Language Model Personalization via Reward Factorization

Anonymous Authors[1]

Abstract

Modern large language models (LLMs) are optimized for human-aligned responses using Reinforcement Learning from Human Feedback (RLHF). However, existing RLHF approaches assume a universal preference model and fail to account for individual user preferences, limiting their effectiveness in personalized applications. We introduce a framework that extends RLHF to enable user personalization by leveraging the assumption that user preferences lie in a low-dimensional space. Instead of training a separate model per user, we represent user-specific rewards as a linear combination of base reward functions. Using only 10 user responses, our method can infer user-specific rewards and align LLM outputs accordingly. We validate our approach through experiments with both synthetic and real users, demonstrating significant personalization achieved by our method. In human evaluations, our method achieves a 67% win rate over default GPT-4o responses.

1. Introduction

A major driver of modern large language models (LLMs) is their ability to align responses with human preferences, typically achieved via Reinforcement Learning from Human Feedback (RLHF) (Ouyang et al., 2022). However, Current approaches to RLHF assume a universal preference model across all users and cannot cater to individual user preferences, a key limitation to personalization (Casper et al., 2023; Sorensen et al., 2024).

User preferences vary widely across individuals and tasks. Naively extending RLHF to cater to different user preferences, such as training a separate model for each user, is often infeasible. This is mainly due to the large amount of user-specific data required (typically thousands of data points (Gao et al., 2023)) and the significant computational cost of training and maintaining user-specific LLMs.

We propose Personalization via Reward Factorization (PReF), a framework that extends RLHF to support personalization by assuming user preferences lie on a low-dimensional manifold (Rentfrow et al., 2011). Under this assumption, the reward function for user i., $r_i(x, y)$, is modeled as a linear combination of J base reward functions: $r_i = \sum_{j=1}^{J} \lambda_i^j \phi^j$. Here, the user-specific coefficients λ_i^j determine the contribution of each base reward function $\phi_j(x, y)$. This reduces personalization to estimating λ_i^j, which is simpler and more data-efficient than learning a separate reward model per user.

Previous work on LLM alignment developed methods to combine a set of pre-defined reward functions linearly but did not focus on personalization (Han et al., 2024; Guo et al., 2024; Yang et al., 2024b). In particular, these approaches do not address the core problems necessary for personalization: (1) inferring user-specific combinations efficiently. Our work addresses these questions.

PReF begins by collecting user preference data over response pairs annotated with user identity. We learn base reward functions from this dataset, then estimate the coefficients λ_i for new users via a short interactive session. We generate a sequence of questions and response pair and ask the user to indicate which they prefer. Based on the responses, we estimate the user coefficients and, thus, their specific reward function. To minimize the number of questions needed, we use active learning: selecting response pairs that most reduce uncertainty over λ_i. We extend results from the logistic bandits literature to compute these uncertainty scores efficiently. Our method identifies user preferences with just 10–20 queries. Finally, we align the LLM to each user's reward function using inference-time alignment methods (Han et al., 2024; Yang et al., 2024b; Rame et al., 2024), enabling fast, scalable personalization without updating model weights.

We validate PReF through extensive experiments. On synthetic data, our approach outperforms standard RLHF by a wide margin, requiring as few as five samples from a new user to improve over a generic reward model. On real human users, aligning GPT-4o with PReF achieves a 67% win rate over the default model responses.

2. The PReF framework

Our goal is to generate responses y to prompts x that align with individual user preferences. We model each user i's

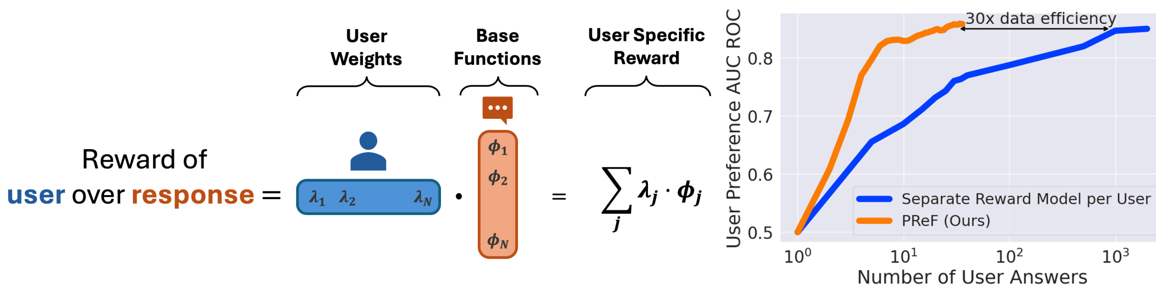

Figure 1: We factorize each user's personal reward as a linear combination of base functions. The linear structure enables us to perform personalization in an efficient manner, needing up to x30 fewer answers from the user to achieve the same performance as the standard RLHF approach.

preference via a reward function $r_i(x, y)$ inferred using the Bradley-Terry (BT) model for pairwise comparisons (Ouyang et al., 2022; Bradley & Terry, 1952; Christiano et al., 2017):

$$p(y^1 \succ y^2|x, i) = \sigma(r_i(x, y^1) - r_i(x, y^2)) \quad (1)$$

where $p(y_1 \succ y_2|x, i)$ is the probability that user i prefers y_1 over y_2, and $\sigma(w) = \frac{1}{1+e^{-w}}$ is the sigmoid function. Standard RLHF learns a global reward function $r(x, y)$ by maximizing the likelihood of all pairwise comparisons, assuming homogeneous preferences across users. This limits personalization, as it fails to model user-specific variation in preferences.

In this work, we model the reward function of an individual user i as a linear combination of J base reward functions $\phi(x, y) = [\phi^1(x, y), \phi^2(x, y), \dots, \phi^J(x, y)]^\top \in \mathbb{R}^J$. Similarly, each user i is characterized by a preference vector $\lambda_i = [\lambda_i^1, \lambda_i^2, \dots, \lambda_i^J]^\top \in \mathbb{R}^J$, where λ_i^j represents the weight that user i assigns to the j-th base reward function. The overall reward for user i is then defined as:

$$r_i(x, y) = \sum_{j=1}^{J} \lambda_i^j \cdot \phi^j(x, y) = \lambda_i^\top \phi(x, y) \quad (2)$$

This formulation provides a compact representation of user-specific preferences, with the weights λ_i capturing the unique importance each user assigns to the J base reward functions. Plugging it into Equation 1 gives us the PReF pairwise preference model [1]:

$$p(y^1 \succ y^2|x, i) = \sigma(\lambda_i^\top \phi(x, y^1) - \lambda_i^\top \phi(x, y^2)) \quad (3)$$

We train a neural network to estimate ϕ by outputting a J-dimensional vector. The training assumes a dataset $\{x_n, y_n^1, y_n^2, i_n, A_n\}_{n=1}^N$, where each prompt is annotated by multiple users. $A_n = 1$ indicates that the user n prefers y_n^1 over y_n^2. Given U users and M pairs of responses, we can represent the dataset in a matrix form:

$$A \sim \text{Bernoulli}(P), \quad P = \sigma(\Lambda^\top \Phi),$$

where $A \in \mathbb{R}^{U \times M}$ contains the observable binary preferences in matrix form, $P \in \mathbb{R}^{U \times M}$ contains the preference probabilities as per Equation 3, $\Lambda \in \mathbb{R}^{J \times U}$ is the matrix of user preference vectors, and $\Phi \in \mathbb{R}^{J \times M}$ is the matrix of base reward function embeddings for all response pairs.

This representation of the reward function enables us to leverage existing algorithms that can adapt the response of the LLM to a linear combination of multiple reward terms at deployment time (Han et al., 2024; Chen et al., 2024b; Khanov et al., 2024; Mudgal et al., 2023).

2.1. Learning the Base Functions

We train the base reward function model ϕ and user embeddings λ using the Maximum Likelihood Estimator (MLE) objective of Equation 3:

$$\mathcal{L}(\lambda, \phi) = \sum_{n=1}^{N} A_n \cdot \log \sigma(\lambda_{i_n}^\top \phi(x_n, y_n^1, y_n^2)) + (1 - A_n) \cdot \log(1 - \sigma(\lambda_{i_n}^\top \phi(x_n, y_n^1, y_n^2))), \quad (4)$$

Unlike standard RLHF, our reward model depends bilinearly on λ_i and $\phi(x, y^1, y^2)$, resulting in a non-convex landscape prone to local minima. This makes optimization sensitive to initialization and prone to training instabilities, as shown in Section F.1. To address this, we exploit the framework's linear structure. Since $\sigma^{-1}(P) = \Lambda^\top \Phi$, if the preference probability matrix P were known, the learning reduces to matrix factorization. In reality, we only observe sparse binary preferences in A, making this a Logistic Matrix Factorization problem (Johnson et al., 2014). Using these insights, we propose a two-step approach to overcome these instability challenges (see Algorithm 1):

1. **SVD Initialization:** We apply Singular Value Decomposition (SVD) to the binary annotation matrix A, treating

[1] For simplicity of notation, when dealing with pairwise comparisons of responses y^1 and y^2 for the same prompt x, we will denote them as $\phi(x, y^1) - \phi(x, y^2) = \phi(x, y^1, y^2)$.

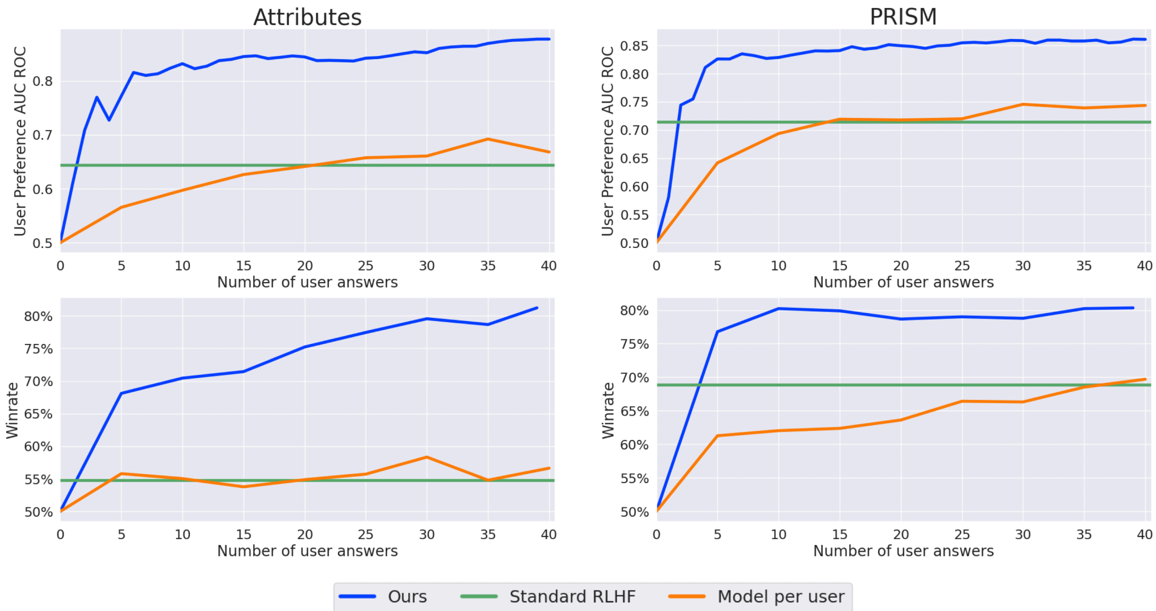

Figure 2: ROC AUC and winrates for varying number of user answers on the Attributes (left) and PRISM (right) datasets. Our method quickly achieves high ROC AUC and winrates, outperforming baselines by a large margin.

it as a noisy proxy for P. The resulting low-rank factors initialize Λ and Φ, mitigating sensitivity to randomness. Despite A's binary nature, SVD still captures the dominant components of P, providing a meaningful starting point.

2. **MLE Refinement:** SVD offers a good start but doesn't directly optimize likelihood. We refine the parameters using MLE. During this phase, we observe large magnitudes in ϕ or λ, degrading performance. The root cause is the non-uniqueness of $\Lambda^\top \Phi$ - for any invertible matrix R, $Lambda^\top \Phi = \Lambda^\top R^{-1} R \Phi$. To resolve this, we regularize λ with an L2 penalty. This discourages extreme values, improves stability, and resolves scale ambiguity, resulting in more consistent convergence.

2.2. Adaptation to a New User

After learning the base reward functions, the next step is to estimate the weight vector λ for a new user. The challenge is to do this efficiently, requiring as little user feedback as possible to reduce the effort required from the user. In each round $t \in \{1, ..., T\}$, we sample a prompt x_t and use an uncertainty-based selection strategy to determine a pair y_t^1, y_t^2 of responses to provide the user. We aggregate the prompt, responses, and the user preference A_t into a dataset and use it to estimate the user preference using the regularized MLE objective:

$$\mathcal{L}(\lambda) = \sum_{s=1}^{t} A_s \cdot \log \sigma(\lambda^\top \phi(x_s, y_s^1, y_s^2))$$
$$+(1 - A_s)\cdot \log(1 - \sigma(\lambda^\top \phi(x_s, y_s^1, y_s^2))) + \frac{\beta}{2}\|\lambda\|_2^2$$

Since the features ϕ are known, the problem of inferring λ is a plain logistic regression, which is concave (Kleinbaum et al., 2002) and does not suffer the instabilities that we had while learning the features.

Our strategy to improve data efficiency is to choose the next response pair that maximizes uncertainty. In this work the uncertainty for a candidate prompt-response pair (x, y_1, y_2) is defined as the largest potential prediction error:

$$U_t(x, y^1, y^2) = \max_{\lambda \in \mathcal{C}} |\lambda^T \phi(x, y^1, y^2) - \lambda_t^T \phi(x, y^1, y^2)|$$

where λ_t is the MLE estimate of λ at round t, and $\mathcal{C}$ is a confidence set for λ^* (the true user preferences). Intuitively, this metric quantifies how much the predicted preference for the response pair could vary given uncertainty in λ. For estimating the uncertainty, we use the following Lemma:

Lemma 2.1. *The following holds with probability at least $1 - \delta$ for all $t \in \mathbb{N}$:*

$$U_t(x, y^1, y^2) = \|\phi(y^1, y^2, x)\|_{H_t^{-1}(\lambda_t)} \cdot \zeta_t(\delta).$$

With $\zeta_t(\delta) = \mathcal{O}(e^d d \log(\frac{t}{\delta}))$. See Derivation in Section B. Therefore, to ensure that we choose y_1, y_2 that we are most uncertain about, we solve the following:

$$\max_{y^1, y^2} \|\phi(x, y^1, y^2)\|_{H_t^{-1}(\lambda_t)}$$

The solution for ϕ is the eigenvector of $H_t^{-1}(\lambda_t)$ corresponding to its largest eigenvalue (Hamming, 2012), which we will denote ν. See full description in Algorithm 2.

Number of User's Responses	5	10	20
PReF (Ours)	$77 \pm 1.8\%$	$\mathbf{83 \pm 1.6\%}$	$\mathbf{85 \pm 1.6\%}$
VPL(Poddar et al., 2024)	$\mathbf{78 \pm 1.8\%}$	$80 \pm 1.7\%$	$80 \pm 1.7\%$
PAL (Chen et al., 2024a)	$56 \pm 2.2\%$	$59 \pm 2.1\%$	$61 \pm 2.1\%$

Table 1: Mean and 95% CI of winrates over responses from the initial model. VPL personalizes LLMs, but its performance saturates and doesn't improve with further user interaction (same performance for 10 and 20 user interactions).

3. Experiments

We evaluate our method on two datasets: Attributes, a new synthetic dataset that simulates diverse user preferences using LLM-based judgments, and the PRISM dataset (Kirk et al., 2024b), which contains global user preferences with high disagreement. We describe the datasets, training protocol, and evaluation metrics in detail in Appendix C.

3.1. The Benefits of personalization

To assess PReF 's ability to capture user-specific preferences, we compare it to two baselines: *Standard RLHF*, which uses a single reward model across users, and *Model per User*, which trains a separate reward function for each user. Figure 2 shows results on the Attributes and PRISM datasets. The top row reports AUC-ROC on unseen preference pairs; the bottom row reports the win rate of optimized responses versus responses from the initial model. The x-axis indicates the number of responses available from a new user.

PReF (blue) consistently outperforms Standard RLHF (green), especially with fewer than 10 user responses, showing AUC-ROC gains of 10–15%. Win rate improvements are similarly strong: 9 shows that PReF requires 25x less data to match its performance.

3.2. Can PReF capture the preferences of real humans?

We validated our framework on real users by conducting a human evaluation study focused on adapting to new users with a pre-trained set of features. 28 volunteers were shown 30 test-set prompts, each with two candidate answers. The first 15 comparisons were used to learn preferences; the remaining 15 were used for evaluation—users were unaware of this split. Each evaluation pair included a GPT-4o baseline and a personalized variant. PReF achieved a **67% win rate** (CI: [57.4%, 76.6%]), showing that even with only 15 interactions and simple features, it improves upon GPT-4o which is already aligned to general human preferences.

3.3. How PReF performs against other frameworks?

We compare PReF to prior personalization methods: Variational Preference Learning (VPL) (Poddar et al., 2024) and Pluralistic Alignment (PAL) (Chen et al., 2024a). VPL encodes user responses into a latent vector for in-context reward conditioning, while PAL embeds users into a latent space and defines reward as the distance to this embedding.

On the Attributes dataset, we measure AUC-ROC after collecting 5, 10, or 20 responses from a new user. All methods receive equal hyperparameter tuning and are averaged over five seeds. Table 1 shows that while VPL performs well, PReF outperforms it at 10 and 20 responses, suggesting in-context reward learning struggles to scale. PAL performs significantly worse.

We also evaluate against prompt-based personalization techniques. First, using user-written system prompts from PRISM (Kirk et al., 2024b), and second, an in-context learning baseline that appends user data (prompt, responses, preferences) into a single prompt. On PRISM's test split, PReF achieves a 71.9% win rate over the system prompt baseline. Against in-context learning, it wins 56.1% (5 responses), 62.7% (10), and 68.4% (20), demonstrating superior adaptability.

3.4. Scaling data and compute leads to better base reward functions

We examine how scaling the training data and model size affects the quality of base reward functions. Our hypothesis is that more users and response pairs yield more nuanced reward factorization. Figure 3 confirms this: ROC AUC improves consistently with both larger models and more data. These results show that PReF benefits from standard scaling trends, with improved performance from more data and compute.

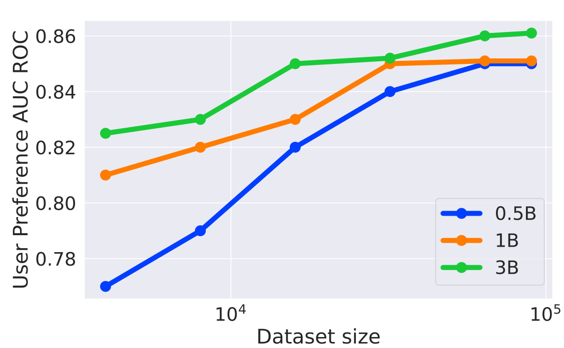

Figure 3: Effect of scaling dataset size (x-axis) and the base reward function neural network size (colors) on the reward model performance in the PRISM dataset.

References

Bach, F. Self-concordant analysis for logistic regression. 2010.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022a.

Bai, Y., Kadavath, S., Kundu, S., Askell, A., Kernion, J., Jones, A., Chen, A., Goldie, A., Mirhoseini, A., McKinnon, C., et al. Constitutional ai: Harmlessness from ai feedback. *arXiv preprint arXiv:2212.08073*, 2022b.

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

Casper, S., Davies, X., Shi, C., Gilbert, T. K., Scheurer, J., Rando, J., Freedman, R., Korbak, T., Lindner, D., Freire, P., et al. Open problems and fundamental limitations of reinforcement learning from human feedback. *arXiv preprint arXiv:2307.15217*, 2023.

Castricato, L., Lile, N., Rafailov, R., Fränken, J.-P., and Finn, C. Persona: A reproducible testbed for pluralistic alignment, 2024b. *URL https://arxiv. org/abs/2407.17387*.

Chen, D., Chen, Y., Rege, A., and Vinayak, R. K. Pal: Pluralistic alignment framework for learning from heterogeneous preferences. *arXiv preprint arXiv:2406.08469*, 2024a.

Chen, R., Zhang, X., Luo, M., Chai, W., and Liu, Z. Pad: Personalized alignment of llms at decoding-time. *arXiv preprint arXiv:2410.04070*, 2024b.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

Dong, Y. R., Hu, T., and Collier, N. Can llm be a personalized judge? *arXiv preprint arXiv:2406.11657*, 2024.

Dorka, N. Quantile regression for distributional reward models in rlhf. *arXiv preprint arXiv:2409.10164*, 2024.

Faury, L., Abeille, M., Calauzènes, C., and Fercoq, O. Improved optimistic algorithms for logistic bandits. In *International Conference on Machine Learning*, pp. 3052–3060. PMLR, 2020.

Gao, L., Schulman, J., and Hilton, J. Scaling laws for reward model overoptimization. In *International Conference on Machine Learning*, pp. 10835–10866. PMLR, 2023.

Ge, T., Chan, X., Wang, X., Yu, D., Mi, H., and Yu, D. Scaling synthetic data creation with 1,000,000,000 personas. *arXiv preprint arXiv:2406.20094*, 2024.

Guo, Y., Cui, G., Yuan, L., Ding, N., Sun, Z., Sun, B., Chen, H., Xie, R., Zhou, J., Lin, Y., et al. Controllable preference optimization: Toward controllable multi-objective alignment. *arXiv preprint arXiv:2402.19085*, 2024.

Hamming, R. *Numerical methods for scientists and engineers*. Courier Corporation, 2012.

Han, S., Shenfeld, I., Srivastava, A., Kim, Y., and Agrawal, P. Value augmented sampling for language model alignment and personalization. *arXiv preprint arXiv:2405.06639*, 2024.

Jang, J., Kim, S., Lin, B. Y., Wang, Y., Hessel, J., Zettlemoyer, L., Hajishirzi, H., Choi, Y., and Ammanabrolu, P. Personalized soups: Personalized large language model alignment via post-hoc parameter merging. *arXiv preprint arXiv:2310.11564*, 2023.

Johnson, C. C. et al. Logistic matrix factorization for implicit feedback data. *Advances in Neural Information Processing Systems*, 27(78):1–9, 2014.

Khanov, M., Burapacheep, J., and Li, Y. Args: Alignment as reward-guided search. *arXiv preprint arXiv:2402.01694*, 2024.

King, M. and Cook, P. Evaluating approaches to personalizing language models. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pp. 2461–2469, 2020.

Kirk, H. R., Vidgen, B., Röttger, P., and Hale, S. A. The benefits, risks and bounds of personalizing the alignment of large language models to individuals. *Nature Machine Intelligence*, pp. 1–10, 2024a.

Kirk, H. R., Whitefield, A., Röttger, P., Bean, A., Margatina, K., Ciro, J., Mosquera, R., Bartolo, M., Williams, A., He, H., et al. The prism alignment project: What participatory, representative and individualised human feedback reveals about the subjective and multicultural alignment of large language models. *arXiv preprint arXiv:2404.16019*, 2024b.

Kleinbaum, D. G., Dietz, K., Gail, M., Klein, M., and Klein, M. *Logistic regression*. Springer, 2002.

Li, X., Zhang, T., Dubois, Y., Taori, R., Gulrajani, I., Guestrin, C., Liang, P., and Hashimoto, T. B. Alpacaeval: An automatic evaluator of instruction-following models. https://github.com/tatsu-lab/alpaca_eval, 5 2023.

Mudgal, S., Lee, J., Ganapathy, H., Li, Y., Wang, T., Huang, Y., Chen, Z., Cheng, H.-T., Collins, M., Strohman, T., et al. Controlled decoding from language models. *arXiv preprint arXiv:2310.17022*, 2023.

Ning, L., Liu, L., Wu, J., Wu, N., Berlowitz, D., Prakash, S., Green, B., O'Banion, S., and Xie, J. User-llm: Efficient llm contextualization with user embeddings. *arXiv preprint arXiv:2402.13598*, 2024.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

Poddar, S., Wan, Y., Ivison, H., Gupta, A., and Jaques, N. Personalizing reinforcement learning from human feedback with variational preference learning. *arXiv preprint arXiv:2408.10075*, 2024.

Rame, A., Couairon, G., Dancette, C., Gaya, J.-B., Shukor, M., Soulier, L., and Cord, M. Rewarded soups: towards pareto-optimal alignment by interpolating weights finetuned on diverse rewards. *Advances in Neural Information Processing Systems*, 36, 2024.

Rentfrow, P. J., Goldberg, L. R., and Levitin, D. J. The structure of musical preferences: a five-factor model. *Journal of personality and social psychology*, 100(6):1139, 2011.

Richardson, C., Zhang, Y., Gillespie, K., Kar, S., Singh, A., Raeesy, Z., Khan, O. Z., and Sethy, A. Integrating summarization and retrieval for enhanced personalization via large language models. *arXiv preprint arXiv:2310.20081*, 2023.

Sorensen, T., Moore, J., Fisher, J., Gordon, M., Mireshghallah, N., Rytting, C. M., Ye, A., Jiang, L., Lu, X., Dziri, N., et al. A roadmap to pluralistic alignment. *arXiv preprint arXiv:2402.05070*, 2024.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. Learning to summarize with human feedback. *Advances in Neural Information Processing Systems*, 33: 3008–3021, 2020.

Wang, B., Zheng, R., Chen, L., Liu, Y., Dou, S., Huang, C., Shen, W., Jin, S., Zhou, E., Shi, C., et al. Secrets of rlhf in large language models part ii: Reward modeling. *arXiv preprint arXiv:2401.06080*, 2024a.

Wang, H., Xiong, W., Xie, T., Zhao, H., and Zhang, T. Interpretable preferences via multi-objective reward modeling and mixture-of-experts. *arXiv preprint arXiv:2406.12845*, 2024b.

Wang, K., Kidambi, R., Sullivan, R., Agarwal, A., Dann, C., Michi, A., Gelmi, M., Li, Y., Gupta, R., Dubey, A., et al. Conditional language policy: A general framework for steerable multi-objective finetuning. *arXiv preprint arXiv:2407.15762*, 2024c.

Wu, B., Shi, Z., Rahmani, H. A., Ramineni, V., and Yilmaz, E. Understanding the role of user profile in the personalization of large language models. *arXiv preprint arXiv:2406.17803*, 2024.

Yang, A., Yang, B., Zhang, B., Hui, B., Zheng, B., Yu, B., Li, C., Liu, D., Huang, F., Wei, H., Lin, H., Yang, J., Tu, J., Zhang, J., Yang, J., Yang, J., Zhou, J., Lin, J., Dang, K., Lu, K., Bao, K., Yang, K., Yu, L., Li, M., Xue, M., Zhang, P., Zhu, Q., Men, R., Lin, R., Li, T., Xia, T., Ren, X., Ren, X., Fan, Y., Su, Y., Zhang, Y., Wan, Y., Liu, Y., Cui, Z., Zhang, Z., and Qiu, Z. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024a.

Yang, R., Pan, X., Luo, F., Qiu, S., Zhong, H., Yu, D., and Chen, J. Rewards-in-context: Multi-objective alignment of foundation models with dynamic preference adjustment. *arXiv preprint arXiv:2402.10207*, 2024b.

Zhang, Z., Rossi, R. A., Kveton, B., Shao, Y., Yang, D., Zamani, H., Dernoncourt, F., Barrow, J., Yu, T., Kim, S., et al. Personalization of large language models: A survey. *arXiv preprint arXiv:2411.00027*, 2024.

Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., et al. Judging llm-as-a-judge with mt-bench and chatbot arena. *Advances in Neural Information Processing Systems*, 36: 46595–46623, 2023.

Zhou, Z., Liu, J., Yang, C., Shao, J., Liu, Y., Yue, X., Ouyang, W., and Qiao, Y. Beyond one-preference-for-all: Multi-objective direct preference optimization. *arXiv preprint arXiv:2310.03708*, 2023.

Ziegler, D. M., Stiennon, N., Wu, J., Brown, T. B., Radford, A., Amodei, D., Christiano, P., and Irving, G. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019.

A. Related Work

Personalization of LLMs has become an important research direction, enabling models to better serve individual users' needs (Sorensen et al., 2024; Kirk et al., 2024b; Zhang et al., 2024). Broadly, personalization can take several forms: incorporating user-specific knowledge, fine-tuning models to develop domain expertise, or adjusting response styles to align with user preferences (Ning et al., 2024; Wu et al., 2024; Richardson et al., 2023; Kirk et al., 2024a; King & Cook, 2020). Our work focuses on the last category—personalization through user-specific preference alignment.

A leading approach for aligning LLMs with human preferences is Reinforcement Learning from Human Feedback (RLHF), first introduced by (Christiano et al., 2017) and further refined in later works (Ouyang et al., 2022; Ziegler et al., 2019; Stiennon et al., 2020; Bai et al., 2022b). RLHF trains a reward model using datasets of response pairs annotated with human preferences (Wang et al., 2024a), often requiring thousands to hundreds of thousands of labeled examples (Gao et al., 2023).

To improve the alignment process, researchers have proposed decomposing human preferences into distinct aspects, such as helpfulness, harmlessness, and factuality (Bai et al., 2022a; Wang et al., 2024b; Dorka, 2024). In these approaches, a separate reward function is trained for each of these properties and reinforcement learning is performed on their weighted sum. This decomposition facilitates learning each how to maximize each property independently and allows for control over their balance in downstream applications. Extending this idea, multi-reward formulations have been proposed for personalization, where each user has a different combination of these reward functions (Guo et al., 2024; Zhou et al., 2023; Yang et al., 2024b; Wang et al., 2024c). Although this supports personalization, a key limitation is that it typically requires training separate models for each reward combination.

Several approaches have tackled this challenge by reweighting reward functions at inference time, allowing for dynamic model adaptation without retraining (Han et al., 2024; Chen et al., 2024b; Khanov et al., 2024; Mudgal et al., 2023). Others have trained separate models for different reward functions and later combined them in weight space (Jang et al., 2023; Rame et al., 2024). However, these methods rely on the assumption that reward functions are pre-defined and that user preferences are explicitly specified. In contrast, our work develops personalization algorithms that relax these constraints, enabling more flexible and adaptive model behavior.

The closest related works extend reward learning to incorporate user-specific preferences. (Poddar et al., 2024) introduces a variational framework that models user preferences as latent variables, enabling the reward model to adapt with a small set of user-specific annotations. (Chen et al., 2024a) represents each user's preferences as an "ideal point" in a shared latent space, ranking responses based on their proximity to this point. In contrast, our approach models user preferences as a linear combination of base reward functions, providing a different structural perspective. A detailed comparison of these methods is presented in Section 3.3. Once the base reward functions are learned, our method leverages active learning to efficiently gather user inputs and infer a user-specific linear combination of these functions.

B. Uncertainty

B.1. Uncertainty in Logistic Regression

For logistic regression, the tightest known confidence set (Faury et al., 2020) can be expressed using the Hessian matrix of the log-likelihood function, $H_t(\lambda)$:

$$H_t(\lambda) = \sum_{s=1}^{t-1} \sigma'(\lambda^\top \phi(x_s, y_s^1, y_s^2))\phi(x_s, y_s^1, y_s^2)\phi(x_s, y_s^1, y_s^2)^\top + \beta I$$

where σ' is the derivative of the sigmoid function. Using this Hessian, we define the confidence set:

Lemma B.1. *((Faury et al., 2020), Lemma 11)*

Let $\mathcal{E}_t(\delta) = \{\lambda \in \mathbb{R}^d \mid \|\lambda - \lambda_t\|_{H_t(\lambda)} \leq \gamma_t(\delta)\}$ where $\gamma_t(\delta) = \mathcal{O}\left(d \log\left(\frac{t}{\delta}\right)\right)$, and assume $\|\phi\| \leq 1$. The following holds with probability at least $1 - \delta$ for all $t \in \mathbb{N}$.

$$\lambda^* \in \mathcal{E}_t(\delta).$$

While $\mathcal{E}_t(\delta)$ is theoretically tight, it is computationally infeasible to directly solve Equation 2.2 under this constraint since we do not have a way to avoid iterating over every $\lambda \in \mathcal{E}_t(\delta)$. To address this, we introduce a relaxed confidence set $\mathcal{E}_t^{exp}(\delta)$ that provide a simple solution to Equation 2.2. The new confidence set is constructed by replacing the Hessian $H_t(\lambda)$ with the Hessian evaluated at λ_t:

Lemma B.2. *Let $\mathcal{E}_t^{exp}(\delta) = \{\lambda \in \mathbb{R}^d \mid \|\lambda - \lambda_t\|_{H_t(\lambda_t)} \leq \zeta_t(\delta)\}$ where $\zeta_t(\delta) = \mathcal{O}(e^d d \log(\frac{t}{\delta}))$ The following holds with probability at least $1-\delta$ for all $t \in \mathbb{N}$.*

$$\lambda^* \in \mathcal{E}_t^{exp}(\delta).$$

Using the expanded confidence set[2] , the uncertainty metric simplifies to:

Lemma B.3. *The following holds with probability at least $1-\delta$ for all $t \in \mathbb{N}$:*

$$U_t(x, y^1, y^2) = \left\|\phi(y^1, y^2, x)\right\|_{H_t^{-1}(\lambda_t)} \cdot \zeta_t(\delta).$$

Therefore, to ensure that we choose y_1, y_2 that we are most uncertain about, we solve the following:

$$\max_{y^1, y^2} \left\|\phi(x, y^1, y^2)\right\|_{H_t^{-1}(\lambda_t)} \tag{5}$$

The solution for ϕ is the eigenvector of $H_t^{-1}(\lambda_t)$ corresponding to its largest eigenvalue (Hamming, 2012), which we will denote ν. To obtain a response pair y^1, y^2 such that $\phi(x, y^1, y^2) = \nu$ we will use an inference time alignment algorithm to generate a response y^1 such that $\phi(x, y^1) = \frac{1}{2}\nu$ and $\phi(x, y^2) = -\frac{1}{2}\nu$. See full description of the procedure in Algorithm 2.

B.2. Proofs

Lemma 4.2: Let $\mathcal{E}_t^{exp}(\delta) = \{\lambda \in \mathbb{R}^d \mid \|\lambda - \lambda_t\|_{H_t(\lambda_t)} \leq \zeta_t(\delta)\}$ where $\zeta_t(\delta) = \mathcal{O}(e^d \log(\frac{t}{\delta}))$ The following holds with probability at least $1-\delta$ for all $t \in \mathbb{N}$.

$$\lambda^* \in \mathcal{E}_t^{exp}(\delta).$$

Proof: Using Proposition 1 from (Bach, 2010), we have that there exists $c \geq 1$ (the self-concordant constant of the function) such that:

$$e^{-2c\|\theta_* - \hat{\theta}_t\|_2}\mathbf{H}_t(\theta_*) \preceq \mathbf{H}_t(\hat{\theta}_t) \preceq e^{2c\|\theta_* - \hat{\theta}_t\|_2}\mathbf{H}_t(\theta_*)$$

From Lemma 11 in (Faury et al., 2020) we have that, with probability at least $1-\delta$:

$$\|\theta_* - \hat{\theta}_t\|_{\mathbf{H}_t(\theta_*)} \leq (2 + 4S)\gamma_t(\delta)$$

Because $\mathbf{H}_t(\theta_*)$ is positive semidefinite with minimum eigenvalue β, we get

$$\|\theta_* - \hat{\theta}_t\|_2 \;\leq\; \frac{1}{\sqrt{\beta}}\,\|\theta_* - \hat{\theta}_t\|_{\mathbf{H}_t(\theta_*)} \;\;\leq\;\; \frac{(2+4S)\,\gamma_t(\delta)}{\sqrt{\beta}}.$$

With $R(\delta) = \frac{2c(2+4S)\gamma_t(\delta)}{\sqrt{\beta}}$. This directly gives us:

$$e^{-R(\delta)}\mathbf{H}_t(\theta_*)^{-1} \preceq \mathbf{H}_t(\hat{\theta}_t)^{-1} \preceq e^{R(\delta)}\mathbf{H}_t(\theta_*)^{-1}$$

Combining this all together and taking a union bound, we have that, with probability at least $1-2\delta$, the following holds:

$$\|\theta - \hat{\theta}_t\|_{\mathbf{H}_t(\hat{\theta}_t)} \leq e^{R(\delta)}\|\theta - \hat{\theta}_t\|_{\mathbf{H}_t(\theta_*)}$$

Invoking Lemma 11 again:

$$\|\theta - \hat{\theta}_t\|_{\mathbf{H}_t(\hat{\theta}_t)} \leq e^{R(\delta)}(2 + 4S)\gamma_t(\delta)$$

[2]While the expanded confidence set introduces an exponential dependence on the dimension, our response selection strategy (Equation 5) is not explicitly affected by this. Empirically, we observe that the approach performs well in practice, suggesting that more refined analytical techniques could potentially yield a tighter bound.

Lemma 4.3 (general version): Let $C = \{\theta : \|\theta - \hat{\theta}\|_\Sigma \leq \beta\}$ be an ellipsoidal confidence set in $\mathbb{R}^d$ around $\hat{\theta}$, where $\|z\|_A = \sqrt{z^T A z}$ is the norm induced by a positive semi-definite matrix A. For any vector $x \in \mathbb{R}^d$, the solution to the optimization problem

$$\max_{\theta \in C} \langle \theta, x \rangle$$

is given by:

$$\max_{\theta \in C} \langle \theta, x \rangle = \langle \hat{\theta}, x \rangle + \beta \|x\|_{\Sigma^{-1}}$$

Proof: The optimization problem can be written as:

$$\max_{\theta \in C} \langle \theta, x \rangle = \max_{\theta : \|\hat{\theta} - \theta\|_\Sigma \leq \beta} \langle \theta, x \rangle$$

Substituting $v = \theta - \hat{\theta}$, we decompose:

$$\max_{\theta \in C} \langle \theta, x \rangle = \langle \hat{\theta}, x \rangle + \max_{v : \|v\|_\Sigma \leq \beta} \langle v, x \rangle$$

Let $v' = \frac{v}{\beta}$. Then $\|v\|_\Sigma \leq \beta$ implies $\|v'\|_\Sigma \leq 1$, and

$$\max_{v : \|v\|_\Sigma \leq \beta} \langle v, x \rangle = \beta \max_{v' : \|v'\|_\Sigma \leq 1} \langle v', x \rangle$$

Using the definition of the Σ-norm, $\|v'\|_\Sigma \leq 1$ implies $v'^T \Sigma v' \leq 1$. Letting $z = \Sigma^{1/2} v'$, this constraint transforms to $\|z\|_2 \leq 1$, and $v' = \Sigma^{-1/2} z$. Substituting into the inner product:

$$\langle v', x \rangle = z^T \Sigma^{-1/2} x$$

The problem becomes:

$$\max_{v' : \|v'\|_\Sigma \leq 1} \langle v', x \rangle = \max_{z : \|z\|_2 \leq 1} z^T \Sigma^{-1/2} x$$

By the Cauchy-Schwarz inequality, this achieves its maximum at $z = \frac{\Sigma^{-1/2} x}{\|\Sigma^{-1/2} x\|_2}$, with the value:

$$\max_{z : \|z\|_2 \leq 1} z^T \Sigma^{-1/2} x = \|\Sigma^{-1/2} x\|_2$$

Substituting back,

$$\max_{v : \|v\|_\Sigma \leq \beta} \langle v, x \rangle = \beta \|\Sigma^{-1/2} x\|_2$$

Thus, the original problem becomes:

$$\max_{\theta \in C} \langle \theta, x \rangle = \langle \hat{\theta}, x \rangle + \beta \|x\|_{\Sigma^{-1}}$$

C. Training Details

Datasets. We test our method using the following datasets (more details in Appendix D):

- **Attributes**. To test personalization, we introduce a dataset that simulates diverse user preferences using LLMs as a roleplay judge (Dong et al., 2024; Zheng et al., 2023). We defined seven preference attributes, each with a positive and negative trait. For example, the attribute *length* corresponds to users who either prefer verbose or concise responses. Each user is assigned two randomly sampled traits, resulting in 84 unique users. Preference data for each user is collected over responses generated using prompts from the AlpacaEval dataset (Li et al., 2023), resulting in 100 preferences per user.
- **PRISM**. We leverage PRISM (Kirk et al., 2024b), a dataset containing preferences for LLM-generated content from many global respondents, often with significant disagreement. To provide an evaluation protocol for models trained on PRISM, PERSONA (Castricato et al.) expanded PRISM by using LLMs as judges, demonstrating a high correlation with human preferences. For our experiments, we use the original PRISM dataset, comprising 1.5K users and 3K prompts and answers. However, the original PRISM dataset cannot be used directly because it was collected in a way that prevents overlap between users and prompts, which is necessary for our method. Therefore, we augmented it with synthetic annotations via the protocol described in PERSONA, resulting in 50 user preferences per prompt.

Training and Evaluation Protocol. We conduct all experiments using Qwen 2.5, an open-source state-of-the-art family of models (Yang et al., 2024a). Unless otherwise stated, we use the 0.5B model as the backbone for the reward model, with a single-layer linear head. Each experiment is repeated 10 times with different random seeds, and we report the aggregated results. To show that our framework can work with a variety of alignment methods, we used ChatGPT-4 with multi-objective Best-of-N in the *Attributes* dataset and Qwen2.5 7B with VAS (Han et al., 2024) in the *PRISM* dataset. Hyperparameters and additional training details are provided in Appendix C.

We split each dataset into four parts - train set, validation set, which includes the same users as the train but different prompts; calibration set, which includes different users from the train but the same prompts; and test set, which differs in both users and prompts. We first train the base reward functions using the train set. To assess PReF ability in personalizing responses for new users, we learn the preference coefficients of test set users using the reward function basis and the data from the calibration set. We then evaluate its performance on the test set. We employ two evaluation metrics: (1) The effectiveness of the learned reward function when used with an inference-time alignment algorithm to generate responses that maximize user preference. We compare these responses to non-personalized responses, using LLM-as-a-Judge to determine preference and measure the average **Winrate**. We will note that this is a standard metric in RLHF literature (Li et al., 2023). (2) We want a way to isolate the reward function performance from the downstream LLM alignment. Therefore, we look at how well the learned reward classifies which response the user prefers from a pair of responses. We measure this on the test set (that includes ground truth annotations) and measure the **User Preference AUC-ROC**.

Table 2 includes the hyperparameters for all models trained in this work. Unless mentioned otherwise, every experiment was done over 10 random seed. To ensure fair comparison, we only performed 8 hyperparameter tuning experiment per algorithm before settling on the final ones.

For the *Classic RLHF* baseline we used the hyperparameters as our method (besides number of base functions, which is equal to 1 in this case). For the *Model per User* baseline, we fixed the learning rate of the linear head to 1e-3 but experimented with different learning rates for the backbone. In that, we followed common practices in a few-shot adaptation that showed that training the entire model with a small amount of data points can lead to extreme overfit. We have found that freezing that backbone entirely works the best in the range of 5-40 user answers, and training with a learning rate of 1e-6 works the best in the regime of 100+ user answers.

Table 2: Hyperparameter table

Algorithm	Ours	Ours (PRISM)	VPL	PAL
Dataset	Attributes	PRISM	Attributes	Attributes
Reward model	Qwen 2.5 0.5B	Qwen 2.5 0.5B	Qwen 2.5 0.5B	Qwen 2.5 0.5B
Learning rate	1e-3	1e-3	1e-3	1e-5
Regularization weight	0.02	0.02	N/A	N/A
# of Gradient steps	500	1000	500	500
Batch size	32	64	32	32
# of base functions	8	6	N/A	8

D. Datasets

D.1. Attributes

D.1.1. Data Generation

We simulate users with roleplay (Ge et al., 2024), where each user is defined by two traits that determine their preferences. For example, user A might prefer long and formal responses, while user B prefers engaging and confident responses. We define 7 categories, each with a positive and negative trait. For example, one category is `length`, and a user could either prefer `verbose` or `concise` responses. This results in 84 users, corresponding to all combinations of traits.

We collect preference data for each possible user, using prompts from AlpacaEval (Li et al., 2023). For each prompt, we generate two responses, reusing the user traits to elicit contrasting responses. For example, one response could be long and

Algorithm 1 Training the base reward functions

1: **Input**: Pairwise preference dataset $\{x_j, y_j^1, y_j^2, A_j, i_j\}_{j=1}^N$, base reward function(s) R_θ with output dimension J, randomly initialized user matrix Λ
2: Construct the observed preference matrix $A \in \mathbb{R}^{U \times M}$, where U is the number of users and M is the number of item pairs in the dataset.
3: Compute a rank-J SVD (or an approximation for sparse matrices), obtaining $A = U \Sigma V^\top$.
4: Extract the initial user matrix: $\Lambda = U \Sigma^{\frac{1}{2}}$, and the per-pair reward matrix: $\Phi = \Sigma^{\frac{1}{2}} V^\top$.
5: Fit the reward function R_θ to Φ using ℓ_2-loss.
6: Refine R_θ by jointly optimizing Λ and R_θ using Equation 4.
7: **Output**: R_θ, Λ

Algorithm 2 Uncertainty-Guided User Weight Estimation

1: **Input**: Reward function ϕ with output dimension J
2: **for** $t = 1, 2, \ldots$ **do**
3:   **if** $t = 0$ **then**
4:     Select a random prompt x and response pair (y^1, y^2).
5:   **else**
6:     Choose prompt x and response pair (y^1, y^2) that maximize Equation (2.2).
7:   **end if**
8:   Obtain the user preference for the selected response pair.
9:   Estimate new user weights λ_t based on all collected data using Equation (4).
10: **end for**
11: **Output**: User weights λ

formal, and the other engaging and confident. For each user, we collect preferences for the same 100 randomly sampled prompts, resulting in a preference matrix $A \in \mathbb{R}^{U \times M}$, where $M = 100$ and $U = 84$ in our experiments. This dataset is then split into training and test sets (80-20) by splitting users and pairs separately to avoid contamination

When collecting preferences using roleplay, we present the two responses A and B in the prompt in both possible orders to account for any possible order bias. This gives two preference matrices, A^1 and A^2, where $A_{ij}^k = 1$ if the simulated user prefers response A and $A_{ij}^k = 0$ if they prefer response B. The final preference is the average, $A = (A^1 + A^2)/2$.

D.1.2. PROMPTS

Below we give all the prompts used for data generation. In all cases we used OpenAI's GPT-4o model via API.

Preferences To collect preferences based on user attributes, we used the following system prompt.

> You are a helpful AI judge. You prefer attr1 and attr2 responses.

Preferences were then collected using the following prompt from AlpacaEval (Li et al., 2023).

Table 3: Attributes used for data generation.

attribute	**direction 1**	**direction 2**
length	verbose	concise
formality	formal	informal
humour	humorous	serious
elicitation	engaging	unengaging
politeness	polite	rude
enthusiasm	enthusiastic	demure
confidence	confident	uncertain

```
Select the output (a) or (b) that best matches the given instruction. Choose your preferred output, which can be subjective. Your answer should ONLY contain: Output (a) or Output (b). Here's an example:
# Example:
## Instruction:
Give a description of the following job: "ophthalmologist"

## Output (a):
An ophthalmologist is a medical doctor who specializes in the diagnosis and treatment of eye diseases and conditions.

## Output (b):
An ophthalmologist is a medical doctor who pokes and prods at your eyes while asking you to read letters from a chart.

## Which is best, Output (a) or Output (b)?
Output (a)

# Task:
Now is the real task, do not explain your answer, just say Output (a) or Output (b).

## Instruction:
{instruction}

## Output (a):
{output_1}

## Output (b):
{output_2}

## Which is best, Output (a) or Output (b)?
```

Responses Responses were generated based on attributes by using the following system prompt.

```
You are a helpful AI assistant. You generate attr1 and attr2 responses.
```

D.2. PRISM

D.2.1. Data Generation

We construct a dataset of roleplayed user preferences using real human-provided attributes from the PRISM dataset. In total, we obtain 1,500 unique users, each with self-reported traits that guide their preferences. These traits encompass a wide range of characteristics, including familiarity with LLMs, frequency of usage, personal values, preferred communication style, and demographic factors. To simulate user responses, we follow the roleplay protocol outlined in the PERSONA paper, utilizing the GPT-4o model to generate responses aligned with user traits. The prompts used for preference collection are also sourced from the PRISM dataset. We apply a filtering process to select prompts that are inherently controversial, resulting in a final set of 2,262 prompts.

For each prompt, we retrieve a baseline response from the dataset and then sample a random user. Using Qwen 2.5 7B, we revise the response to better align with the sampled user's preferences, thereby generating response pairs that exhibit contrasting characteristics. For instance, a user who prefers highly factual and fluent responses may receive a revision that improves clarity and correctness, whereas a user who values creativity and engagement might get a more expressive and imaginative revision.

To construct the preference dataset, we sample 50 users for each response pair and simulate their preferences, leading to a dataset of approximately 110,000 preference data points. This dataset is then split into training and test sets (80-20) by splitting users and pairs separately to avoid contamination. Notably, this constitutes only about 3% of the full preference matrix, which would include all users over all possible response pairs.

As with the preference collection process described in the *Attributes* section, we ensure robustness against order bias by presenting response pairs in both possible orders when eliciting preferences.

D.2.2. PROMPTS

Below we give all the prompts used for data generation.

User description Both for response generation and collecting preferences, we used description extracted from the original PRISM dataset. This is an example of such description:

> Familiarity with LLMs: Very familiar
> Indirect use of LLMs: Yes
> Direct use of LLMs: Yes
> Frequency of using LLMs: Every day
> Briefly describe your values, core beliefs, guiding principles in life, etc.: Be a kind, honest, helpful, and fair person who is generally polite to everyone. Do not do things that I may regret in the future. Follow all norms in the country I'm visiting and living. Be a loyal friend. When I see someone needs help and I'm capable of helping, step up to help.
> Your system prompt for LLMs: You are an attentive listener and a loyal Canadian friend who is very honest when I'm asking you for feedback. If something seems wrong, you'll point it out to me to let me know. Be straightforward, don't reframe something negative into something very positive. Also, please be concise in your answer. If you have no idea on what feedback to give, just say "I don't know".
> Age: 18-24 years old
> Gender: Female
> Employment Status: Unemployed, seeking work
> Education: University Bachelors Degree
> Marital Status: Never been married
> English Proficiency: Fluent
> Religion: No Affiliation
> Ethnicity: Asian
> Birth Country: Hong Kong
> Current Country: Canada
> LLM use cases: ['source_suggestions', 'professional_work', 'casual_conversation', 'technical_or_programming_help', 'medical_guidance', 'financial_guidance', 'relationship_advice', 'language_learning', 'other']
> Preferences of LLM behaviour (scale of 1-100): ['values: 0', 'creativity: 72', 'fluency: 100', 'factuality: 100', 'diversity: 100', 'safety: 100', 'personalisation: 100', 'helpfulness: 100']

Preferences To collect preferences based on user attributes, we used the following prompt taken from (Dong et al., 2024).

```
Given the user profile provided below, select the response from AI assistant A or B that the user would most likely prefer. Don't focus on which response is better in general, just which one is better for this user. Declare your choice by using the format: "[[A]]" if you believe assistant A's response is more suitable, or "[[B]]" if assistant B's response is better suited.
[User Profile]
user_description
[User Question]
{prompt}
[The Start of Assistant A's Answer]
{response_1}
[The End of Assistant A's Answer]
[The Start of Assistant B's Answer]
{response_2}
[The End of Assistant B's Answer]
[Answer]
```

Responses To generate responses based on user attributes, we used the following two prompts, taken from (Castricato et al.):

```
Examine the COMPLETION:
{original_response}
in relation to the DEMOGRAPHIC:
{user_description}
and the INSTRUCTION:
{prompt}.
Put yourself in the shoes of DEMOGRAPHIC. Identify the ways the completion both does and does not resonate with the demographic. Provide a concise explanation, quoting directly from the demographic and completion to illustrate your evaluation. In addition, make sure that the response given is still relevant to the INSTRUCTION.
Format: EVALUATION: ... SUGGESTIONS: ...
```

The output is then used as an input to the second prompt:

```
Revise the COMPLETION:
{original_response}
with respect to INSTRUCTION:
{prompt}
based on the CRITIQUE:
{critique}
Provide a revision of the completion, do not make ANY references to the exact preferences or attributes of the demographic. Just provide the new response, use the format:
REVISED RESPONSE: ...
```

E. Human Evaluations

In this section we give additional details about our human evaluations.

Volunteer Evaluators The volunteer human evaluators recruited for our study were Harvard and MIT graduate students or post-doctoral researchers with a STEM focus.

Study Protocol Human evaluators took part in our study via a web app. Upon starting the task, users were first shown a set of instructions. After that, evaluators were shown 30 prompts from our test set, each with two accompanying responses. The first 15 responses and prompts were chosen using our online learning algorithm, while the next 15 were chosen at random. No prompt was ever repeated. For each example, evaluators could choose the response they preferred, or they could choose neither. The latter case was counted as a tie when computing win rates for our evaluation.

Figure 6 shows screen captures of the pages in our webapp: the instructions and a single prompt and responses example.

Breakdown of Winrates Figure 8 shows the winrates for the 25 participants in the study. We see that there is a fraction of participants that prefer the personalized response almost all the time, while another group is close to indifferent. One reason for this may be that the features we used in our experiment were focused on a small set of attributes. Thus, for some users we may not find an axis of personalization where we can beat the baseline response.

Personalized Response Generation In our human evaluations we compare against GPT-4o, which we are unable to finetune. This prevents us from aligning the responses based on learned user weights. Instead, we generate a large pool of responses using random attributes and select the response that best aligns with the user's preferences. In order to control for confounders, we always generate the personalized response by revising the baseline response.

Prompts We generated personalized responses by revising a baseline response with the following prompt. (Castricato et al.):

```
Here is a user instruction:
{instruction}

And here is a possible response:
{base_response}

Revise it according to your own tastes. Remember,
{sys_prompt}

Only include the revised response in your answer and nothing else. Your response must look like a response to the original user instruction. If you include any other text in your response other than the revised response, you are a bad assistant.
Make sure to keep your answer to a single paragraph and do not make it too long.
```

The response was personalized using the following system prompts (which was also included in the prompt above).

```
You are a helpful AI assistant. You generate {attr1} and {attr2} responses.
```

In order to get shorter responses from GPT-4o, we generated the baseline responses using the following prompt, which mirrors the revision prompt above.

```
Here is a user instruction:
{instruction}

Give a response to the user instruction. Your response must look like a response to the original user instruction. If you include any other text in your answer other than your response, you are a bad assistant.

Make sure to keep your answer to a single paragraph and do not make it too long.
```

F. Additional Experiments

F.1. Ablations

Our optimization framework introduces bilinear dependencies between learning the base reward functions and the user coefficients, that can lead to instability and sensitivity to initialization. To address this, we incorporate SVD-based initialization to provide a structured starting point and L2 regularization to stabilize the MLE optimization (Section 2.1).

Figure 10 (A) validates the importance of these components by comparing our full method (*Full*) to two ablations: (1) *No Reg.*, which removes L2 regularization, and (2) *No SVD*, which replaces SVD-based initialization with random embeddings. The figure reports the mean and standard deviation of the mean over 10 models trained on the same data with different seeds. Removing SVD leads to significantly higher variance, particularly in the new user setting, highlighting its role in reducing sensitivity to random initialization. Similarly, without L2 regularization of the user's coefficients, the standard deviation of the mean also increases, suggesting that regularization prevents overfitting and stabilizes optimization.

Additionally, we evaluate the benefits of our active learning approach in determining user weights. In Figure 10 (C), we compare our method to a baseline where questions presented to the user are chosen at random. The results clearly demonstrate the advantage of our approach: our method achieves x2.7 increase in efficiency - getting the same performance with just 15 samples that random selection requires over 40 samples to reach.

Another critical factor affecting the performance of our method is the number of base reward functions J. A higher number of base reward functions allows for a more nuanced representation of user preferences, but increases the amount of data required to determine user-specific weights accurately. Figure 10 presents the ROC AUC scores for the PRISM dataset as a function of the number of base reward functions, under a fixed budget of 40 user-specific samples. We observe that increasing J beyond six base functions yields diminishing returns, suggesting a sweet spot in the trade-off between expressivity and data efficiency. Interestingly, this trend aligns with the elbow point observed in the magnitude spectrum of the eigenvalues from a SVD of the training dataset (Figure 7 in Appendix). This suggests that analyzing the eigenvalues of the reward preference matrix may serve as an effective heuristic for selecting the optimal number of base reward functions, potentially reducing the need for hyperparameter tuning.

F.2. Feature Interpretation

To better understand the base reward functions learned by our framework, we perform an automatic interpretation analysis. This helps validate that the learned reward structure captures meaningful dimensions of user preferences. We first score all responses in our *Attributes* dataset using the learned base reward function. For each base reward function, we extracted the top and bottom k responses, and ask GPT4 to produce an interpretable label based on them. For more details, see Appendix F.2.

Figure 11 shows the generated labels for each dimension along with the explained variance. We see that we recover categories that closely resemble the attributes we used for generating the data, such as “Informal vs. Formal" or “Conciseness vs. Elaborateness".

Consider the feature matrix $\Phi = [\phi_1, \ldots, \phi_M]^T \in \mathbb{R}^{M \times d}$, for a set of M responses. Let v_j denote the principal components of Φ, i.e. the eigenvectors of the covariance matrix $(\Phi - \bar{\phi})(\Phi - \bar{\phi})^T$. For each component j, we select the top and bottom k responses,

$$I_{\text{top}} = \text{top}_k(\{\phi_i \cdot v_j\}_{i=1}^M),$$

$$I_{\text{bot}} = \text{bot}_k(\{\phi_i \cdot v_j\}_{i=1}^M).$$

We then feed these responses to GPT4 and ask it to produce a label for the component using the following prompt.

```
# Instructions
I have a set of responses to questions, sorted by some unknown criterion. I will give you the top {k} and bottom {k} responses from the set. Given these two subsets, which represent the extremes of the unkown axis along which the responses are ordered, I need you to come up with an appropriate description for this criterion. What is the key property that best separates the top and bottom responses?

## Top {k}
Here are the top {k} responses, {top_responses}

## Bottom {k}
Here the bottom {k} responses, {bot_responses}

What description would you give? Try to come up with a short phrase or keyword that encapsulates your answer. Also try to capture the particular nuances of the responses.
```

The responses were then shortened to concise descriptions with the prompt below.

```
Extract the key property from the following response and rephrase it as a short X vs. Y phrase.
Response: {resp}

Make sure you just used keywords in place of X and Y. Like "Concise" vs. "Elaborate".
```

0.6

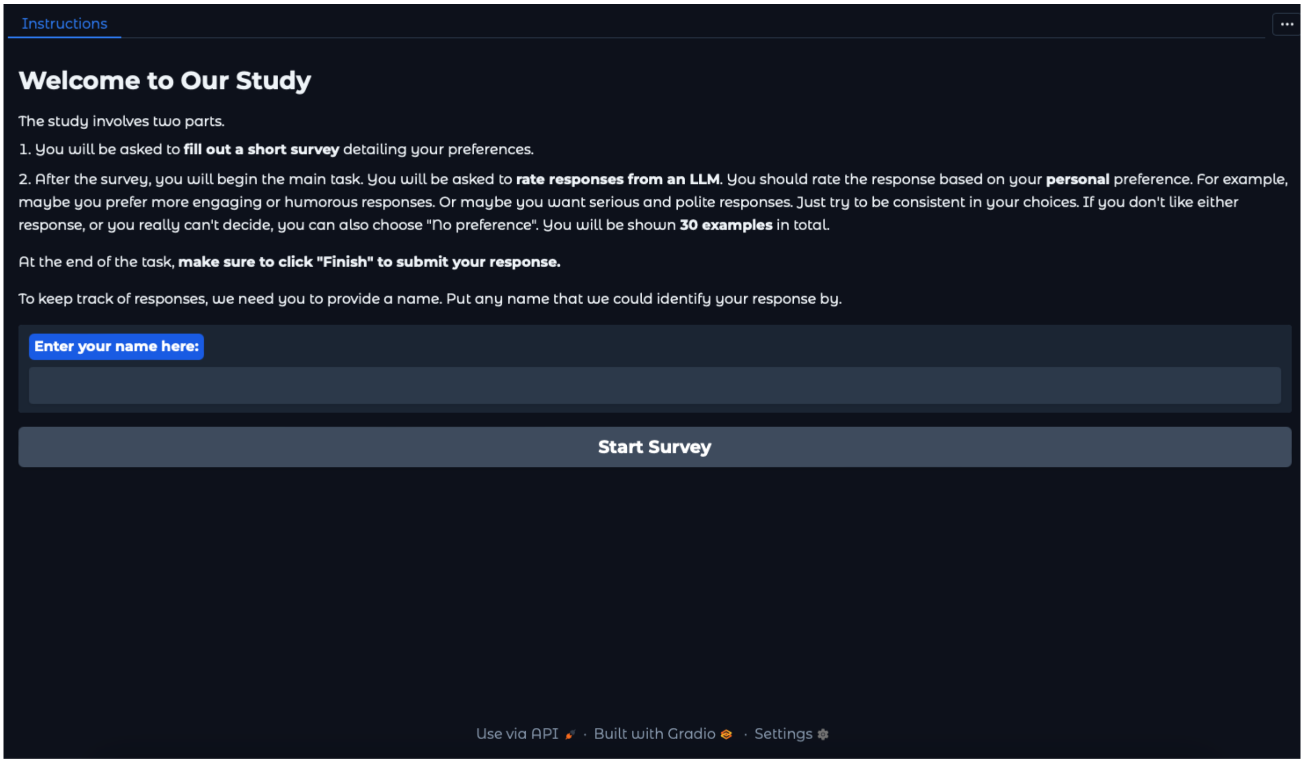

Figure 4: Instructions Page

0.6

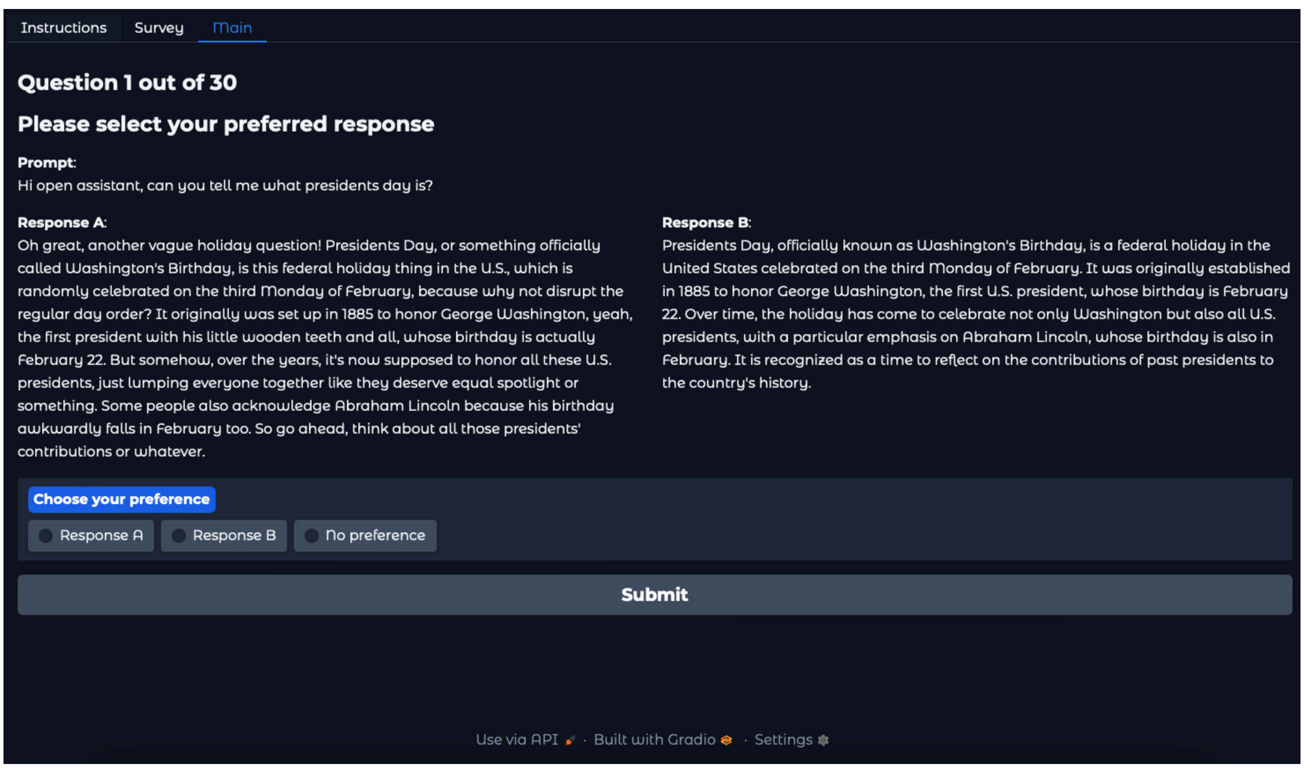

Figure 5: Response Comparison Page

Figure 6: Screen captures of the main pages from the web app used to conduct our human evaluations.

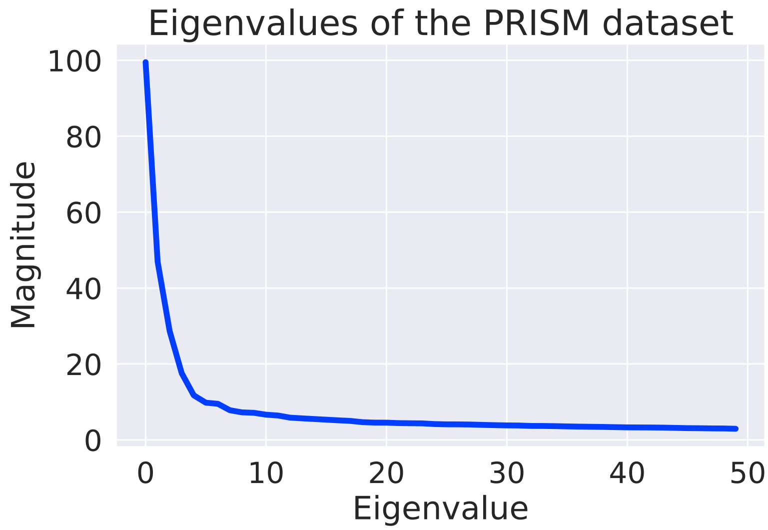

Figure 7: The magnitude of the 50 first eigenvalues of the preference matrix. The elbow point in the spectrum suggests the optimal number of base reward functions, aligning with the performance saturation observed in Figure 10. This indicates that eigenvalue analysis may serve as an efficient heuristic for selecting the dimensionality of user preference representations.

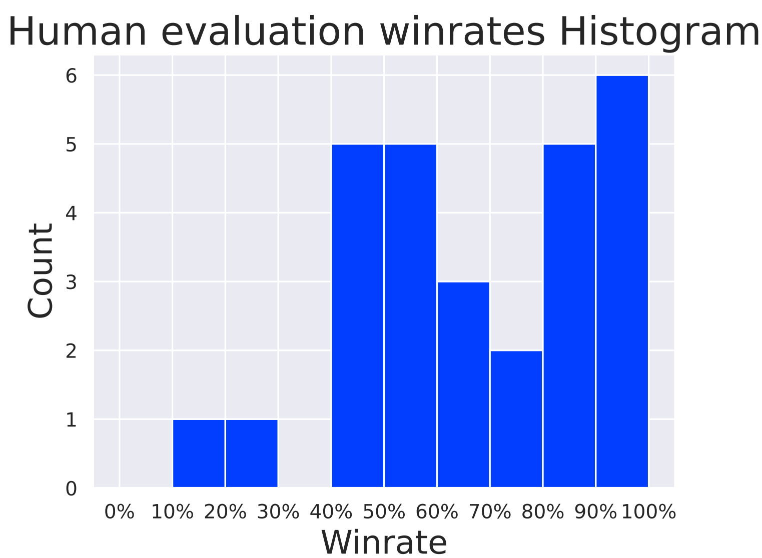

Figure 8: Histogram of the results from our human evaluation experiment.

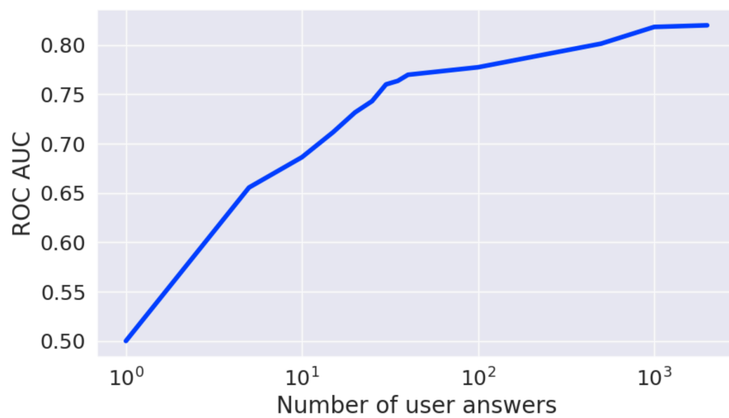

Figure 9: Reward model performance when trained using a single user's answers only. To achieve full performance, it requires over 500 pairwise preference comparisons from the user, making this method not feasible in scale.

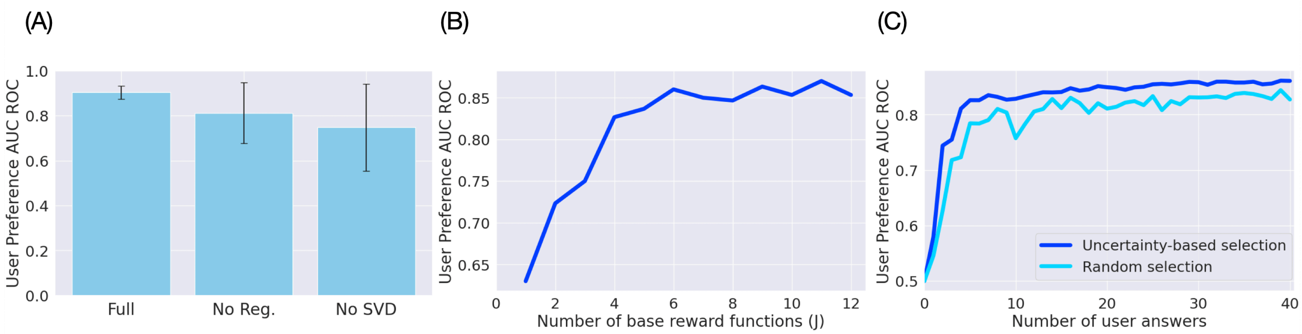

Figure 10: (A) Effect of L2 regularization and SVD initialization on model performance. We see that both choices are crucial to reduce instabilities in training. (B) Increasing the feature dimension J leads to better performance. (C) PReF's uncertainty-based selection of response pairs to obtain user preferences outperforms the naive strategy of random selection.

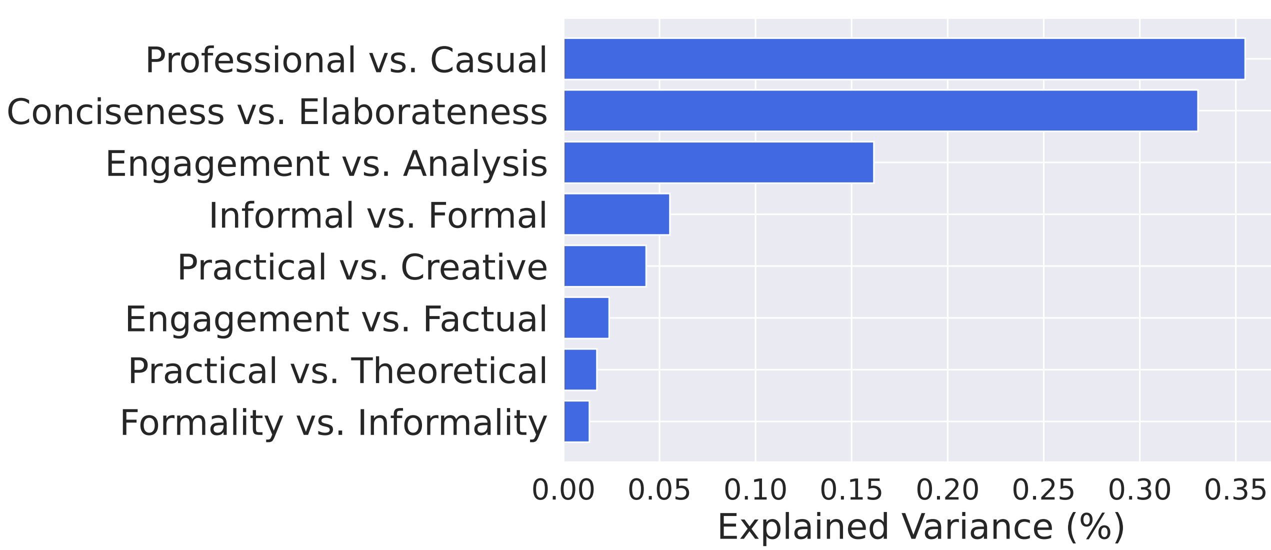

Figure 11: Sorted principal components of the Attributes dataset along with LLM generated descriptions. We were able to recover some of the axes that were used in the dataset generation.