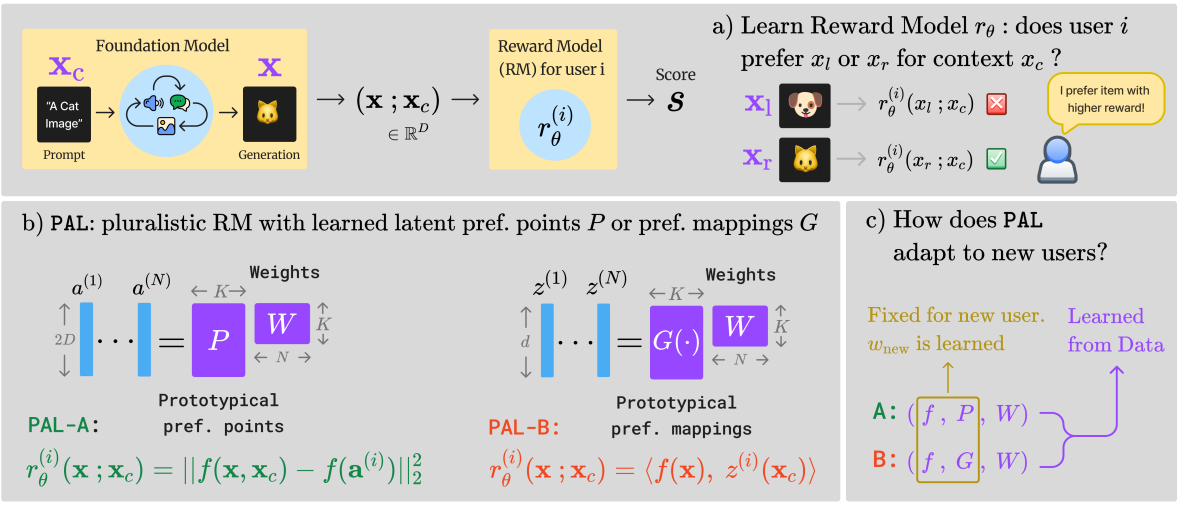


Figure 1. (a) Using preference data, the PAL framework learns a personalized reward model for each user i, $r_\theta^{(i)}(\cdot)$, which captures the user's preference for any output $\mathbf{x}$ given context $\mathbf{x}_c$. (b) PAL models the common perceptions of similarity across users through a shared representation $f(\cdot)$, and represents the individual aspects of preferences via either a preference point $\mathbf{a}^{(i)}$ in PAL-A or a preference mapping $z^{(i)}(\mathbf{x}_c)$ in PAL-B. In particular, we assume a *low-rank* structure with K prototypical preference points or preference mappings; see Section 2 for details. (c) PAL enables efficient few-shot preference learning for a new user—only a K-dimensional weight vector is learned. This reduces computational cost as well as data needed for generalization.

new prototypes to account for dynamically changing heterogeneous preferences.

Background. Our reward modeling builds on the popular preference learning models, the ***Bradley-Terry-Luce (BTL) model*** [10] and the ***ideal point model*** [17], both of which are special cases of the *linear stochastic transitivity* (LST) models. We provide a brief discussion of these models, including their assumptions and limitations here. Let D denote the dimension of the representation space of the foundation models. Given a context/prompt $\mathbf{x}_c \in \mathbb{R}^D$, generative model(s) can produce different outputs, denoted by $\mathbf{x} \in \mathbb{R}^D$.[1] The LST models make the following assumption: Suppose each output for the prompt is associated with a *true* score $s^\star(\mathbf{x}; \mathbf{x}_c) \in \mathbb{R}$. Given any list of alternates, the true scores lead to a *true* ranking of them: for any two alternates, $\mathbf{x}_l$ is preferred over $\mathbf{x}_r$ if $s^\star(\mathbf{x}_l; \mathbf{x}_c) > s^\star(\mathbf{x}_r; \mathbf{x}_c)$. However, when we elicit comparison feedback from humans, the answers may be *noisy* and are modeled as, $\Pr(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c) = h(s^\star(\mathbf{x}_l; \mathbf{x}_c) - s^\star(\mathbf{x}_r; \mathbf{x}_c)))$, where $h : \mathbb{R} \to [0, 1]$ is a strictly monotonic *link function* that satisfies $h(z) = 1 - h(-z)$. In other words, $\Pr(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c) = 1/2$ when $s^\star(\mathbf{x}_l, \mathbf{x}_c) = s^\star(\mathbf{x}_r, \mathbf{x}_c)$ and $\Pr(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c) > 1/2$ when $s^\star(\mathbf{x}_l, \mathbf{x}_c) > s^\star(\mathbf{x}_r, \mathbf{x}_c)$.

The BTL model uses the logistic sigmoid function as the link function:

$$\Pr(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c) = \frac{1}{1 + \exp\left(s^\star(\mathbf{x}_r; \mathbf{x}_c) - s^\star(\mathbf{x}_l; \mathbf{x}_c)\right)}. \quad (1)$$

On the other hand, the ideal point model uses a latent vector $\mathbf{a} \in \mathbb{R}^D$ to denote the *ideal* preference point of a user, along with (negative) distance-based scoring function; the model is given by,

$$\Pr(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c) = h(\text{dist}^2(\mathbf{x}_r, \mathbf{a}) - \text{dist}^2(\mathbf{x}_l, \mathbf{a})), \quad (2)$$

where h can be any valid link function. The key idea here is that the larger the difference in distances between the alternates to the ideal point, the easier it is to choose between them, and hence the answer is less noisy. We note that with a sigmoid link function, the ideal point model can be viewed as the BTL model with a distance-based scoring function.

A key limitation of these approaches is that they *assume a single true ranking of the alternates* and model the differences in elicited preference as noisy observations. In reality, the differences in elicited preferences are not merely noise, instead a reflection of plurality of human preferences.

The ideal point model provides a natural starting point to incorporate individual preferences since each user can be modeled using their own latent preference. However, the assumption that we know the distance function that reflects a human notion of similarity of alternates can be too strong in practice. Further, completely individualized models devoid of any shared structure will lead to unnecessary burden on per user sample complexity for learning and would be difficult to generalize. Our aim is to capture the heterogeneity of preferences while also leveraging commonalities

[1] These representations can be taken from penultimate layer(s) of a foundation model. While we use the same D for the prompt and output for simplicity, this can easily be extended to different dimensional spaces.

(Section 2), allowing for sample efficient learning and generalization.

2. PAL: Reward Modeling for Pluralistic ALignment

In this section, we describe our proposed personalizable reward modeling framework that captures commonalities shared across the population which can be learned using the pooled data and individual aspects that is learned per user in a sample efficient way. For user i, given a context $\mathbf{x}_c$, the probability of alternate $\mathbf{x}_l$ being preferred over $\mathbf{x}_r$ is as follows,

$$\Pr(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c, i) = h^{(i)}(r_\theta^{(i)}(\mathbf{x}_r; \mathbf{x}_c) - r_\theta^{(i)}(\mathbf{x}_l; \mathbf{x}_c)), \quad (3)$$

where $h^{(i)}$ is any valid link function that can depend on the user, and $r^{(i)}(\cdot)$ is the personalized reward function for user i.[2] We do not assume the knowledge of the link function for our learning algorithms (Section 2.1). We propose two models for the personalized reward function[3]:

PAL-B: Diverse preferences modeled via latent preference mappings. Here each user's preference also incorporates the given context and is modeled using an *unknown preference mapping* $z : \mathbb{R}^D \to \mathbb{R}^d$. The shared sense of similarity of different alternates being compared is modeled as cosine similarity in an *unknown* mapped space, $f : \mathbb{R}^D \to \mathbb{S}^{d-1}$. The commonality in the preferences among users is captured by modeling each user's preference mapping as a convex combination of K prototypical mappings, i.e., $z^{(i)}(\mathbf{x}_c) = \sum_{k=1}^K w_k^{(i)} g_k(\mathbf{x}_c)$, where $\{g_1, ..., g_K\}$ with $g_k : \mathbb{R}^D \to \mathbb{S}^{d-1}$ are the prototypical mappings, $w_k^{(i)} \geq 0$ and $\sum_{k=1}^K w_k^{(i)} = 1$. Formally, the personalized reward function and the corresponding probabilistic preference model is given by,

$$\textbf{PAL-B:} \quad r_\theta^{(i)}(\mathbf{x}; \mathbf{x}_c) := \langle f(\mathbf{x}), z^{(i)}(\mathbf{x}_c) \rangle, \quad (4)$$

$$\Pr_i(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c) = h\left(\langle f(\mathbf{x}_r), z^{(i)}(\mathbf{x}_c) \rangle - \langle f(\mathbf{x}_l), z^{(i)}(\mathbf{x}_c) \rangle \right), \quad (5)$$

For any context $\mathbf{x}_c$, denoting $\mathbf{G}(\mathbf{x}_c) := [g_1(\mathbf{x}_c), \cdots, g_K(\mathbf{x}_c)]$, each user's preference mapping of the given context is, $z^{(i)}(\mathbf{x}_c) := \mathbf{G}(\mathbf{x}_c)\mathbf{w}^{(i)}$ with $\mathbf{w}^{(i)} \in \Delta^{K-1}$.

PAL modeling framework is **modular**, i.e., it provides a *systematic way* to incorporate *shared and personalized portions of preferences*, as well as transparent way to control multiple notions of complexity via cross validation: (i) the complexity of mapping f (and g's in PAL-B) captures the shared human notion of similarity between alternates. If the underlying foundation model used to obtain the representations of outputs and contexts ($\mathbf{x}$ and $\mathbf{x}_c$) are rich and semantically meaningful, then much smaller models suffice in capturing the rewards (Figure 2); (ii) the number of prototypes K capture the level of heterogeneity of human preferences in the dataset: more diverse preferences mandate a larger K for good generalization (Figure C.1(b)); and (iii) in conjunction with the prototypes, the individualized weights allow for personalization with much fewer samples per user for both seen and unseen users. This reduces the annotation burden on individuals during data collection (Figure C.1(c)) and the amount of samples needed for few-shot learning for new users arriving on deployment (Figure D.3). We illustrate the PAL framework in Figure 1 and Figure B.2 (Appendix B).

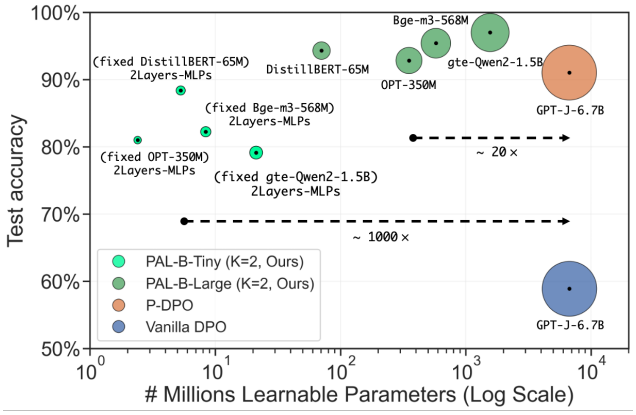


Figure 2. On Reddit TL;DR, PAL is accuracy-compute optimal and shows state-of-the-art (SoTA) performance.

2.1. Learning PAL models from Diverse Preferences

Let $\mathcal{D} := \left\{ \{(\mathbf{x}_l, \mathbf{x}_r; \mathbf{x}_c, y)_j^{(i)}\}_{j=1}^{m_i} \right\}_{i=1}^N$ be a dataset of preference comparisons, where m_i denotes the number of pairs answered by user i, and y is the answer given to the pair ($y = -1$ if $\mathbf{x}_l$ is preferred, $y = 1$ otherwise). This can be looked at as a supervised learning problem with binary labels. The **goal** of the learning algorithm in the PAL framework is to learn the mappings and prototypes shared across the population, and for each user i, the weights $\mathbf{w}^{(i)} \in \Delta^{K-1}$. For PAL-A, the mapping f and the prototypes $\{\mathbf{p}_k\}_{k=1}^K$ are shared, while for PAL-B, the mapping f and the prototype mappings $\{g_k\}_{k=1}^K$ are shared. These shared portions are learned using data pooled from all users while the user specific weights are learned using each individual user's preferences.

Given the comparison dataset $\mathcal{D}$, loss function $\ell : \mathbb{R} \to [0, 1]$, model class for f_θ and prototypical mappings $\{g_1, ..., g_K\}$, the learning algorithm for PAL-B starts

[2] We drop the superscript i on h denoting user specificity for simplicity in the rest of discussions, however, we note that the link function need not be the same for all users and our learning methods are agnostic to them.

[3] In Appendix B.2, we discuss when to choose one model over the other in practice.

by randomly initializing these functions, and user weights $\mathbf{w}^{(i)} \in \Delta^{K-1}$, $i = 1, ..., N$. Then, in each iteration until convergence criteria, the following steps are repeated,

- **Sample** a random mini-batch $\left\{(\mathbf{x}_l, \mathbf{x}_r; \mathbf{x}_c, y)_j^{(i)}\right\}$ of comparison data from $\mathcal{D}$.
- For each comparison j from user i:
  - **Compute user ideal mappings:** $z^{(i)}(\mathbf{x}_c) := [g_1(\mathbf{x}_c) \ \dots \ g_K(\mathbf{x}_c)] \cdot \mathbf{w}^{(i)}$.
  - **Compute distances**: $d_{l,j}^{(i)} = \langle f_\theta(\mathbf{x}_l), z^{(i)}(\mathbf{x}_c) \rangle$, $d_{r,j}^{(i)} = \langle f_\theta(\mathbf{x}_r), z^{(i)}(\mathbf{x}_c) \rangle$.
  - **Compute loss**: $\psi_j^{(i)}(\mathbf{x}_l, \mathbf{x}_r; \mathbf{x}_c, y) = \ell\left(y \cdot (d_{l,j}^{(i)} - d_{r,j}^{(i)})\right)$.
- **Update Step:** $\text{argmin}_{\theta, \{g_1, ..., g_K\}, \{\mathbf{w}^{(i)}\}_{i=1}^N} \sum_{i,j} \psi_j^{(i)}(\mathbf{x}_l, \mathbf{x}_r; \mathbf{x}_c, y)$.

Learning steps are similar for PAL-A. See Appendix B for pseudocode details.

2.2. Generalization of preference predictions for *seen* versus *unseen* users

When learning a reward function from diverse preferences, there are two types of generalization to consider: Predicting preferences for (1) *unseen pairs* for *seen users*, i.e., the people for whom the weights have already been learned from the training data; (2) *unseen users*, i.e., people whose data was not part of the training data at all. For such new users, a portion of their data will be used to learn weights to localize them within the learned model while keeping the shared mappings and prototypes fixed. We also note that the weighted combination of the prototypes, i.e., an average of all the seen users, can be used as the *zero-shot* ideal point for new users (e.g. Netflix recommendations for new accounts). However, we emphasize that it is important for reward functions to generalize to *unseen* users and PAL provides a natural way to localize new users, as we demonstrate in Appendix D.3. Appendix C.1 provides theoretical guarantees on the per-user sample complexity of PAL for few-shot generalization to unseen users.

3. Experiments on Real Datasets

We verify the following claims through extensive empirical evaluation:

C1. PAL can effectively capture the diversity of user preferences and outperform status-quo homogeneous reward models.

C2. Compared to existing pluralistic reward modeling methods, PAL can achieve state-of-the-art (SoTA) performance with significantly fewer parameters.

C3. PAL works for both text-to-text (T2T) and text-to-image (T2I) tasks.

We employ two strategies for defining and training the learnable mapping f between the embeddings from a foundation model and latent space. In the Tiny strategy, f is a simple two-layer MLP operating on a frozen foundation model. In the Large strategy, f is a combination of the foundation model and a two-layer MLP, with both components being learnable. Models employing these strategies are referred to as PAL-A-Tiny/ PAL-B-Tiny and PAL-A-Large/ PAL-B-Large respectively. Appendix D.1 describes the general training procedure of PAL.

Table 1. Seen user test accuracy of PAL compared to baselines on Pick-a-Pic v2. Entries with asterisk* have inflated accuracy due to the V2 test set overlap with V1 train (See dataset details in Section 3.1).

Model	Params	Trainset	V1 Test Accuracy (%)	V2 Test Accuracy (%)	
				No-Leakage	Leakage
CLIP-H/14	986M	-	59.23	62.57	58.59
ImageReward	447M	-	61.10	-	-
HPS v2.1	986M	-	66.70	-	-
PickScore	**986M**	V1	**71.85**	68.04	74.16*
PAL-A-Tiny (CLIP-H)	8.4M	V1	69.29 ± 0.6	-	-
PAL-B-Tiny (CLIP-H)	**6.3M**	V1	$\mathbf{71.13 \pm 0.3}$	$\mathbf{70.02 \pm 0.39}$	79.32 ± 1.68*
PAL-B-Tiny (CLIP-H)	6.3M	V2	-	$\mathbf{70.51 \pm 0.22}$	68.67 ± 0.51
PAL-B-Tiny (Pickscore)	6.3M	V2	-	70.16 ± 0.19	74.79 ± 0.13*

We perform experiments on datasets in different domains: the Reddit TL;DR Summary dataset (Appendix D.3) and the Pick-a-Pic dataset (Section 3.1). Due to limitations in currently available datasets for pluralistic alignment, which we highlight in Section 3.1, we also created semi-synthetic datasets, Pick-a-Filter (Section 3.2) and Persona (Appendix D.4), to further validate the above claims.

3.1. Pick-a-Pic (Text-to-Image)

In this section, we examine how agnostic PAL is to data modality (**C3**) via the Pick-a-Pic dataset [31]. Popular T2I reward models usually require fine-tuning foundation models with billions of parameters [31, 71, 74]. We show that PAL achieves competitive performance or SoTA performance while having significantly fewer parameters than baselines (**C2**).

Dataset. The Pick-a-Pic dataset [31] is a large, open dataset for human feedback in T2I generation. There are two versions of Pick-a-Pic, v1 and v2, where v2 extends v1. To ensure a fair model evaluation, we divide the v2 test set into "no-leakage" and "leakage" subsets due to overlap ("leakage") with the v1 train set. We only consider the 18391 samples with no preference ties, i.e. one generated image is always preferred to the other. Out of these, 10587 samples ($\sim 58\%$) overlap with the training and validation sets of v1, which was used to train PickScore [31] – we call this the v2 "leakage" subset. The remaining 7804 test samples ($\sim 42\%$) in v2 do not overlap with v1 train and val, ensuring they are distinct for evaluation purposes – we call this the v2 "no-leakage" subset.

Setup. We train PAL-B with logistic loss on both v1 and v2 for 10 epochs on top of CLIP-H/14 or PickScore [31] embeddings (details in Appendix D.5).

Baselines. We compare to SoTA PickScore [31] and a vanilla CLIP-H/14.

Results. Table 1 shows (a) PAL matches SoTA

Table 2. On Reddit TL;DR Summary, PAL-B-Large outperforms SoTA P-DPO on seen users (**+1.7%**) and unseen users (**+36%**) with 6.3 billion fewer parameters. Note that we use only 10 samples per unseen user to localize their weights.

Model	Seen Acc (%)	Unseen Acc (%)
Vanilla DPO	58.91	55.37
P-DPO Individual	91.04	55.34
P-DPO Cluster ($K = 5$)	91.12	54.55
PAL-B-Tiny ($K = 2$)	79.54 ± 0.54	74.72 ± 0.54
PAL-B-Large ($K = 2$)	$\mathbf{92.82 \pm 0.95}$	$\mathbf{91.63 \pm 0.54}$

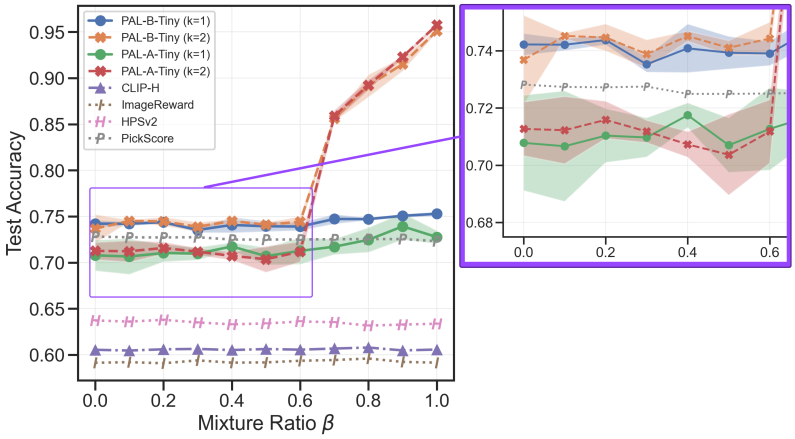


Figure 3. On Pick-a-Filter, PAL-B-Tiny outperforms homogeneous models as user groups become more heterogeneous (↑ β).

PickScore when trained on V1 with 165× fewer parameters (b) PAL exceeds SoTA performance on v2-no-leakage (i.e. fair comparison) by **2%** when training on v1, and by 2.5% if training on v2 (**C1, C3**). Training PAL from "scratch" i.e. with CLIP-H embeddings, outperforms training on PickScore embeddings (which were trained on v1). We note that PAL-B-Tiny (∼6M params) exceeds SoTA performance while training on a single RTX 4090 GPU (see Appendix E), whereas PickScore (∼1B params) is trained with 8×A100 GPUs – highlighting the suitability of PAL for efficient and democratic reward modeling (**C2**).

Remark. Since the data collection process for existing datasets involves the usage of strict rubrics [31, 62, 71], labeler performance monitoring [74], and a disproportionate amount of data provided by a small fraction of users, these datasets may not be heterogeneous. We note that a strict rubric leads to uniformity as it essentially crowdsources the criteria of the rubric instead of eliciting the preferences of the users. Therefore, even using PAL with $K = 1$, we can surpass existing SoTA performance. These results highlight the need for more nuanced approaches to collect datasets that elicit diverse opinions.

3.2. Pick-a-Filter (Text-to-Image)

As PickScore has been shown to lack significant heterogeneity [31], we artificially inject plurality to create the *Pick-a-Filter* dataset, and show that PAL significantly surpasses homogeneous reward models when pluralistic preferences are present (**C1, C2, C3**).

Dataset. Motivated by a natural human color preference distribution [45], we choose adding different color filters to the generated images as the mechanism by which we explicitly "inject" preference diversity into the v1 dataset (group 1 prefers 'red' tones and group 2 prefers 'blue' tones). To avoid the model latching on purely to color features when learning preferences, we use a hyperparameter called **mixture ratio** $\beta = N_f/N_o$, where N_f is the number of v1 pairs we choose to apply filters on, and N_o is the total number of original v1 pairs. The larger the β, the more color-filtered v1 pairs in the training set. We show and discuss our careful construction of Pick-a-Filter in Appendix D (Figure D.8).

Setup. We train PAL-B-Tiny with logistic loss on the *Pick-a-Filter* dataset with different mixture ratios. Detailed training setups are deferred to Appendix D.6.

Baselines. We compare PAL to CLIP-H, following [31], as well as strong T2I reward models: PickScore [31], ImageReward [74] and HPS v2 [71].

Results. Figure 3 shows that PAL-B-Tiny can learn diverse user preference groups across mixture ratios **(C1, C3)**. We can view β as indicating how much the two user groups prefer their respective color filters (higher $\beta \rightarrow$ affinity for filters). PAL significantly outperforms the homogeneous reward model in predicting user preferences – at β =1, PAL achieves 95.2% test accuracy compared to 75.4% from the homogeneous reward model **(C2)**.

4. Conclusions

We propose PAL, a novel framework for modeling personalizable rewards for pluralistic alignment (Section 2), which leverages shared structure across the population while learning to personalize in a sample-efficient way. We demonstrate that PAL is agnostic to modality, showing strong results on both image (Section 3.1, 3.2) and text (Appendix D.3) tasks. We also provide sample complexity bounds for generalization of the learned rewards to both seen and unseen users (Appendix C.1). Our work aids in building much-needed foundations toward plurality for the alignment of ML/AI models. Our experiments also highlight the limitations of many real human preference datasets that are collected with rubrics that make the dataset homogeneous, and call for a more nuanced approach to data collection in the future (Section 3.1). While the mixture modeling approach of PAL is flexible, a limitation of using it in a static setting is that it will not generalize to new users who fall outside the convex hull of learned prototypes (Appendix C.2). A more pragmatic and exciting

approach would be to continually learn prototypes to adapt to new users on the fly, which we leave for future work.

References

[1] 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. 9

[2] AI Anthropic. The claude 3 model family: Opus, sonnet, haiku. *Claude-3 Model Card*, 2024. 9

[3] Mohammad Gheshlaghi Azar, Zhaohan Daniel Guo, Bilal Piot, Remi Munos, Mark Rowland, Michal Valko, and Daniele Calandriello. A general theoretical paradigm to understand learning from human preferences. In *International Conference on Artificial Intelligence and Statistics*, pages 4447–4455. PMLR, 2024. 9

[4] Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022. 1, 9

[5] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022. 1, 9

[6] Michiel Bakker, Martin Chadwick, Hannah Sheahan, Michael Tessler, Lucy Campbell-Gillingham, Jan Balaguer, Nat McAleese, Amelia Glaese, John Aslanides, Matt Botvinick, et al. Fine-tuning language models to find agreement among humans with diverse preferences. *Advances in Neural Information Processing Systems*, 35:38176–38189, 2022. 1, 9

[7] Aurélien Bellet and Amaury Habrard. Robustness and generalization for metric learning. *Neurocomputing*, 151:259–267, 2015. 9

[8] Aurélien Bellet, Amaury Habrard, and Marc Sebban. Metric learning. *Synthesis lectures on artificial intelligence and machine learning*, 9(1):1–151, 2015. 9

[9] Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric learning*. Springer Nature, 2022. 9

[10] 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. 2, 9

[11] Mark Braverman and Elchanan Mossel. Noisy sorting without resampling. *arXiv preprint arXiv:0707.1051*, 2007. 9

[12] Gregory Canal, Blake Mason, Ramya Korlakai Vinayak, and Robert Nowak. One for all: Simultaneous metric and preference learning over multiple users. *arXiv preprint arXiv:2207.03609*, 2022. 9, 11, 12, 13

[13] Daiwei Chen, Yi Chen, Aniket Rege, Zhi Wang, and Ramya Korlakai Vinayak. PAL: Sample-efficient personalized reward modeling for pluralistic alignment. In *The Thirteenth International Conference on Learning Representations*, 2025. 12, 13

[14] Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. Bge m3-embedding: Multilingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation, 2024. 15

[15] Myra Cheng, Esin Durmus, and Dan Jurafsky. Marked personas: Using natural language prompts to measure stereotypes in language models. *arXiv preprint arXiv:2305.18189*, 2023. 1, 9

[16] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017. 1, 9

[17] Clyde H Coombs. Psychological scaling without a unit of measurement. *Psychological review*, 57(3):145, 1950. 2, 9

[18] Cody Ding. Evaluating change in behavioral preferences: Multidimensional scaling single-ideal point model. *Measurement and Evaluation in Counseling and Development*, 49(1):77–88, 2016. 9

[19] Gabriel Dulac-Arnold, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021. 9

[20] Esin Durmus, Karina Nguyen, Thomas I. Liao, Nicholas Schiefer, Amanda Askell, Anton Bakhtin, Carol Chen, Zac Hatfield-Dodds, Danny Hernandez, Nicholas Joseph, Liane Lovitt, Sam McCandlish, Orowa Sikder, Alex Tamkin, Janel Thamkul, Jared Kaplan, Jack Clark, and Deep Ganguli. Towards measuring the representation of subjective global opinions in language models, 2024. 1

[21] Brian Eriksson. Learning to top-k search using pairwise comparisons. In *Artificial Intelligence and Statistics*, pages 265–273. PMLR, 2013. 9

[22] Kawin Ethayarajh, Winnie Xu, Niklas Muennighoff, Dan Jurafsky, and Douwe Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024. 9

[23] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning and ranking by pairwise comparison. In *Preference learning*, pages 65–82. Springer, 2010. 9

[24] Deep Ganguli, Liane Lovitt, Jackson Kernion, Amanda Askell, Yuntao Bai, Saurav Kadavath, Ben Mann, Ethan Perez, Nicholas Schiefer, Kamal Ndousse, et al. Red teaming language models to reduce harms: Methods, scaling behaviors, and lessons learned. *arXiv preprint arXiv:2209.07858*, 2022. 1, 9

[25] Joel Huber. Ideal point models of preference. *ACR North American Advances*, 1976. 9

[26] David R Hunter. Mm algorithms for generalized bradley-terry models. *The annals of statistics*, 32(1):384–406, 2004. 9

[27] Robert Irvine, Douglas Boubert, Vyas Raina, Adian Liusie, Ziyi Zhu, Vineet Mudupalli, Aliaksei Korshuk, Zongyi Liu, Fritz Cremer, Valentin Assassi, et al. Rewarding chatbots for real-world engagement with millions of users. *arXiv preprint arXiv:2303.06135*, 2023. 9

[28] Kevin G Jamieson and Robert Nowak. Active ranking using pairwise comparisons. *Advances in neural information processing systems*, 24, 2011. 9

[29] Jiaming Ji, Tianyi Qiu, Boyuan Chen, Borong Zhang, Hantao Lou, Kaile Wang, Yawen Duan, Zhonghao He, Jiayi Zhou, Zhaowei Zhang, et al. Ai alignment: A comprehensive survey. *arXiv preprint arXiv:2310.19852*, 2023. 9

[30] Claire Kenyon-Mathieu and Warren Schudy. How to rank with few errors. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 95–103, 2007. 9

[31] Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2024. 4, 5, 9, 21

[32] Matthäus Kleindessner and Ulrike Luxburg. Uniqueness of ordinal embedding. In *Conference on Learning Theory*, pages 40–67. PMLR, 2014. 9

[33] Grgur Kovač, Masataka Sawayama, Rémy Portelas, Cédric Colas, Peter Ford Dominey, and Pierre-Yves Oudeyer. Large language models as superpositions of cultural perspectives. *arXiv preprint arXiv:2307.07870*, 2023. 1, 9

[34] Brian Kulis. Metric learning: A survey. *Foundations and Trends® in Machine Learning*, 5(4):287–364, 2013. 9

[35] Xinyu Li, Zachary C Lipton, and Liu Leqi. Personalized language modeling from personalized human feedback. *arXiv preprint arXiv:2402.05133*, 2024. 9, 15, 17, 18

[36] Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*, 2023. 15

[37] Zhixuan Liu, Zhanhui Zhou, Yuanfu Wang, Chao Yang, and Yu Qiao. Inference-time language model alignment via integrated value guidance. *arXiv preprint arXiv:2409.17819*, 2024. 1

[38] R Duncan Luce. *Individual choice behavior: A theoretical analysis*. New York: Wiley, 1959. 9

[39] Blake Mason, Lalit Jain, and Robert Nowak. Learning low-dimensional metrics. *Advances in neural information processing systems*, 30, 2017. 9

[40] Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81): 1–32, 2016. 13

[41] Sidharth Mudgal, Jong Lee, Harish Ganapathy, YaGuang Li, Tao Wang, Yanping Huang, Zhifeng Chen, Heng-Tze Cheng, Michael Collins, Trevor Strohman, et al. Controlled decoding from language models. In *Forty-first International Conference on Machine Learning*, 2024. 1

[42] Marcos Nadal and Anjan Chatterjee. Neuroaesthetics and art's diversity and universality. *Wiley Interdisciplinary Reviews: Cognitive Science*, 10(3):e1487, 2019. 1, 9

[43] Sahand Negahban, Sewoong Oh, and Devavrat Shah. Iterative ranking from pair-wise comparisons. *Advances in neural information processing systems*, 25, 2012. 9

[44] 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. 1, 9

[45] Stephen E Palmer and Karen B Schloss. Human preference for individual colors. In *Human Vision and Electronic Imaging XV*, pages 353–364. SPIE, 2010. 5

[46] Ethan Perez, Sam Ringer, Kamilė Lukošiūtė, Karina Nguyen, Edwin Chen, Scott Heiner, Craig Pettit, Catherine Olsson, Sandipan Kundu, Saurav Kadavath, Andy Jones, Anna Chen, Ben Mann, Brian Israel, Bryan Seethor, Cameron McKinnon, Christopher Olah, Da Yan, Daniela Amodei, Dario Amodei, Dawn Drain, Dustin Li, Eli Tran-Johnson, Guro Khundadze, Jackson Kernion, James Landis, Jamie Kerr, Jared Mueller, Jeeyoon Hyun, Joshua Landau, Kamal Ndousse, Landon Goldberg, Liane Lovitt, Martin Lucas, Michael Sellitto, Miranda Zhang, Neerav Kingsland, Nelson Elhage, Nicholas Joseph, Noemí Mercado, Nova DasSarma, Oliver Rausch, Robin Larson, Sam McCandlish, Scott Johnston, Shauna Kravec, Sheer El Showk, Tamera Lanham, Timothy Telleen-Lawton, Tom Brown, Tom Henighan, Tristan Hume, Yuntao Bai, Zac Hatfield-Dodds, Jack Clark, Samuel R. Bowman, Amanda Askell, Roger Grosse, Danny Hernandez, Deep Ganguli, Evan Hubinger, Nicholas Schiefer, and Jared Kaplan. Discovering language model behaviors with model-written evaluations, 2022. 14, 18

[47] 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. 9, 15, 18

[48] Arun Rajkumar and Shivani Agarwal. A statistical convergence perspective of algorithms for rank aggregation from pairwise data. In *International conference on machine learning*, pages 118–126. PMLR, 2014. 9

[49] Alexandre Rame, Guillaume Couairon, Corentin Dancette, Jean-Baptiste Gaya, Mustafa Shukor, Laure Soulier, and Matthieu Cord. Rewarded soups: towards pareto-optimal alignment by interpolating weights fine-tuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36, 2024. 1, 9

[50] Aniket Rege, Aditya Kusupati, Alan Fan, Qingqing Cao, Sham Kakade, Prateek Jain, Ali Farhadi, et al. Adanns: A framework for adaptive semantic search. *Advances in Neural Information Processing Systems*, 36:76311–76335, 2023. 1

[51] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 9

[52] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019. 15

[53] Shibani Santurkar, Esin Durmus, Faisal Ladhak, Cinoo Lee, Percy Liang, and Tatsunori Hashimoto. Whose opinions do

language models reflect? In *Proceedings of the 40th International Conference on Machine Learning*, pages 29971–30004. PMLR, 2023. 1, 9

[54] Matthew Schultz and Thorsten Joachims. Learning a distance metric from relative comparisons. *Advances in neural information processing systems*, 16, 2003. 9

[55] Nihar Shah, Sivaraman Balakrishnan, Aditya Guntuboyina, and Martin Wainwright. Stochastically transitive models for pairwise comparisons: Statistical and computational issues. In *International Conference on Machine Learning*, pages 11–20. PMLR, 2016. 9

[56] Nihar B Shah and Martin J Wainwright. Simple, robust and optimal ranking from pairwise comparisons. *The Journal of Machine Learning Research*, 18(1):7246–7283, 2017. 9

[57] Roger N Shepard. The analysis of proximities: multidimensional scaling with an unknown distance function. i. *Psychometrika*, 27(2):125–140, 1962. 9

[58] Roger N Shepard. The analysis of proximities: Multidimensional scaling with an unknown distance function. ii. *Psychometrika*, 27(3):219–246, 1962.

[59] Roger N Shepard. Metric structures in ordinal data. *Journal of Mathematical Psychology*, 3(2):287–315, 1966. 9

[60] Adish Singla, Sebastian Tschiatschek, and Andreas Krause. Actively learning hemimetrics with applications to eliciting user preferences. In *International Conference on Machine Learning*, pages 412–420. PMLR, 2016. 9

[61] Taylor Sorensen, Jared Moore, Jillian Fisher, Mitchell Gordon, Niloofar Mireshghallah, Christopher Michael Rytting, Andre Ye, Liwei Jiang, Ximing Lu, Nouha Dziri, et al. A roadmap to pluralistic alignment. *arXiv preprint arXiv:2402.05070*, 2024. 1, 9

[62] Nisan Stiennon, Long Ouyang, Jeffrey Wu, Daniel Ziegler, Ryan Lowe, Chelsea Voss, Alec Radford, Dario Amodei, and Paul F Christiano. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33:3008–3021, 2020. 5, 9, 15

[63] Omer Tamuz, Ce Liu, Serge Belongie, Ohad Shamir, and Adam Tauman Kalai. Adaptively learning the crowd kernel. *arXiv preprint arXiv:1105.1033*, 2011. 9

[64] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023. 9

[65] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of lm alignment. *arXiv preprint arXiv:2310.16944*, 2023. 9

[66] Binghai Wang, Rui Zheng, Lu Chen, Yan Liu, Shihan Dou, Caishuang Huang, Wei Shen, Senjie Jin, Enyu Zhou, Chenyu Shi, et al. Secrets of rlhf in large language models part ii: Reward modeling. *arXiv preprint arXiv:2401.06080*, 2024. 9

[67] Haoxiang Wang, Yong Lin, Wei Xiong, Rui Yang, Shizhe Diao, Shuang Qiu, Han Zhao, and Tong Zhang. Arithmetic control of llms for diverse user preferences: Directional preference alignment with multi-objective rewards. *arXiv preprint arXiv:2402.18571*, 2024. 9

[68] Zhi Wang, Geelon So, and Ramya Korlakai Vinayak. Metric learning from limited pairwise preference comparisons. In *UAI*, 2024. 9

[69] Aaron Wildavsky. Choosing preferences by constructing institutions: A cultural theory of preference formation. *American political science review*, 81(1):3–21, 1987. 1, 9

[70] Tianhao Wu, Banghua Zhu, Ruoyu Zhang, Zhaojin Wen, Kannan Ramchandran, and Jiantao Jiao. Pairwise proximal policy optimization: Harnessing relative feedback for llm alignment. *arXiv preprint arXiv:2310.00212*, 2023. 1, 9

[71] Xiaoshi Wu, Yiming Hao, Keqiang Sun, Yixiong Chen, Feng Zhu, Rui Zhao, and Hongsheng Li. Human preference score v2: A solid benchmark for evaluating human preferences of text-to-image synthesis. *arXiv preprint arXiv:2306.09341*, 2023. 4, 5, 9

[72] Zeqiu Wu, Yushi Hu, Weijia Shi, Nouha Dziri, Alane Suhr, Prithviraj Ammanabrolu, Noah A Smith, Mari Ostendorf, and Hannaneh Hajishirzi. Fine-grained human feedback gives better rewards for language model training. *Advances in Neural Information Processing Systems*, 36, 2024. 9

[73] Austin Xu and Mark Davenport. Simultaneous preference and metric learning from paired comparisons. *Advances in Neural Information Processing Systems*, 33:454–465, 2020. 9, 11

[74] Jiazheng Xu, Xiao Liu, Yuchen Wu, Yuxuan Tong, Qinkai Li, Ming Ding, Jie Tang, and Yuxiao Dong. Imagereward: Learning and evaluating human preferences for text-to-image generation. *Advances in Neural Information Processing Systems*, 36, 2024. 4, 5, 9

[75] Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. Opt: Open pre-trained transformer language models, 2022. 15

A. Extended Related Work

Alignment Status Quo. Popular existing foundation models [1, 2, 44, 64] typically use RLHF [16, 22, 62, 70] to align models after pretraining. Recent foundation models such as Zephyr [65] and the Archangel suite[4] have shifted to directly optimizing on human preferences [3, 22, 47] to avoid the nuances of RL optimization [19]. There has also been significant recent work in collecting large human preference datasets for reward model training in the text-to-image space [31, 71, 74] (typically diffusion models [51]).

Reward Modeling. These existing alignment frameworks generally assume that all humans share a single unified preference (e.g. LLM "helpfulness" or "harmlessness" [4]) and ascribe to the Bradley-Terry [10] model of pairwise preferences. Consensus-based methods [6] aim to find agreement among labelers for specific goals like harmlessness [5, 24], helpfulness [4], or engagement [27]. By design, these methods inherently prioritize the universal preference (and biases) induced by the labelers [15, 33, 53]. In reality, humans have diverse, heterogeneous preferences [42, 61, 69] that depend on individual contexts, and may even share a group structure [6]. Rewarded soups [49] make a case to capture diversity through post-hoc weight-space interpolation over a mixture of experts that learn diverse rewards. However, these rewards are learned by predefining what aspects are important which is done by the system designer. Separate datasets are collected to elicit human preferences on these axes as to how much people care of them. DPA [67] models rewards as directions instead of scalars, and trains a multi-objective reward model for RLHF. Wu et al. [72] propose fine-grained multi-objective rewards to provide more focused signal for RLHF. Recently, Li et al. [35] propose personalized reward modeling by learning a general user embedding and treating each individual as a perturbation to the embedding. As this preference formulation is still homogeneous for users not in the dataset, they use the fixed general user embedding for generalizing to unseen users, i.e., do not personalize to new users.

Recent survey works provide excellent summaries of literature for alignment [29] and reward modeling [66].

Human Preference Datasets. The preference universality assumption also extends into the data annotation/labeling processing, where labelers are given a rubric to select preferences (e.g. to rank an image pair considering image aesthetics and image-prompt alignment [31]). Due to this rubric, the current largest scale text-to-image generation preference datasets [31, 71, 74] show limited diversity among labelers. In the Pick-a-Pic train set [31], there are only 701 disagreements among the 12487 image pairs labeled by different users (94.38% agreement), and there are zero disagreements in validation (1261 pairs) and test (1453 pairs) sets. HPS [71] found that labeler agreement over diffusion model generations was higher for models of similar quality or size, though this diversity comes with the caveat of the labelers being provided a rubric to provide their preferences. Imagereward [74] use researcher agreement as a *criteria* to hire labelers. In the LLM domain, the popular Summarize from Feedback dataset [62] is also collected with rigid rubric, with labeler performance measured via agreement to the preferred answer of the authors. During the data collection period, only labelers with satisfactory agreement were retained, which led to a small number of users, all in agreement with the authors' rubric, being responsible for a majority of labeled comparisons. Status quo preference datasets used to align foundation models thus suffer from a lack of diversity due to the nature of their data collection.

Preference learning. There is rich literature on preference learning and ranking in various domains ranging from psychology, marketing, recommendation systems, quantifying social science surveys to crowdsourced democracy, voting theory and social choice theory. We provide a few relevant works here and direct reader to surveys such as [23]. Ranking based models, e.g., BTL-model [10, 38], stochastic transitivity models [55] focus on finding ranking of m items or finding top-k items by pairwise comparisons [11, 21, 26, 30, 43, 48, 56]. Ranking m items in these settings requires $\mathcal{O}(m \log m)$ queries. There is also rich literature that stems from ideal point model [12, 17, 18, 25, 28, 60, 73]. Under the ideal point based models, the query complexity for ranking m items reduces to $\mathcal{O}(d \log m)$, where d is the dimension of the domain of representations which is usually much smaller than the number of items being ranked [28]. This is due to the fact that once the preference point is learned, it can then be used to predict rankings of new items without needing more comparisons.

Metric learning has been studied quite extensively and we direct the reader to surveys [34] and books [9]. In particular, metric learning based on triplet querying has also been quite extesively studied [7, 8, 32, 34, 39, 54, 57–59, 63] which aims to learn the underlying unknown metric under the assumption that the people base their judgement for a triple query with concepts $\mathbf{x}_a, \mathbf{x}_b, \mathbf{x}_c \in \mathcal{D}$ on the relative similarities based on the distances between these concepts under the unknown metric.

Simultaneous metric and preference learning. More recently a few works have considered the problem of unknown metric in preference learning and proposed methods [12, 68, 73] and provided sample complexity analysis [12, 68] for simultaneously learning an unknown Mahalanobis metric and unknown user preference(s). Learning the unknown Mahalanobis metric can be viewed as learning linear layer on top of the embeddings from a foundation

[4] https://github.com/ContextualAI/HALOs

model. From our reframing of alignment, these works can be looked as PAL-A with linear function for f and individual user preferences instead of having any structure over them.

B. Detailed Model Overview

PAL-A: Diverse preferences modeled via latent ideal preference points. The shared sense of similarity of different alternates being compared is modeled as Euclidean distance in an *unknown* mapped space, captured by $f : \mathbb{R}^{2D} \to \mathbb{R}^d$ that jointly maps the output and context, $(\mathbf{x}; \mathbf{x}_c)$, to this unknown space. The *latent* preference of each user i is modeled using an *unknown* ideal preference point $\mathbf{a}^{(i)} \in \mathbb{R}^{2D}$. How much the user i values output $\mathbf{x}$ for given context $\mathbf{x}_c$ is modeled as inversely proportional how far away the mapping of $(\mathbf{x}; \mathbf{x}_c)$ is from the user's ideal point. To further capture the commonality in the preferences among users, each user's preference points is modeled as a convex combination of K prototypical points, that is, $\mathbf{a}^{(i)} := \sum_{k=1}^{K} w_k^{(i)} \mathbf{p}_k$ where the weights $w_k^{(i)} \geq 0$ and $\sum_{k=1}^{K} w_k^{(i)} = 1$, and $\{\mathbf{p}_1, ..., \mathbf{p}_K\}$ with $\mathbf{p}_i \in \mathbb{R}^{2D}$ are K prototypical ideal preference points. More formally, the personalized reward function and the corresponding personalized probabilistic preference model is given by,

$$\textbf{PAL-A:} \quad r_\theta^{(i)}(\mathbf{x}; \mathbf{x}_c) := ||f(\mathbf{x}; \mathbf{x}_c) - f(\mathbf{a}^{(i)})||_2^2, \tag{6}$$

$$\begin{aligned} &\Pr(\mathbf{x}_l \succ \mathbf{x}_r | \mathbf{x}_c, i) \\ &= h(||f(\mathbf{x}_r; \mathbf{x}_c) - f(\mathbf{a}^{(i)})||_2^2 - ||f(\mathbf{x}_l; \mathbf{x}_c) - f(\mathbf{a}^{(i)})||_2^2), \end{aligned} \tag{7}$$

Denoting $\mathbf{P} := [\mathbf{p}_1, \cdots, \mathbf{p}_K]$, each user's preference point $\mathbf{a}^{(i)} := \mathbf{P}\mathbf{w}^{(i)}$, where $\mathbf{w}^{(i)} := [w_1^{(i)}, \cdots, w_K^{(i)}]^\top$, lies in the $(K-1)$-dimensional simplex, Δ^{K-1}.

B.1. Illustrations and Pseudocode for the PAL Framework

Figure B.1 illustrates the ideal point model considered in PAL-A. We show the modeling mechanism of PAL (Section 2) in slightly more detail in Figure B.2. Algorithm 1 and 2 provide the pseudocode for the learning algorithms of PAL-A and PAL-B.

B.2. Modeling Choices: When to Use PAL-A or PAL-B in Practice?

We note that PAL-B is more natural in generative modeling settings, as it learns a personalized mapping $z^{(i)}(\mathbf{x}_c)$ for each user i and any given prompt $\mathbf{x}_c$, and it learns a separate mapping for outputs $\mathbf{x}$. In contrast, PAL-A learns an ideal point $\mathbf{a}^{(i)}$ for each user i fixed across all prompts and it learns to jointly map the prompt $\mathbf{x}_c$ and output $\mathbf{x}$ in the same space.

From experiments, we found that PAL-B serves as a reliable default choice (see Figure 2, Figure D.3 and Table 2 on Reddit TL;DR Summary; Table 1 on Pick-a-Pic; and Figure 3 on Pick-a-Filter). We are able to get PAL-A to work competitively to PAL-B in most settings (see Table D.1, Table D.3 and Section D.4 in the Appendix); however, in practice, this may require additional engineering effort to optimize effectively.

C. Sample complexity for learning rewards under PAL modeling

In this section, we shed light on the per-user sample complexity of PAL for (1) generalization to unseen pairs involving known users, and (2) few-shot generalization to unseen users. We present theoretical guarantees in Section C.1, and empirical results from numerical simulations in Section C.2.

C.1. Theoretical Guarantees

To shed light on the benefits of using mixture modeling approach, we provide theoretical analysis of PAL-A with fully-connected linear layer in neural network mapping f. For simplicity of exposition, we subsume the context/prompt vectors into the item embeddings, treating $(\mathbf{x}; \mathbf{x}_c)$ as $\mathbf{x} \in \mathbb{R}^D$. Consider a class linear transformations, $\mathcal{F} \subset \left\{ f : \mathbb{R}^D \to \mathbb{R}^D \;\middle|\; f(\mathbf{x}) = \mathbf{A}\mathbf{x}, \mathbf{A} \in \mathbb{R}^{D \times D} \right\}$. Then, the difference of scores between two items for a user with ideal point $\mathbf{a} = \mathbf{P}\mathbf{w}$ is given by

$$\begin{aligned} &\|f(\mathbf{x}_l) - f(\mathbf{a})\|_2^2 - \|f(\mathbf{x}_r) - f(\mathbf{a})\|_2^2 \\ &= (\mathbf{x}_l - \mathbf{a})^\top \mathbf{A}^\top \mathbf{A} (\mathbf{x}_l - \mathbf{a}) - (\mathbf{x}_r - \mathbf{a})^\top \mathbf{A}^\top \mathbf{A} (\mathbf{x}_r - \mathbf{a}) \\ &= \mathbf{x}_l^\top (\mathbf{A}^\top \mathbf{A}) \mathbf{x}_l - \mathbf{x}_r^\top (\mathbf{A}^\top \mathbf{A}) \mathbf{x}_r - 2(\mathbf{x}_l - \mathbf{x}_r)^\top (\mathbf{A}^\top \mathbf{A}) \mathbf{P}\mathbf{w}. \end{aligned}$$

Algorithm 1 Learning algorithm for PAL-A

Require: Dataset $\mathcal{D} = \bigcup_{i=1}^{N} \left\{ \left(\mathbf{x}_{j,l}^{(i)}, \mathbf{x}_{j,r}^{(i)}; \mathbf{x}_{j,c}^{(i)}, y_j^{(i)} \right) \right\}_{j=1}^{m_i}$, loss function ℓ, model class for f_θ, prototypes $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_K]$ where $\mathbf{p}_k \in \mathbb{R}^D$, user weights $\mathbf{W} = [\mathbf{w}^{(1)}, ..., \mathbf{w}^{(N)}]$ where $\mathbf{w}^{(i)} \in \Delta^{K-1}$.

for each iteration **do**
  Sample a mini-batch $\left\{ \left(\mathbf{x}_{j,l}^{(i)}, \mathbf{x}_{j,r}^{(i)}; \mathbf{x}_{j,c}^{(i)}, y_j^{(i)} \right) \right\}$ ▷ random pairs, not ordered by users
  User Ideal Points: $\mathbf{a}^{(i)} = \mathbf{P} \cdot \mathbf{w}^{(i)}$
  Distances:
  $d_{l,j}^{(i)} = ||f_\theta \left(\mathbf{x}_{l,j}^{(i)}; \mathbf{x}_{c,j}^{(i)} \right) - f_\theta(\mathbf{a}^{(i)})||_2^2$, $d_{r,j}^{(i)} = ||f_\theta \left(\mathbf{x}_{r,j}^{(i)}; \mathbf{x}_{c,j}^{(i)} \right) - f_\theta(\mathbf{a}^{(i)})||_2^2$
  Loss: $\psi_j^{(i)}(\mathbf{x}_{l,j}^{(i)}, \mathbf{x}_{r,j}^{(i)}; \mathbf{x}_{c,j}^{(i)}, y_j^{(i)}) = \ell(y_j^{(i)} \cdot (d_{l,j}^{(i)} - d_{r,j}^{(i)}))$
  Update Step:
  $\operatorname{argmin}_{\theta, \mathbf{P}, \{\mathbf{w}^{(i)}\}_{i=1}^N} \sum_{i,j} \psi_j^{(i)}(\mathbf{x}_{l,j}^{(i)}, \mathbf{x}_{r,j}^{(i)}; \mathbf{x}_{c,j}^{(i)}, y_j^{(i)})$
end for

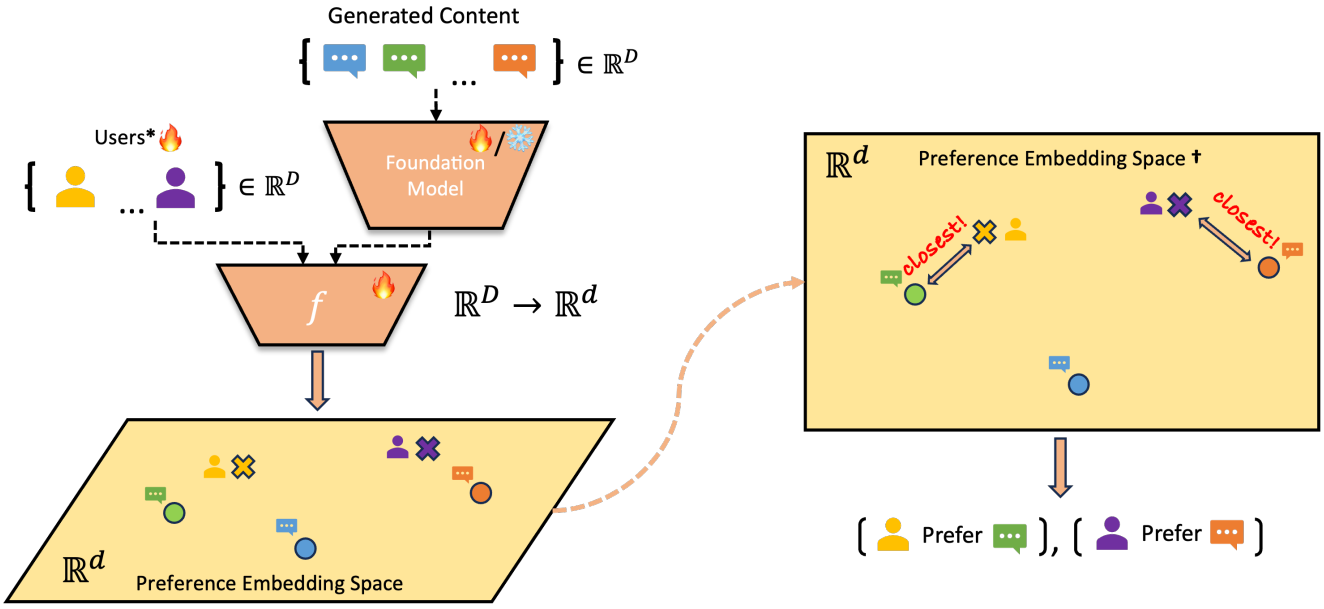


Figure B.1. In the ideal point model considered in PAL-A, items and users (represented by their ideal points) are mapped to $\mathbb{R}^d$ via an unknown function f. A user's preference for an item (e.g., an image or a text summary) is inversely related to the distance between the item and the user's ideal point. For example, the yellow user prefers the green text summary, and the purple user prefers the orange text summary, as these items are "closer" to their respective ideal points.

Algorithm 2 Learning algorithm for PAL-B

Require: Dataset $\mathcal{D} = \bigcup_{i=1}^{N} \left\{ \left(\mathbf{x}_{j,l}^{(i)}, \mathbf{x}_{j,r}^{(i)}; \mathbf{x}_{j,c}^{(i)}, y_j^{(i)} \right) \right\}_{j=1}^{m_i}$, loss function ℓ, mapping function f_θ, prototype mapping functions $\{g_{\theta_k}\}_{k=1}^{K}$, user weights $\{\mathbf{w}^{(i)} := [w_1^{(i)}, ..., w_K^{(i)}]\}_{i=1}^{N}$.

for each iteration **do**
  Sample a mini-batch $\left\{ \left(\mathbf{x}_{j,l}^{(i)}, \mathbf{x}_{j,r}^{(i)}; \mathbf{x}_{j,c}^{(i)}, y_j^{(i)} \right) \right\}$ ▷ random pairs, not ordered by users
  User Ideal Point (conditioned on prompts):
  $\mathbf{a}^{(i)} = \left[g_{\theta_i}(\mathbf{x}_{c,j}^{(i)}), ..., g_{\theta_K}(\mathbf{x}_{c,j}^{(i)}) \right]^\top \cdot \mathbf{w}^{(i)}$
  Distance:
  $d_{l,j}^{(i)} = \langle f_\theta \left(\mathbf{x}_{l,j}^{(i)} \right), \mathbf{a}^{(i)} \rangle$, $d_{r,j}^{(i)} = \langle f_\theta \left(\mathbf{x}_{r,j}^{(i)} \right), \mathbf{a}^{(i)} \rangle$
  Loss: $\psi_j^{(i)}(\mathbf{x}_{l,j}^{(i)}, \mathbf{x}_{r,j}^{(i)}; \mathbf{x}_{c,j}^{(i)}, y_j^{(i)}) = \ell(y_j^{(i)} \cdot (d_{l,j}^{(i)} - d_{r,j}^{(i)}))$
  Update Step:
  $\operatorname{argmin}_{\theta, \mathbf{P}, \{\mathbf{w}^{(i)}\}_{i=1}^{N}} \sum \psi_j^{(i)}(\mathbf{x}_{l,j}^{(i)}, \mathbf{x}_{r,j}^{(i)}; \mathbf{x}_{c,j}^{(i)}, y_j^{(i)})$
end for

Observe that the right-hand side of the first equality represents the difference between the squared Mahalanobis distances[5] from $\mathbf{x}_l$ to $\mathbf{a}$ and from $\mathbf{x}_r$ to $\mathbf{a}$, where the Mahalanobis distance is defined by $\mathbf{A}^\top \mathbf{A}$. In other words, the problem is equivalent to simultaneously learning a Mahalanobis distance and the user ideal points [12, 73].

Now, let $\mathbf{M} := \mathbf{A}^\top \mathbf{A}$ and $\mathbf{Q} := -2\mathbf{M}\mathbf{P}$. The difference in scores can then be written as:

$$\begin{aligned} &\|f(\mathbf{x}_l) - f(\mathbf{a})\|_2^2 - \|f(\mathbf{x}_r) - f(\mathbf{a})\|_2^2 \\ &\quad = \mathbf{x}_l^\top \mathbf{M} \mathbf{x}_l - \mathbf{x}_r^\top \mathbf{M} \mathbf{x}_r + (\mathbf{x}_l - \mathbf{x}_r)^\top \mathbf{Q} \mathbf{w}. \end{aligned}$$

Given these reparameterizations from f to $\mathbf{M}$ and from $\mathbf{P}$ to $\mathbf{Q}$, for the remainder of this section, we assume that the reward modeling problem is defined over $\mathbf{M} \in \left\{ \mathbf{M} \in \mathbb{R}^{D \times D} : \|\mathbf{M}\|_{\mathrm{F}} \leq \zeta_M, \mathbf{M} \succeq 0 \right\}$, $\mathbf{Q} \in \left\{ \mathbf{Q} \in \mathbb{R}^{D \times K} : \|\mathbf{Q}_k\|_2 \leq \zeta_v \ \forall k \in [K] \right\}$, and $\mathbf{w}^{(i)} \in \Delta^{K-1}$ for each user i. In addition, we let $\ell : \mathbb{R} \to [0, 1]$ be an L-Lipschitz loss function.

Work of Canal et al. [12, Theorem 3.1] shows that a set of $N \geq \Omega(D^2)$ known users, the per-user sample complexity for generalization to *unseen pairs* of items is $\tilde{\mathcal{O}}(D)$. Key questions remain: Does our mixture modeling lead to improved per-user sample complexity, e.g., $\tilde{\mathcal{O}}(K)$? Can we

[5] The Mahalanobis distance defined by a symmetric, positive semidefinite matrix $\mathbf{M}$ is given by $d_{\mathbf{M}}(\mathbf{x}, \mathbf{y}) := \sqrt{(\mathbf{x} - \mathbf{y})^\top \mathbf{M} (\mathbf{x} - \mathbf{y})}$.

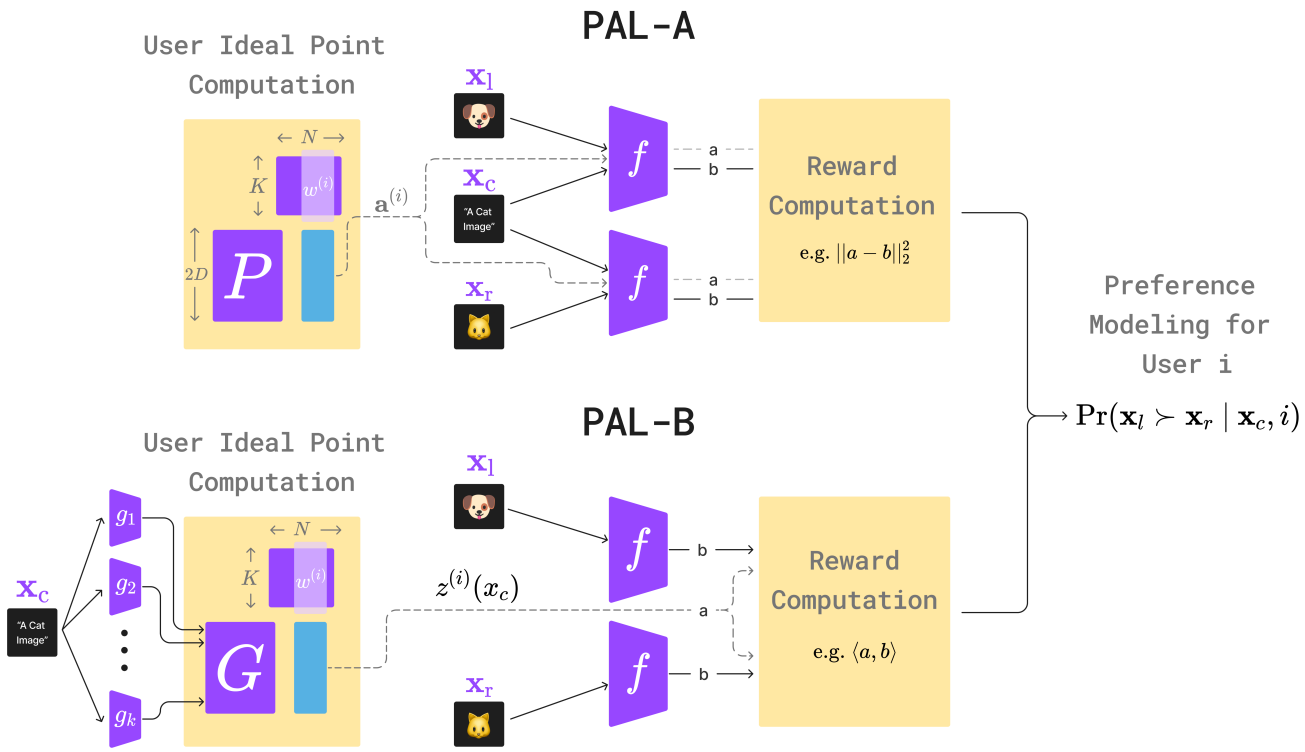


Figure B.2. Illustration of PAL framework for learning from diverse preferences (Section 2). For any user i, the probability of preferring $\mathbf{x}_l$ to $\mathbf{x}_r$ for the context $\mathbf{x}_c$ is computed by a reward model $r_\theta^{(i)}$ which uses a mixture modeling approach to assign a scalar reward to a sample (e.g. $\mathbf{x}_l$ or $\mathbf{x}_r$) given context ($\mathbf{x}_c$). In PAL-A, each user i's preference $a^{(i)}$ is modeled as a convex combination of K prototypical preferences, i.e. $a^{(i)} = Pw^{(i)}$. In PAL-B, each user i's preference $z^{(i)}(\mathbf{x}_c)$ is modeled as a convex combination of K prototypical functions $g_1 \cdots g_K$, i.e. $z^{(i)}(\mathbf{x}_c) =$. Reward functions formulated using the PAL framework can be used flexibly, e.g., with fixed preference points (PAL-A), with preference points that are functions of the context/prompt $\mathbf{x}_c$ (PAL-B).

characterize PAL's generalization ability to *unseen users*, and determine the per-user sample complexity for few-shot localization of preference points (going beyond analysis in [12])?

Generalization for seen users and unseen pairs. Without loss of generality, let us assume that each user answers m preference comparisons. Let $S_i = \left\{\left(\mathbf{x}_{j,l}^{(i)}, \mathbf{x}_{j,r}^{(i)}, y_j^{(i)}\right)\right\}_{j=1}^m$ denote an i.i.d. sample of size m from user i, and let $\mathcal{S} = \bigcup_{i=1}^N S_i$ represent the dataset from N users. Given $\mathcal{S}$, for any $(\mathbf{M}, \mathbf{Q}, \{\mathbf{w}^{(i)}\}_{i=1}^N)$, we define its empirical risk $\widehat{R}_{\mathcal{S}}(\mathbf{M}, \mathbf{Q}, \{\mathbf{w}^{(i)}\}_{i=1}^N)$ as

$$\frac{1}{Nm} \sum_{(i,j)} \ell\Big(y_j^{(i)} \Big(\mathbf{x}_{j,l}^{(i)\top} \mathbf{M} \mathbf{x}_{j,l}^{(i)} - \mathbf{x}_{j,r}^{(i)\top} \mathbf{M} \mathbf{x}_{j,r}^{(i)} + (\mathbf{x}_{j,l}^{(i)} - \mathbf{x}_{j,r}^{(i)})^\top \mathbf{Q} \mathbf{w}^{(i)}\Big)\Big). \tag{8}$$

Let $(\widehat{\mathbf{M}}, \widehat{\mathbf{Q}}, \{\widehat{\mathbf{w}}^{(i)}\}_{i=1}^N)$ minimize the empirical risk given $\mathcal{D}$, and $(\mathbf{M}^*, \mathbf{Q}^*, \{\mathbf{w}^{*(i)}\}_{i=1}^N)$ minimize the true risk, $\mathbb{E}_{\mathcal{S}}[\widehat{R}_{\mathcal{S}}(\mathbf{M}, \mathbf{Q}, \{\mathbf{w}^{(i)}\}_{i=1}^N)]$. Theorem C.1 below provides an upper bound on the excess risk. See Chen et al. [13] (Appendix C) for detailed proofs.

Theorem C.1. ***(Seen user generalization)*** *Suppose $K < \min(N, D)$, and for each comparison, the user is asked to compare two items drawn i.i.d. from the uniform distribution on the unit sphere,* $\mathrm{Unif}(\mathbb{S}^{D-1})$. *Then, with probability at least $1-\delta$,*

$$\mathbb{E}_{\mathcal{S}}[\widehat{R}_{\mathcal{S}}(\widehat{\mathbf{M}}, \widehat{\mathbf{Q}}, \{\widehat{\mathbf{w}}^{(i)}\}_{i=1}^N)] - \mathbb{E}_{\mathcal{S}}[\widehat{R}_{\mathcal{S}}(\mathbf{M}^*, \mathbf{Q}^*, \{\mathbf{w}^{*(i)}\}_{i=1}^N)]$$
$$\leq 12L \sqrt{\frac{\zeta_M^2 + \left(\frac{KN}{D} + K\right)\zeta_v^2}{Nm}} \log(N+D) + \sqrt{\frac{2\log\frac{2}{\delta}}{Nm}}.$$

To interpret this result in a practical setting, let us further assume that the entries of $\mathbf{M}^*$ and $\mathbf{Q}^*$ are bounded by some constant, and set $\zeta_M = D$ and $\zeta_v = \sqrt{D}$, as done in [12]. Then, the above bound becomes $\tilde{\mathcal{O}}\left(\sqrt{\frac{D^2+KD+KN}{Nm}}\right)$,

where $\tilde{\mathcal{O}}$ hides the parameters δ and L and ignores logarithmic terms. Observe first that the bound decays as either the number of users N or the number of samples per user m increases. In addition, when $N \geq \Omega(D^2)$, the bound simplifies to $\tilde{\mathcal{O}}(\sqrt{K/m})$, which implies a per-user sample complexity of $\tilde{\mathcal{O}}(K)$. This contrasts with the existing result of $\tilde{\mathcal{O}}(D)$ without mixture modeling [12, Theorem 3.1]. The result captures the intuition that if the users amortize the cost of learning the common $\mathbf{M}$ and $\mathbf{Q}$, then each user only needs to individually learn their weights $w^{(i)} \in \Delta^{K-1}$.

Generalization for unseen users. So far, we have discussed the generalization ability of PAL to unseen pairs among known users, but how well does our model generalize to new users, whose data was not included in the training set at all? In this work, we provide the first answer to this question in the context of preference alignment under the ideal point model, leveraging the framework and tools for multi-task learning and learning-to-learn [40].

Suppose there is a distribution η over a set of users $\mathcal{U}$. Given any representation $\mathbf{M}$ and prototypes $\mathbf{Q}$, along with an i.i.d. sample $S = \{(\mathbf{x}_{j,l}, \mathbf{x}_{j,r}, y_j)\}_{j=1}^m$ of m comparisons from some new user $u \in \mathcal{U}$, the natural approach is to few-shot learn the weights that minimize the empirical loss:

$$\tilde{\mathbf{w}}_{S;\mathbf{M},\mathbf{Q}} := \operatorname*{argmin}_{\mathbf{w}\in\Delta^{K-1}} \frac{1}{m}\sum_{j=1}^{m} \ell\Big(y_j\Big(\mathbf{x}_{j,l}^\top \mathbf{M}\mathbf{x}_{j,l} - \mathbf{x}_{j,r}^\top \mathbf{M}\mathbf{x}_{j,r} + (\mathbf{x}_{j,l}-\mathbf{x}_{j,r})^\top \mathbf{Q}\mathbf{w}\Big)\Big);$$

and one can evaluate the expected performance of $(\mathbf{M}, \mathbf{Q}, \tilde{\mathbf{w}}_{S;\mathbf{M},\mathbf{Q}})$ for user u:

$$\tilde{L}(\mathbf{M},\mathbf{Q};u,S) := \mathbb{E}_{(\mathbf{x}_l,\mathbf{x}_r,y)}\Big[\ell\Big(y\Big(\mathbf{x}_l^\top \mathbf{M}\mathbf{x}_l - \mathbf{x}_r^\top \mathbf{M}\mathbf{x}_r + (\mathbf{x}_l-\mathbf{x}_r)^\top \mathbf{Q}\tilde{\mathbf{w}}_{S;\mathbf{M},\mathbf{Q}}\Big)\Big)\Big].$$

The unseen-user risk of $(\mathbf{M}, \mathbf{Q})$ is defined as: $L^{\text{unseen}}_{\text{user}}(\mathbf{M},\mathbf{Q}) := \mathbb{E}_u[L(\mathbf{M},\mathbf{Q};u)]$, where $L(\mathbf{M},\mathbf{Q};u) := \mathbb{E}_S\left[\tilde{L}(\mathbf{M},\mathbf{Q};u,S)\right]$.

Suppose that for training, N users are drawn according to η, and each answers m comparison queries, resulting in the dataset, $S_1 \cup \ldots \cup S_N$. Let $(\widehat{\mathbf{M}}, \widehat{\mathbf{Q}})$ be components of the minimizer of the empirical risk given by Eq. (8). Theorem C.2 provides an upper bound on its excess risk when compared with

$$(\mathbf{M}^*,\mathbf{Q}^*) := \operatorname*{argmin}_{\mathbf{M},\mathbf{Q}} \mathbb{E}_u\Big[\min_{\mathbf{w}\in\Delta^{K-1}} \mathbb{E}_{(\mathbf{x}_l,\mathbf{x}_r,y)}\Big[\ell\Big(y\Big(\mathbf{x}_l^\top \mathbf{M}\mathbf{x}_l - \mathbf{x}_r^\top \mathbf{M}\mathbf{x}_r + (\mathbf{x}_l-\mathbf{x}_r)^\top \mathbf{Q}\mathbf{w}\Big)\Big)\Big]\Big],$$

which assumes oracle knowledge of each user's preferences. See Chen et al. [13] (Appendix C) for a detailed proof.

Theorem C.2. ***(Unseen user generalization)*** *Suppose $K < \min(N, d)$, and for each comparison, the user is asked to compare two items that are drawn i.i.d. from the uniform distribution on the unit sphere,* $\mathrm{Unif}(\mathbb{S}^{D-1})$. *Then, with probability at least $1-\delta$ over $S_1, \ldots, S_N$,*

$$L^{\text{unseen}}_{\text{user}}(\widehat{\mathbf{M}},\widehat{\mathbf{Q}}) - L^{\text{unseen}}_{\text{user}}(\mathbf{M}^*,\mathbf{Q}^*) \leq 18L\sqrt{\frac{\zeta_M^2 + K^2\zeta_v^2}{N}} + 3L\sqrt{\frac{K\zeta_v^2}{Dm}} + \sqrt{\frac{8\log\frac{4}{\delta}}{N}}.$$

Again, let us set $\zeta_M = D$ and $\zeta_v = \sqrt{D}$. The above bound becomes $\tilde{\mathcal{O}}\left(\sqrt{\frac{D^2+DK^2}{N}} + \sqrt{\frac{K}{m}}\right)$. Intuitively, the first term captures how well the common mapping and the prototypes learned on seen users' dataset translate to new unseen users. This term decays as the number of seen users N increases. The second term characterizes how well our few-shot preference localization for a new unseen user generalizes to unseen pairs of this user. This term indicates a sample complexity of $\tilde{\mathcal{O}}(K)$ and suggests efficient generalization, especially since K can be quite small in practice.

C.2. Numerical Simulation

In this section, we carefully and systematically examine how PAL adapts to a plurality of preferences via numerical simulations. To this end, we construct a synthetic, heterogeneous dataset similar to [12], where each item $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}I)$ and the user weight $\mathbf{W} \sim \mathcal{N}(\mathbf{0}, I)$. The true $f^\star$ is a linear mapping from $\mathbb{R}^d \to \mathbb{R}^d$. Let $K^\star$ denote the number of user prototypes and let $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{K^\star}$ denote the set of user prototypes, where each $\mathbf{p}_i \sim \mathcal{N}(\mathbf{0}, \frac{1}{d}I)$. We assume that the distance between any pair of user prototypes is lower bounded by some value δ.

Experiment Setting. We study a mixture setting, where each user is located in the convex hull of $\mathcal{P}$. Let $\mathbf{a}_i$ denote the i^{th} user. To learn this user's ideal point, we draw n pairs of items $\{\mathbf{x}_l, \mathbf{x}_r\}$ uniformly at random and assign the user's preference as $\mathrm{sign}(\|f^*(\mathbf{x}_l) - f^*(\mathbf{a}_i)\|_2 - \|f^*(\mathbf{x}_r) - f^*(\mathbf{a}_i)\|_2)$. We generate datasets with different $K^\star$, K, latent dimension d, number of prototypes, and number of samples per user n. We evaluate PAL-A on these synthetic datasets.

Results. Figure C.1 (a) shows that PAL can learn the user ideal points in the representation space. Figure C.1 (b) shows that the homogeneous reward model ($K = 1$) can only achieve sub-optimal performance when diverse preferences exist. Incorporating plurality via multiple learnable prototypes with PAL, we gain a significant 7% accuracy boost. Figure C.1 (c) shows that as we increase the number of training samples for seen users, PAL achieves higher

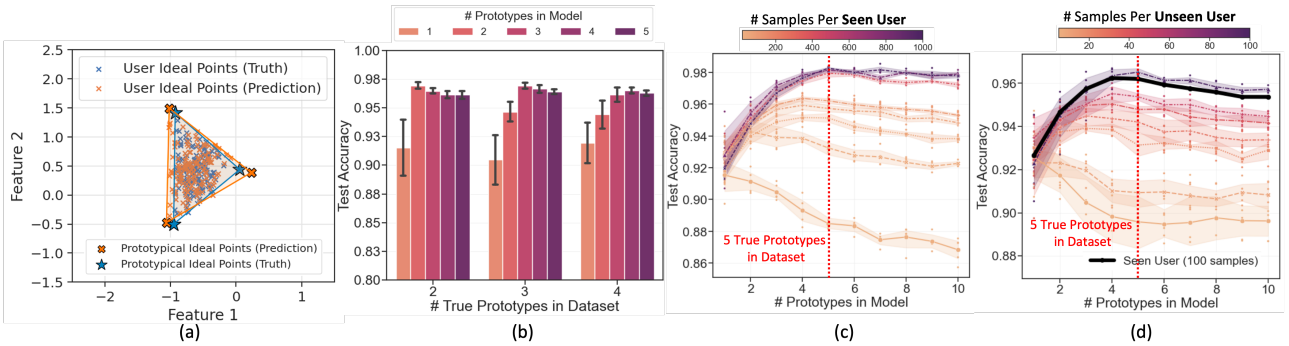


Figure C.1. **(a)** shows that PAL can accurately capture the user prototypes and each user's ideal point. We set $d = 2$ for visualization of the user's ideal point. **(b)** illustrates the performance of PAL on diverse preference datasets. **(c)** empirically highlights the importance of the number of samples per seen user and the number of prototypes in PAL. **(d)** demonstrates the sample efficiency of the PAL framework on unseen users. With as few as 40 samples, we can still achieve performance comparable to that of seen users.

test accuracy, and is also more accurate in capturing the true number of prototypes in the dataset. Figure C.1 (d) presents PAL's potential to generalize to new unseen users via few-shot learning to only learn their weights. We also studied a partition setting, where each user is drawn from $\mathcal{P}$ uniformly at random. Figure D.1 in the Appendix D.2 shows the learned ideal points under partition setting have similar trends to the mixture setting.

To further validate our findings, we also examine PAL's generalization properties via Anthropic personas [46], which we restrict to Appendix D.4 for brevity.

D. Experiment Details

D.1. General Procedure

We initialize our models with random weights for all of our experiments, including the prototypical weight for each user. We apply the softmax function to each prototypical weight to ensure that it is a probability vector. For the unseen user, we initialize its prototypical weight randomly. However, when we update the model using gradient descent, we fix the learned projectors and the prototypical points and only update the prototypical weight of the unseen users. During the training of the Large variant of PAL, we sample # batchsize samples from the preference datasets and concurrently update the shared foundation model, the mapping functions, and the corresponding user-specific weights for each sample. For the Tiny variant of PAL, we fix the foundation model and keep updating other components.

D.2. Numerical Simulation

Experiment Setup. We introduce the dataset simulation procedure in Section C.2. We use the following hyperparameters to generate the synthetic dataset: $d = 16, K = 3, N = 100, n = 100, \delta = 1$. We generate another 50 comparison pairs per user as the held-out dataset. Note that here we don't follow the prompt-guided item generation, i.e. conditioning x_c generates x_l and x_r. Instead, we directly draw the item $\{x_l, x_r\}$ from a normal distribution for simplicity. In the experimental setup, we apply a toy version of PAL-A, where the distance between the synthetic item and the user's ideal point is measured by $\|f(x) - f(u)\|_2$. We use a projection matrix (i.e. one-layer MLP network without bias term and activation function) as the model architecture. We randomly initialize the learnable parameters of prototypical user groups and user weights and use the Adam optimizer. The projector f has learning rate $5e-4$ and weight decay $1e-3$. The learning rate of the learnable parameters of prototypical user groups and user weights is $5e-3$. With the aim of good convergence, we train for 1000 epochs per run. We run multiple trials to explore the influence of each hyperparameter: 1) varying the number of samples of seen users $n = \{20, 40, 60, 80, 100, 400, 800, 1000\}$, $d = \{2, 16\}$, $K = 5$, $N = 250$, 2) varying the number of samples of new users $n_{new} = \{5, 10, 20, 30, 40, 50, 100\}$, $d = \{2, 16\}$, $K = 5$, $n = 50$, 3) varying the number of groups $K = \{2, 3, 4, 5, 6\}$, $d = \{2, 16\}$, $n = 50$, $N = 50 * K$. We plot the results of this experiment in Figure C.1 and discuss implications in Section C.2.

We consider two variants of modeling each user's ideal point through the lens of a shared group structure of preferences via prototypical ideal points (henceforth referred to as "prototypes"):

S1. Mixture Model: a user ideal point is a convex combination of K prototypes, i.e. lies in the convex hull of all prototypes.

S2. Partition Model: a user ideal point is one of K prototypes.

To visualize how well PAL can adjust to the *true* number of user groups present in data, via learnable prototypical

points to represent each group, we consider a simple setting with $d = 2$, $K^* = 3$, $K = \{1, 2, 3\}$ and $N = 100$ and plot the results in Figure D.1 for both partition and mixture settings. We also plot items in the partition setting in Figure D.2.

Partition Model : With only a single allowed assignment for a learnable prototype (Figure D.1a., $K = 1$), the predicted prototype is approximately the centroid of the true prototypes, i.e. the model tries to predict a good group assignment on average. Also note that since we have a single prototype, all predicted user ideal points lie on the prototype itself and performance is close to random. As we increase the degrees of freedom for learnable prototypes to two (Figure D.1b., $K = 2$), the model can predict one prototype close to a true prototype (in red), while the other predicted prototype is approximately an average of the blue and green true prototypes. User ideal points now lie in the convex hull of these two predicted prototypes, i.e. the line joining these points. It is only when we increase $K = K^*$, i.e. we match the "true" number of groups in the data (Figure D.1c., $K = 3$), the model can correctly predict close to all three true prototypes, and user ideal points are concentrated around the predicted prototypes. These observations extend to Figure D.2, where we additionally plot normally distributed items. Recall that in our modeling design, the distance between the user ideal point and the item reflects the user's preference; hence the closer the predicted user ideal point is to the true ideal point, the higher the performance.

Mixture Model : The results for the mixture model are similar to those of the partition model. With a single allowed assignment for a learnable prototype (Figure D.1(d.), $K = 1$), the predicted prototype is approximately the centroid of the true prototypes. As we increase the degrees of freedom to two (Figure D.1(e.), $K = 2$), predicted prototypes are close to two true prototypes, but one is neglected. When we increase $K = K^*$ (Figure D.1(f.), $K = 3$), matching the true number of groups in the data, the mixture model successfully predicts prototypes that lie close to all three true prototypes. This demonstrates that similar to the partition model, the mixture model can also adjust well to the true number of user groups present in the data.

D.3. Reddit TL;DR Summary (Text-to-Text)

Dataset. Reddit TL;DR Summary dataset curated by [62] contains a series of preferences over summaries generated by language models. For each pair of summaries, $\mathbf{x}_l$ and $\mathbf{x}_r$, a labeler i determines if $\mathbf{x}_l$ is preferred or not. Each pair is also accompanied by the unique identifier of the labeler. We used the variant of the TL;DR dataset proposed by [35], which uses the summary length as the preference. The majority group prefers longer summaries while the minority prefers shorter summaries.

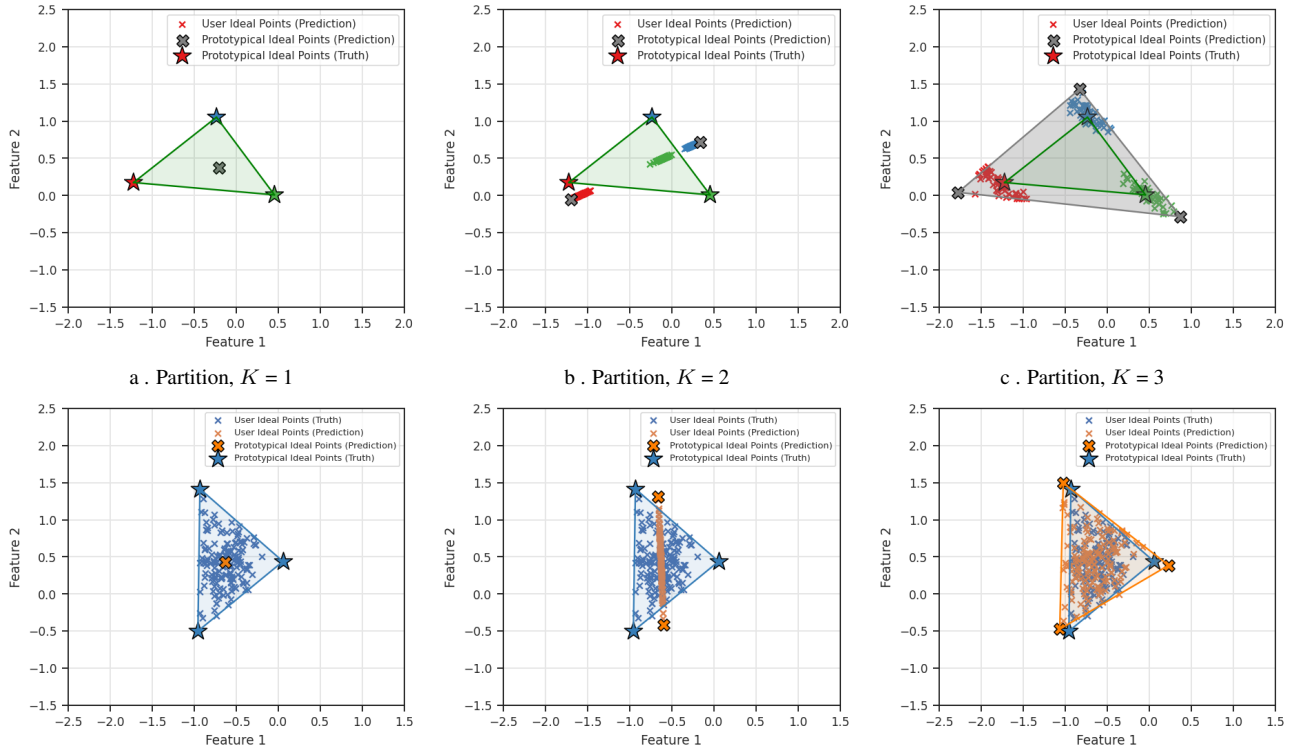
Setup. We train `PAL` with sentence embeddings from foundation models including `OPT-350M` [75], `DistilBERT` [52], `BGE-M3` [14], and `gte-Qwen2-1.5B` [36], which make up `Large` and `Tiny` variants depending on their size (#parameters). The loss design follows the typical loss of the Reward Model, we use the cumulative loss which weights the per-token reward loss. Details of the loss function, hyperparameter setting, unseen dataset, and training setup are deferred to Appendix D.3.

Baselines. We compare to personalized [35] and vanilla DPO [47].

Results. We train our model with 5 different seeds and report the mean and standard deviation. Figure 2 shows the performance of `PAL` with foundation models of different sizes. `PAL` shows strong accuracy-compute optimality with no hyperparameter tuning: compared to SoTA, `PAL-B-Large ( gte-Qwen2-1.5B)` is 5.9% more accurate on seen users with 4× less parameters while `PAL-B-Tiny (DistilBERT)` is on-par with 1000× fewer parameters. Furthermore, Figure D.3 illustrates `PAL`'s ability to generalize effectively to unseen users in few-shot settings. As the number of samples per unseen user increases, `PAL` progressively adapts to their preferences. With just 20 samples per unseen user, `PAL` achieves performance comparable to that of seen users. Additionally, as depicted in Figure D.4 (Appendix D), `PAL` exhibits superior performance over baseline models on unseen users, even when provided with as few as 2 samples.

Table 2 reports the performance of `PAL-B-Large (OPT-350M)` compared to SoTA P-DPO ([35]). We observe that `PAL`, with around 6.3 billion fewer parameters, is 1.7% more accurate on seen users, and 36% more accurate on unseen users (**C2**). Figure D.3 shows that for new users, `PAL` can match seen user performance with only 20 samples, showing its promising potential to flexibly adapt to new users. We further highlight `PAL`'s strong few-shot generalization to unseen users compared to baselines in Figure D.4 in the Appendix. Lastly, we also show exhaustive results for all our model configurations based on `OPT-350M` in Table D.1 and D.3 (Appendix D.3).

Details of seen dataset. We train `PAL` reward models on a variant of the Reddit TL;DR Summary dataset from [35]. In this variant, only the ten workers who gave the most feedback were chosen (Each user contains at least 1,000 samples). These ten workers are then divided into a majority and minority group, where the majority prefers the longer response, and the minority prefers the shorter response. More details about how the dataset is generated can be found in Section 6.1 of [35]. This processed dataset



a . Partition, $K = 1$ b . Partition, $K = 2$ c . Partition, $K = 3$

d . Mixture, $K = 1$ e . Mixture, $K = 2$ f . Mixture, $K = 3$

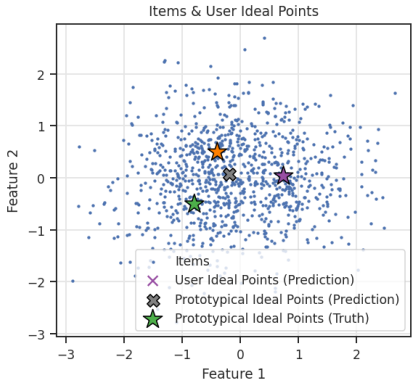
Figure D.1. In synthetic dataset experiments (Section C.2), we model user ideal points in two distinct ways: the partition model and the mixture model. To visualize how PAL performs in these settings, we set $d = 2$, $K^* = 3$, $K = \{1, 2, 3\}$ and $N = 100$ where each user ideal point and prototype is represented as a point in a two-dimensional space. In both scenarios, as the number of prototypes or user groups in our model (K) approaches the true number in the synthetic dataset (K^*), the PAL framework effectively learns both the prototypes and the heterogeneous user ideal points.

Table D.1. Seen user test accuracy of PAL vs. P-DPO (Li et al. v2) and vanilla DPO on the Reddit TL;DR Summary dataset. We run 5 trials with $K = 2$ and $K = 1$. Our method consistently performs better than the baseline methods. For the Large version of PAL, we use OPT-350M as the foundation model.

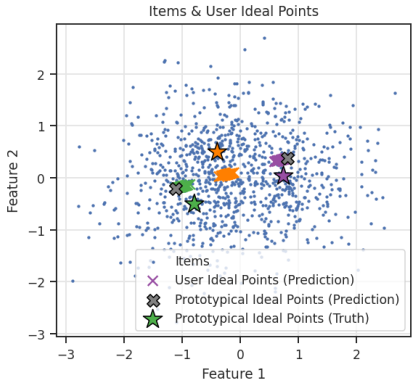
Model	Params	Seen User Test Accuracy
PAL-A-Tiny ($K = 2$)	1.6M	52.91 ± 0.55
PAL-B-Tiny ($K = 2$)	2.4M	79.54 ± 0.54
PAL-A-Large ($K = 2$)	352M	90.00 ± 1.89
PAL-B-Large ($K = 2$)	352M	$\mathbf{92.82 \pm 0.95}$
P-DPO Individual	6.7B	91.04
P-DPO Cluster ($K = 5$)	6.7B	91.12
PAL-A-Tiny ($K = 1$)	1.6M	49.99 ± 0.17
PAL-B-Tiny ($K = 1$)	1.6M	51.51 ± 0.08
PAL-A-Large ($K = 1$)	352M	$\mathbf{61.28 \pm 2.25}$
PAL-B-Large ($K = 1$)	352M	59.96 ± 3.45
Vanilla DPO	6.7B	58.91

contains 20,969 training samples, 2,330 validation samples, and 4,921 test samples. Each example consists of one user ID, one prompt, two responses, and the user's preference.

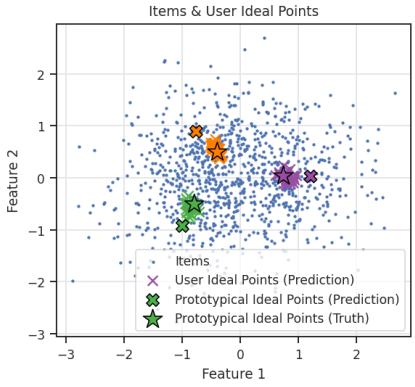
Details of unseen dataset. We selected all workers, excluding the ten used in the seen dataset, as candidates for the unseen dataset. From this pool, we filtered users with



a . $K = 1$, Test Acc = 72.2%



b . $K = 2$, Test Acc = 83.96%



c . $K = 3$, Test Acc = 91.26%

Figure D.2. Here we plot all items, the predicted user ideal points, and predicted and true prototypes in two-dimensional feature space. The items are normally distributed with $d = 2, K^* = 3, N = 100, n = 100$. As seen in the figure, when we set the number of prototypes in the model K equal to the true number of user groups $K^* = 3$, PAL can accurately capture the group structure and predict each user's ideal preference point, as well as the prototypes that represent each group ($K = 3$).

at least 100 valid comparison pairs (i.e., no missing values), resulting in a total of 31 users. We randomly assigned 70% of these users to prefer longer summaries, while the remaining users were designated as preferring shorter summaries. Based on these assignments, the users' preferences were relabeled. Given the varying numbers of few-shot samples in the training set, we partitioned each user's comparison pairs into training, validation, and test sets, resulting in multiple datasets.

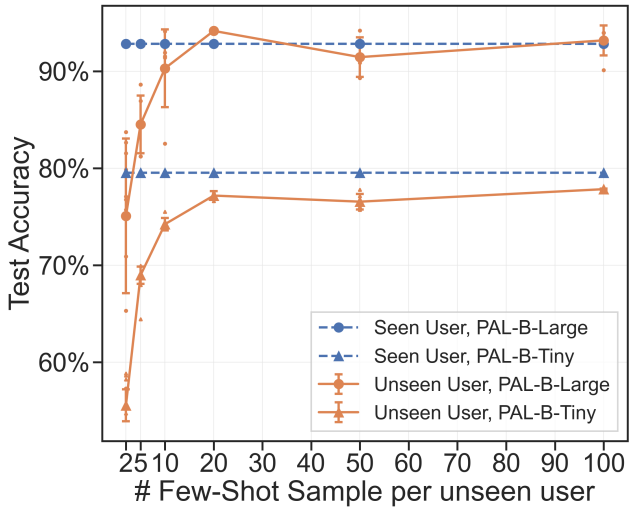


Figure D.3. On Reddit TL;DR, PAL generalizes well to unseen data using just 20 samples per unseen user (few-shot).

We leverage multiple pretrained LLMs, including OPT-350M, DistillBERT, Bge-m3, and gte-Qwen2,[6] as the base model, combined with two-layer MLPs utilizing GELU activation. In the Tiny variant of PAL, we fix the pretrained foundation model and only train the two-layer MLP, whereas in the Large variant, we also train the foundation model. In our PAL reward model, we set $K = 2$ and apply different learning rates for various model components: 9.65e-6 for pretrained LLMs (Large), 1e-4 for the two-layer MLPs, and 5e-3 for user weights. The higher learning rate of user weights can enhance the exploration of each user's weight across user groups. As with typical reward models, we train for only 1 epoch to avoid overfitting. The hyperparameter configurations are detailed in Table D.2. The training process takes roughly 1 hour on 1×RTX4090 GPU.

The loss design follows the typical loss of the Reward Model, we use the cumulative loss which weights the per-token reward loss,

$$L_{RM}(x, y_w, y_l; \theta) = \frac{\sum_{i=1}^{L} i \cdot \log \sigma \left(r(x, y_l^{(i)}) - r(x, y_w^{(i)}) \right)}{(L+1)L/2}$$

where x is the prompt, $y^{(i)}$ represents the LLM backbone prediction at generation timestep i, y_w and y_l sep-

[6] [35] use GPT-J 6.7B. However, the model card for that model on Hugging Face is broken.

arately represent the winning response and the losing response. Note that in our implementation of PAL-A, we concatenate the prompt and the item on the token level. Therefore, the embedding for an item produced by a foundation model already contains the information of the prompt. This implementation allows us to use the embedding directly without needing to concatenate it with the embedding of the prompt. Table D.1 reports the performance of our models and the numbers reported in [35]. We run our model 5 times and report the mean and standard deviation. We want to note that even though we did not conduct any hyperparameter tuning, With heterogeneous modeling ($K > 1$), PAL-B-Large achieves approximately **+1.8%** higher prediction accuracy compared to the state-of-the-art heterogeneous P-DPO ([35]) with 5 clusters. With homogeneous modeling ($K = 1$), PAL-A-Large is able to outperform vanilla DPO by **+2.4%**.

D.3.1. Few-shot generalization to unseen users

The procedure for few-shot generalization to unseen users is as follows: We randomly initialize the user weights, as done during seen user training, and then learn the user weights while keeping the LLM components and MLP projectors fixed. Since only the user weights need to be learned, the sample efficiency is significantly higher compared to seen user training. Results indicate that with just 20 samples per new user, we can achieve performance comparable to that of seen user generalization (Figure D.3). In Table D.3, we compare the performance of PAL-A-Tiny, PAL-A-Large, PAL-B-Tiny, and PAL-B-Large trained on OPT-350M embeddings when $K = 2$ and $N = 10, 20, 50, 100$. Our results show that PAL-A-Large, PAL-B-Tiny, and PAL-B-Large outperform the baselines substantially. We note that while P-DPO was state-of-the-art on seen users, it's performance drops off dramatically for unseen users (-36.6%), while **PAL-B-Tiny (91.63%) exceeds P-DPO's state-of-the-art seen accuracy (91.12%) with only 10 samples on unseen users**! This indicates the promising potential of PAL for cheap few-shot adaptation to new, unseen users in a sample-efficient manner.

We observe that increasing sample complexity N from 10 to 100 is impactful only for PAL-B-Tiny (+3%), while the other configurations gain only +0.4 to +1%. PAL-B vastly outperforms PAL-A for the same size and sample complexity (up to +27%). Lastly, Large models outperform their Tiny counterparts across sample complexities from +14.2 to +23.5%.

D.3.2. Choice of foundation model

The choice of foundation model is an important factor that impacts the performance of PAL, especially for PAL-A-Tiny and PAL-A-Large. This is because the foundation model directly decides the quality of the embeddings we obtain for the items. Table D.4 illustrates that there is a performance gap between using OPT-350M and DistilBERT as the foundation model, especially for Tiny variants where we do not train the foundation model. The existence of such a gap is possibly due to the fact that DistilBERT is an encoder-based model, which provides a better sentence embedding than the decoder-based OPT-350M.

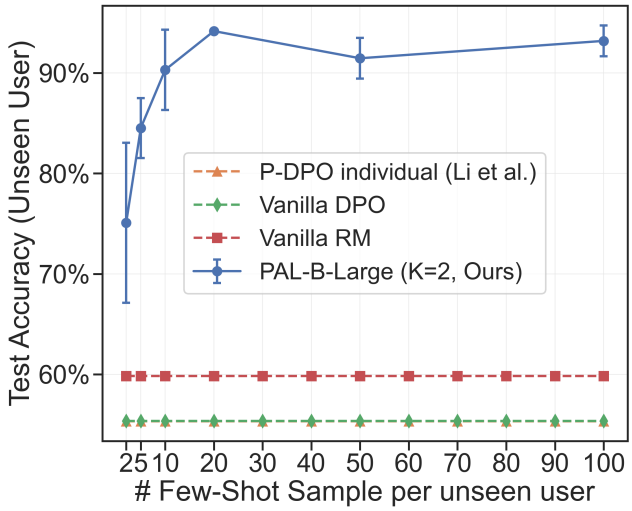


Figure D.4. We evaluate few-shot adaptation capabilities via unseen accuracy on the Reddit TL;DR dataset comparing to state-of-the-art [35] and simple vanilla DPO [47].

D.4. Persona (Text-to-Text)

Anthropic's Persona dataset [46] consists of a series of personalities (personas), each corresponding with 500 statements that agree with the persona and 500 statements that do not. We denote the set of statements that agrees with a persona ρ as $S(\rho)$. We construct a semi-synthetic dataset using Anthropic's Persona to evaluate PAL.

Dataset. Let $\rho = \{\rho_1, \ldots, \rho_{K^\star}\}$ denote the set of personas that exists in our semi-synthetic heterogeneous dataset with $K^\star$ "true" preference (prototypical) groups i.e. each person (user) has one of the $K^\star$ personalities. For each $\rho_j \in \rho$, we generate N synthetic *seen* and *unseen* users. For each seen synthetic user, we generate n_p queries that ask if the user agrees with a given statement from the persona dataset. For each unseen synthetic user, we generate $n_{p,\text{unseen}}$ queries. If the statement aligns with the persona ρ_j of the user, i.e. the statement belongs to $S(\rho_j)$, then the user answers yes, otherwise no. Table D.5 lists the personas used for each $K^\star$, and Figure D.5 shows a sample question.

Experiment Setup. We evaluate the performance of PAL-A-Tiny with hinge loss and model PAL-B-Tiny with logistic loss on the heterogeneous persona dataset in

Table D.2. The training hyperparameter setting of PAL reward modeling on Reddit TL;DR. The corresponding experiment setup is described in Section D.3.

Hyperparameters	Values
K	2
Batch size	4
Projectors	mlp-2layer-gelu-dropout0
Epoch	1
Learning rate of LLM	9.65e-6
Learning rate of projectors	1e-4
Learning rate of user weights	5e-3
Weight decay of LLM	0.0
Weight decay of projectors	0.01
Weight decay of user weights	0.0
Loss weighting	cumulative
Dimension of preference embedding	512
End of conversation token	<|endoftext|>
Maximum sequence length	600

Table D.3. Unseen user generalization of PAL compared to baselines on the Reddit TL;DR Summary dataset. Here N refers to the number of samples used to learn the weights for the unseen users, and seen accuracies are those reported in Table D.1. We note that with just 10 samples for each new (unseen) user, PAL-B-Large exceeds state-of-the-art performance of P-DPO with K=5 (91.12%), demonstrating the suitability of PAL for few-shot adaptation and generalization to new users.

Model	Seen Accuracy (%)	Unseen Accuracy (%)
P-DPO individual	91.04	55.34
P-DPO K=5	91.12	54.55
Vanilla DPO	58.91	55.37
PAL-A-Tiny $K=2, N=10$	52.91 ± 0.55	50.12 ± 1.20
PAL-A-Tiny $K=2, N=20$		50.55 ± 0.43
PAL-A-Tiny $K=2, N=50$		50.49 ± 0.39
PAL-A-Tiny $K=2, N=100$		50.40 ± 0.55
PAL-B-Tiny $K=2, N=10$	79.54 ± 0.54	74.88 ± 0.79
PAL-B-Tiny $K=2, N=20$		77.37 ± 0.37
PAL-B-Tiny $K=2, N=50$		76.19 ± 0.49
PAL-B-Tiny $K=2, N=100$		77.80 ± 0.07
PAL-A-Large $K=2, N=10$	90.00 ± 1.89	73.39 ± 1.82
PAL-A-Large $K=2, N=20$		74.04 ± 2.50
PAL-A-Large $K=2, N=50$		72.33 ± 2.36
PAL-A-Large $K=2, N=100$		74.39 ± 1.70
PAL-B-Large $K=2, N=10$	$\mathbf{92.82 \pm 0.95}$	$\mathbf{91.63 \pm 1.43}$
PAL-B-Large $K=2, N=20$		$\mathbf{91.72 \pm 1.40}$
PAL-B-Large $K=2, N=50$		$\mathbf{92.02 \pm 1.10}$
PAL-B-Large $K=2, N=100$		$\mathbf{91.97 \pm 1.91}$

various settings. Both model utitlize a 2-layer MLP as the f function. To examine the impact of various hyperparameters, we conduct experiments varying the number of true prototypes in the dataset $K^\star$, the number of prototypical groups used in the model K, queries per seen user n_p, and latent dimension d with a fixed number of users per group $N = 10,000$. Details of the values for each hyperparameter used are listed below:

(a) varying $K^\star = \{2, \ldots, 6\}$ and $K = \{1, \ldots, 8\}$ while fixing $n_p = 1000$, $d = 16$,

Table D.4. Comparison of performance of PAL-A-Tiny and PAL-A-Large with different foundation models on the Summary dataset.

Model	Foundation Model	Seen User Test Accuracy
PAL-A-Tiny ($K = 2$)	OPT-350M	52.91 ± 0.55
	DistilBERT	72.99 ± 1.21
PAL-A-Large ($K = 2$)	OPT-350M	90 ± 1.89
	DistilBERT	91.75 ± 0.41

Table D.5. Personas used across various "true" number of user groups $K^\star$ in our heterogeneous persona dataset.

$K^\star$	Personas
2	interest in art, interest in literature
3	interest in art, interest in literature, interest in math
4	interest in art, interest in literature, interest in math, interest in music
5	interest in art, interest in literature, interest in math, interest in music, interest in science
6	interest in art, interest in literature, interest in math, interest in music, interest in science, interest in sports

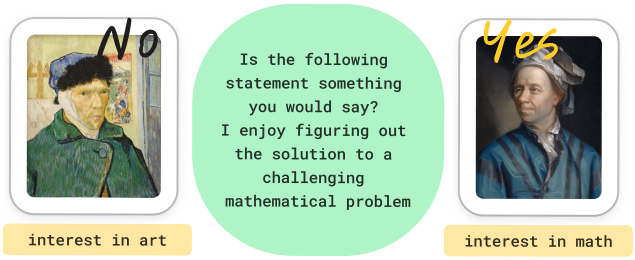


Figure D.5. An example of a pairwise comparison query with a prompt from our heterogeneous persona dataset generated using Anthropic's Personas. For example, a synthetic user assigned the persona *interest in art* will have ground truth $y = -1$ by answering no, whereas a synthetic user assigned the persona *interest in math* will have ground truth $y = +1$ by answering yes.

(b) varying $n_p = \{75, 100, 200, 500, 1000\}$ and $K = \{1, \ldots, 5\}$ while fixing $K^\star = 4$, and $d = 16$,
(c) varying $d = \{4, 8, 16, 32, 64\}$ and $K = \{1, \ldots, 5\}$ while fixing $K^\star = 4$, and $n_p = 1000$,
(d) varying $n_{p,\text{unseen}} = \{1, 10, 20, 50, 100, 200, 500, 1000\}$ and $K = \{1, \ldots, 5\}$ while fixing $K^\star = 4$, $n_p = 1000$, and $d = 16$.

Both PAL-A-Tiny and PAL-B-Tiny used the same value, except for d. This is because PAL-A utilizes a residual connection. Therefore, the latent dimension is fixed to 768, the dimension of the input embedding.

Results. We repeat these experiments five times and report the results on PAL-A-Tiny in Figure D.6 and on PAL-B-Tiny in Figure D.7. Both Figure D.7 and Figure D.6 (a, b) illustrate the generalization performance of PAL-A-Tiny and PAL-B-Tiny on the heterogeneous persona dataset. We observe that as $K \to K^\star$, the seen accuracy increases to 100% given a sufficient number of users and number of comparisons per user. Figure D.6 and Figure D.7 (b) shows that as we get more comparisons per user, we achieve better *seen user* accuracy, i.e. we can generalize to unseen pairs for users who are seen (provide training samples) in the dataset. Figure D.7 (c) shows that the size of latent dimension d does not affect the seen accuracy dramatically. Figure D.6 (c) and Figure D.7 show the accuracy for *unseen users*, i.e., users who do not provide training samples. When $K = 1$, no further learning is needed to generalize to new users. However, when $K > 1$, we require weights over the K prototypes for the new users to be learned. To learn these new user weights, as discussed in Section 2.2, we fix the K prototypes and the mapping f and use only a few test data samples to learn the user weights (**C4**). We use these learned weights to make predictions on the remaining test data. From Figure D.6 (c) and Figure D.7 (d), we see that for $K = 1$ the number of samples used to learn weights makes no difference since there are no weights to learn over a single prototype. For $K = 2$, we see that as we use more data for learning the new user weights, the performance shows diminishing returns until saturation. We also demonstrate that as the number of prototypes K increases, more comparisons per user are needed to learn the new user weights, since the dimension of the weight vector increases with K.

D.5. Pick-a-Pic (Text-to-Image)

Dataset. The Pick-a-Pic dataset is a large, open dataset designed to capture human preferences in text-to-image generation. It includes over 500,000 examples where users compare two AI-generated images based on a text prompt and

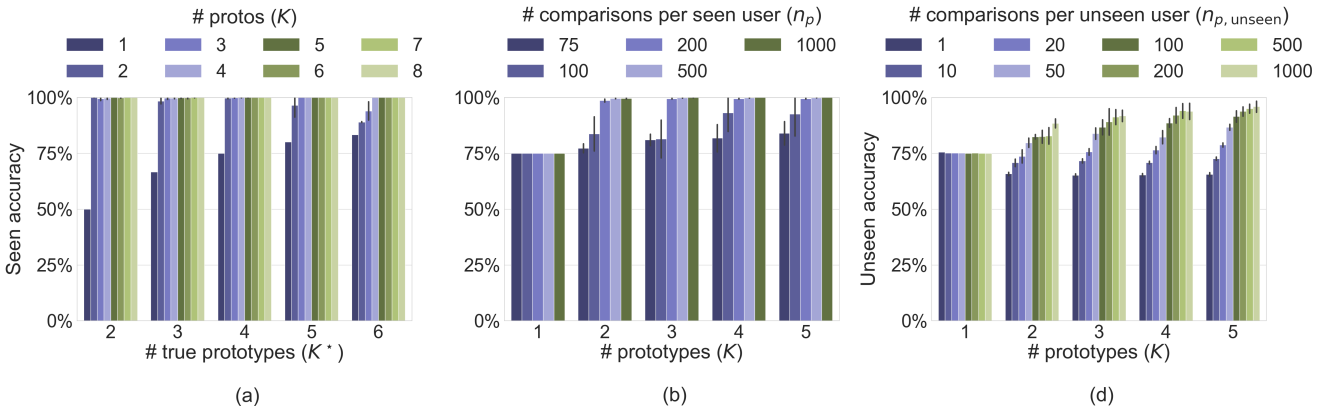


Figure D.6. Seen accuracy (a,b) and unseen accuracy (c) evaluated on the heterogeneous persona dataset across the number of prototypes K used in `PAL-A-Tiny`. The number of true prototypes $K^\star$ in (b, c) is 4. We vary (a) the number of true prototypes $K^\star$, (b) the number of comparisons per seen user n_p, (c) the number of comparisons per unseen user $n_{p,\text{unseen}}$. Since we used a residual connection in the `PAL-A-Tiny`, we could not vary the size of the latent dimension.

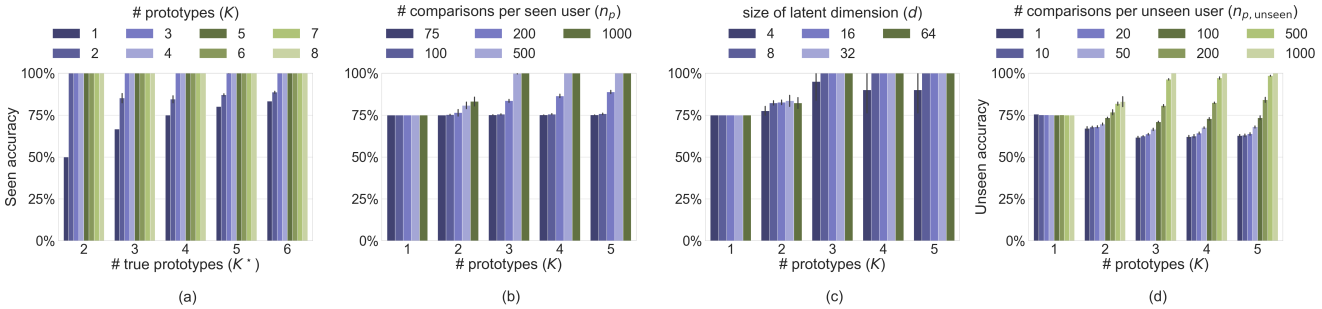


Figure D.7. Seen accuracy (a,b,c) and unseen accuracy (d) evaluated on the heterogeneous persona dataset across the number of prototypes K used in the `PAL-B-Tiny`. The number of true prototypes $K^\star$ in (b, c, d) is 4. We vary (a) the number of true prototypes $K^\star$, (b) the number of comparisons per seen user n_p, (c) the size of latent dimension d, (d) the number of comparisons per unseen user $n_{p,\text{unseen}}$.

choose their preferred one. This dataset is used to align models with human preferences.

Experiment Setup. We apply `PAL-B-Tiny` on the Pick-a-Pic dataset. Since the Pick-a-Pic dataset collection process requires strict rubrics, the labels collected from workers may not reflect the worker's diverse preferences. Thus we set $K = 1$ for the `PAL` model. We use two-layer MLP networks with ReLU activation and residual connections as the mapping functions. To avoid overfitting we set the dropout rate to 0.5 and weight decay to $1e-2$. We apply different learning rates for various model components: $1e-4$ for the two-layer MLPs and $5e-3$ for user weights.

D.6. Pick-a-Filter (Text-to-Image)

Dataset: due to the high level of "agreement" among labelers over image preferences on Pick-a-Pic v1 [31], we construct a semi-synthetic dataset by applying filters to a subset of Pick-a-Pic v1, which we call the Pick-a-Filter dataset. To construct the dataset, we consider only samples that have no ties, i.e. the labeler decides that one image is decisively preferable to the other, given the text prompt. As Pick-a-Pic provides unique and anonymous user IDs for all preference pairs, we consider a subset of users who provide samples in **both** the train and test sets (468 / 4223 users). We further only consider users who provide more than 50 labels (234 / 468 users) and sort the users by number of samples provided. We split these users into equal groups of 117 each, and we assume without loss of generality that the first group of users (G1) prefers "cold" tones (blue filter) and the second group (G2) prefers "warm" tones (red filter). Lastly, we arbitrarily consider the first 50 users (who provide the most number of samples) as "seen" users, i.e. users that provide samples in both the train and test sets of Pick-a-Filter. We add this seen vs. unseen distinction to evaluate how well `PAL` can adapt to unseen (i.e. new) users after training. Currently, our experiments on Pick-a-Filter (Section 3.2) train on v1-train-seen (116031 samples) and evaluate on v1-test-seen (3693 samples). We show the number of samples in each of these splits in Table D.6. After constructing splits, we apply the following filtering logic:

1. Apply "winning" and "losing" filters to appropriate images depending on label. For G1 the winning filter is

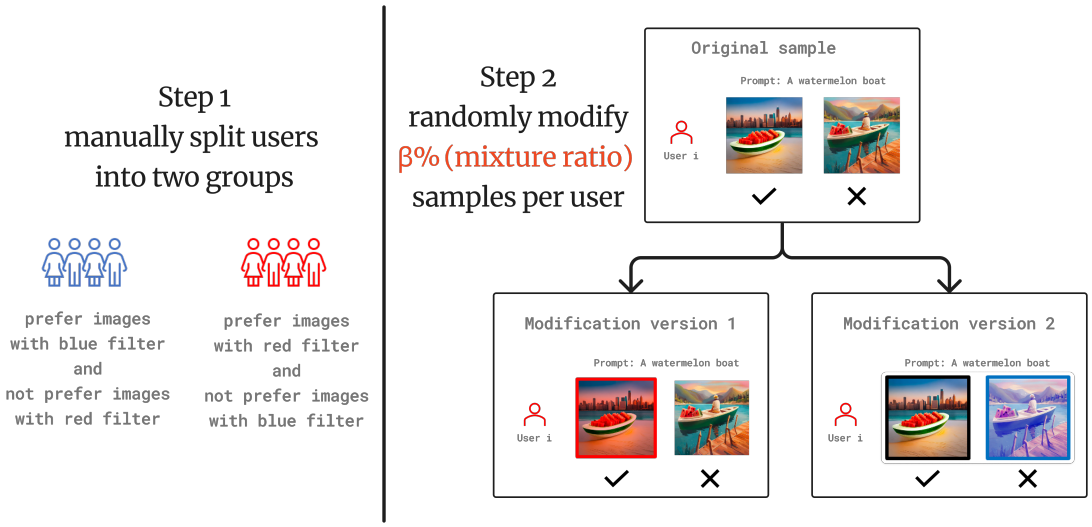


Figure D.8. The construction diagram for the semi-synthetic Pick-a-Filter dataset. It involves randomly selecting approximately 135,000 samples from the Pick-a-Pic v1 dataset and dividing the user IDs into two disjoint groups. We assume one group prefers images with "cold tone" (blue) filters and the other with "warm tone" (red) filters. To incorporate diverse color filter preferences, we randomly select $\beta\%$ of samples per user on which to apply filters.

Table D.6. Number of samples in each split of the newly constructed Pick-a-Filter dataset.

Category		Train	Val	Test
Group 1	Seen	58831	628	1597
	Unseen	9527	79	1886
	Total	68358	707	3483
Group 2	Seen	57200	404	2096
	Unseen	9402	52	1812
	Total	66602	456	3908

blue, and for G2 the winning filter is red.

2. Randomly shortlist $\beta\%$ of samples to add filters. The remaining $(1-\beta)\%$ of samples will remain unaltered (default images from Pick-a-Pic v1).
3. Randomly select 50% of above-shortlisted samples to apply a filter to only the winning image, and the remaining 50% to apply a filter to only losing image

We add these sources of randomness to make learning preferences on Pick-a-Filter less prone to hacking (e.g. the model could trivially learn to predict an image with a filter as the preferred image).

Experiment Setup. We choose 2-layer MLP networks with ReLU activation and residual connection as the prompt mapping function g_k and the output mapping function f. To avoid overfitting, we set the dropout rate to 0.5 and weight decay to $1e-2$. We use the Adam optimizer with a learning rate $1e-4$. To evaluate the model's performance, we use the checkpoint with the highest accuracy on the validation set.

E. Computational Resources

We conducted most of our experiments using 4×RTX 4090, each with 24 GB of VRAM. For the experiments involving a foundation model that has 1.3B parameters or more, we used 2 × A100, each with 80GB of VRAM. A typical run of the experimentw finished within 2 hours.