Personalizing Reinforcement Learning from Human Feedback with Variational Preference Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Current Reinforcement Learning from Human Feedback (RLHF) techniques cannot account for differences in human preferences across a diverse population. When these differences arise, these frameworks average over them, leading to inaccurate rewards and poor performance for individual subgroups. To address the need for pluralistic alignment, we develop a class of multimodal RLHF methods based on a latent variable formulation - inferring a novel user-specific latent and learning reward models and policies conditioned on this latent without additional user-specific data. While conceptually simple, we show that in practice, this reward modeling requires careful algorithmic considerations around model architecture and reward scaling. To empirically validate our proposed technique, we first show that it can provide a way to combat under-specification in simulated control problems, inferring and optimizing user-specific reward functions. Next, we conduct experiments on pluralistic language datasets representing diverse user preferences and demonstrate improved reward function accuracy. We additionally show the benefits of this probabilistic framework in actively learning user preferences. This work enables learning from diverse populations, an important challenge naturally occurring in problems from robot learning to foundation model alignment.

1 Introduction

Reinforcement learning from human feedback (RLHF) has become the predominant technique for aligning foundation AI models to human values. The question then becomes: whose values? Current RLHF approaches [42] typically assume all end-users share the same set of values, not accounting for the range of social, moral, and political values informing preferences in real human populations. Specifically, RLHF relies on the Bradley-Terry-Luce (BTL) [12] preference model, which makes the *unimodal* assumption that all human preferences are derived from a single utility function. This fails to capture scenarios where preferences are multi-modal—due to fundamentally different utilities, e.g., one group prefers detailed responses, while another prefers concise ones (Figure 3). By doing maximum likelihood under the unimodal model, current methods learn a reward function that averages the multi-modal preferences, and this misspecification leads to *inaccurate* reward models. The policies optimized on these rewards fail to accomplish tasks per *any* of the distinct preferences.

To build foundation models serving a population, we need *pluralistic alignment* [48] methods that explicitly account for and adapt to the inherent plurality in human preferences. This suggests that preferences are not derived from a single utility function, but are affected by a *hidden* user context [47]. To address this, we formulate RLHF as a latent variable problem. Building on techniques from variational inference [9, 31], we propose a method—Variational Preference Learning (VPL) for multi-modal reward modeling. Intuitively, given a few user preference annotations, VPL leverages a variational encoder to infer a distribution over user contexts, allowing for a latent conditional reward model that accurately captures diverse user preferences. We derive an evidence lower bound (ELBO) for optimizing these rewards, facilitating the learning of reward distributions from large datasets of user preferences. In Section 3 we use our approach to learn latent-conditioned policies that can personalize to particular users at test time.

A challenge here is that binary comparisons lack information about reward scales (i.e. preference labels between A and B can only provide information about $r_A - r_B$), which leads to inconsistencies in the reward function across different users. To mitigate this, we show that a simple pairwise classification scheme [50, 40] can appropriately scale reward estimates, improving the optimization landscape for multi-user RL and enhancing downstream policy performance. Additionally, the latent variable reward models can measure uncertainty in the reward distribution [45]. So, in Section 3, we use our approach to actively query [6, 8, 7] users to infer their distinct preferences with fewer queries.

Overall, we introduce a latent variable framework for reward modeling that captures and adapts to diverse user preferences. We conduct a range of experiments across simulated robotics environments and two language tasks with conflicting user preferences. Our results show that in simulated domains, VPL accurately models rewards and improves task performance and personalization. We scale this method and further, use active learning to adapt efficiently to particular users with significantly fewer test-time queries. In the language domain, our approach outperforms existing models by more precisely predicting rewards that align with different users and objectives across multiple datasets.

2 Related Work

Reinforcement Learning from Human Feedback (RLHF): We focus on the problem of RLHF using the BTL model [12]. This has a rich history in the field of RL and robotics [53, 23, 1, 6, 8]. We specifically build on the framework outlined in Christiano et al. [14] and expanded in recent works [42, 2, 60, 49, 30, 47]. Our proposed method applies to *any* preference learning method, including recent techniques [44, 20] that circumvent reward modeling altogether.

RLHF under non-BTL models: Prior work has aimed to study non-BTL models of human preferences [10, 36, 35, 30, 50] to account for irrationality, or intransitivity [40, 50, 52]. However, our key argument is not about human irrationality, but about divergent but equally valid preferences between rational users. Thus, our work is complementary—VPL can easily be adapted to a non-BTL model.

Personalized RLHF: Some works [32, 33, 28] on pluralistic preferences largely focus on exploring the societal issues underpinning the need for personalization and introduce datasets with diverse annotations. Conitzer et al. [16] argues that Social Choice Theory provides insights for aggregating diverse preferences, but does not propose a technical method. Prior works looked at trading off conflicting alignment objectives (such as helpfulness vs harmlessness) through techniques like Pareto-optimal optimization [11, 13] or multi-objective RL [55, 18, 27]. Further, previous methods have approached personalization assuming explicit user groups or classes [21, 37, 58], while VPL doesn't assume access to such data. The closest work to ours is Distributional Preference Learning (DPL) [47], which aims to account for hidden context in RLHF, and proposes using a learned Gaussian or categorical distribution as the reward function. While DPL captures uncertainty in the inferred rewards, unlike VPL it cannot accurately predict a personalized reward for a particular user, perform active learning, or specialize a policy to particular users at test time.

3 VPL: Incorporating Latent Context into Preference-Based Learning

Technical Preliminaries In this work, we focus on preference-based learning [14, 1], which has two phases: (1) inferring a reward function from human-provided labels of ordinal preferences using a *maximum likelihood objective* (MLE) on the preferences; (2) reinforcement learning (RL) to train a decision-making policy that maximizes the rewards inferred in step (1). We note that while the BTL model accounts for some IID noise in preferences through the probabilistic formulation [14, 3, 19], it does not account for hidden-context and divergent human preferences [47], and does not allow the underlying reward models and policies to be personalized to specific users.

Variational Preference Learning The standard BTL formulation assumes all annotators $h \in \mathrm{H}$ share a single underlying reward function $r_\phi(s)$, but this doesn't hold in practice for diverse annotators. To model pluralistic preferences, we frame reward learning as a latent variable problem, where the latent variable z represents the hidden context influencing an annotator's underlying reward function (and thereby the preferences). This leads to a latent-conditional reward $r_\phi(s, z)$ and BTL model:

$$p_\phi(y = 1 \mid s_A, s_B, z) = p_\phi(s_A \succ s_B \mid z) = \frac{e^{r_\phi(s_A, z)}}{e^{r_\phi(s_A, z)} + e^{r_\phi(s_B, z)}} \tag{1}$$

The MLE objective for this model is intractable due to marginalization over the unobserved latent variable z *i.e.* $\max_\phi \mathop{\mathbb{E}}_{s_A, s_B, y \sim \mathcal{D}} [\log p_\phi(y \mid s_A, s_B)] = \mathop{\mathbb{E}}_{s_A, s_B, y \sim \mathcal{D}} \left[\log \int p_\phi(y \mid s_A, s_B, z) p(z) dz\right]$. To tackle this, a variational posterior approximation $q_\psi(z \mid \{(s_A^i, s_B^i, y^i)\}_{i=1}^N)$, conditional on multiple annotations $\{(s_A^i, s_B^i, y^i = h(s_A^i, s_B^i)\}_{i=1}^N$ provided by the same user h[1]. This results in a corresponding evidence lower bound (ELBO), $\mathcal{L}(\phi, \psi)$, for the intractable marginal $\log p_\phi(y \mid s_A, s_B)$:

$$\mathop{\mathbb{E}}_{\substack{h \sim \mathrm{H} \\ \{s_A^i, s_B^i, y^i = h(s_A^i, s_B^i)\}_{i=1}^N \sim \mathcal{D} \\ (s_A, s_B, y = h(s_A, s_B)) \sim \mathcal{D}}} \left[\mathop{\mathbb{E}}_{z \sim q_\psi(z \mid \{(s_A^i, s_B^i, y^i)\}_{i=1}^N)} [\log p_\phi(y \mid s_A, s_B, z)] - D_{\mathrm{KL}}(q_\psi(z \mid \{(s_A^i, s_B^i, y^i)\}_{i=1}^N) \parallel p(z)) \right] \tag{2}$$

Intuitively this objective encodes a set of user-provided annotations $\{(s_A^i, s_B^i, y^i)\}_{i=1}^N$ into a latent distribution using the encoder q_ψ, and then learns a latent-conditional reward function $r(s, z)$ using the contextual BTL model, with a regularization term $(D_{\mathrm{KL}}(q_\psi(z \mid \{(s_A^i, s_B^i, y^i)\}_{i=1}^N)) \parallel p(z)))$ against a prior $p(z)$. We describe details further in Appendix E.5. VPL clusters users without explicit class labels, unlike previous methods [58, 21, 37]. It only requires a few preferences from the same user, a minimal addition to standard RLHF [14], and can be easily collected in batch mode [8].

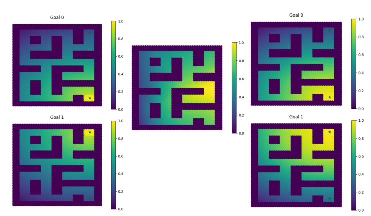
Personalized, latent-conditioned policies. We can learn a latent-conditioned policy $\pi_\theta(\cdot|s, z)$ using any RL algorithm [46, 24, 34] to optimize the latent-conditional reward maximization objective: $\pi_\theta = \arg\max_\theta \mathop{\mathbb{E}}_{\pi_\theta, z \in p(z)} [\sum_t \gamma^t r_\phi(s_t, z)]$, where $z \sim p(z)$. At test time, a user's latent context z is estimated via posterior inference using preference queries. The personalized policy $\pi_\theta(\cdot|s, z)$ is then deployed (complete algorithms in Appendix F). We further introduce an approach to scale the rewards across different z in Appendix A to improve optimization for this multi-objective problem.

Active Learning of preferences to minimize latent uncertainty. Using the posterior distribution in VPL over the latent vector z, we can leverage active learning to select preference queries that most effectively reduce uncertainty about z. An information gain-based approach based on prior work [6, 8, 41] ensures that users are required to answer the minimal number of questions necessary to effectively personalize the model to user preferences. (Details in Appendix B).

Scaling VPL to Large Language Models (LLMs). The architectural considerations for scaling VPL to LLMs are presented in Appendix C.

4 Experiments

In our experiments, we address four key questions: (1) Can VPL learn a multi-modal reward distribution from diverse user preferences? (2) Do the inferred latent user vectors enable multi-task personalized policy learning? (3) Can the posterior be used to actively query preferences for improved latent estimation? (4) Does VPL help to make LLM reward models more pluralistically aligned? We validate VPL on four diverse simulated control tasks: Figure 2): Maze-Navigation, Ravens-Manipulation, Habitat-Rearrange, and Habitat-Tidy (details in Appendix E). For LLMs, we compare the reward modeling performance of our method against two baselines: the vanilla BTL model and DPL [47]. We experiment with two LLMs: GPT2 [43] and Llama2-7B [51], and two pluralistic preference datasets - Pets and UF-P-N. (Details in Appendix E.4)



(a) GT (b) BTL (c) VPL

Figure 1: Ground truth (a) shows annotators prefer the robot to navigate to two different goals. Unimodal BTL (b) averages over the two modes, leading to an inaccurate reward function and poor policy performance. VPL (c) reconstructs diverse preferences and learns z conditioned policies that can reach either goal.

[1]Having multiple annotations are important here to be able to accurately infer the user's latent vector z

Can VPL capture multi-modal reward functions from a dataset of diverse preferences? We generate preferences using multi-modal reward functions across various didactic and control experiments, as shown in Figures 6, 1, 2, 12 and 11. The BTL baseline with an MLP averages different underlying rewards, leading to inaccurate reward models (Figure 1b). It converges to majority preferences, ignoring minority groups (Figure 12). DPL [47] captures uncertainty due to underspecification but cannot recover individual rewards, estimating high-variance rewards for each user (Figures 6, 11). In contrast, VPL infers hidden user contexts via a latent variable approach, accurately recovering the multi-modal reward distribution.

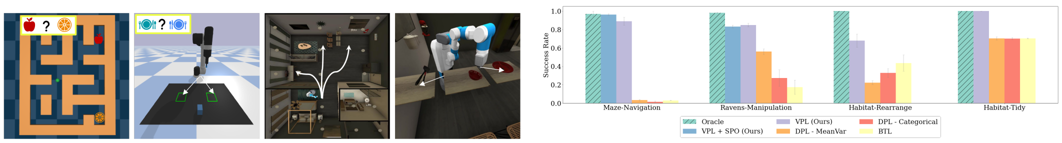


Figure 2: Performance of a downstream policy on diverse control and reasoning tasks, using the rewards trained using different baselines. We report the mean and standard error over five seeds. Note: Habitat tasks have a one-step greedy policy so reward scaling and SPO+VPL are not required.

Do distributional reward functions enable learning a steerable multi-task policy? As discussed, baselines like BTL [14] and DPL [47] average reward modes across users, leading to inaccurate reward functions. Across the different environments, this results in the policy converging to inaccurate goals or randomly selecting one, failing to adapt to user preferences at test time. In contrast, VPL outperforms the baselines in task success rate, aligning with users' reward functions. VPL accurately infers goals and achieves performance comparable to a goal-conditioned oracle. We note that scaling the rewards via VPL + SPO improves the performance of multi-task RL for optimizing diverse user preferences. In Appendix D.3, we scale to settings with a larger ($\sim$100) number of user-providing preferences, and we analyse the affect of context length on performance in Appendix D.5.

Can we use the posterior to actively query preferences for better latent estimation? In Appendix B, we show that active learning enables personalization to users with as low as $\sim$2 test-time queries.

Does VPL help to make LLM reward models more pluralistically aligned? In Table 1, VPL is able to learn a more accurate reward model across all the datasets, capturing the multi-modality in the language preference data. This indicates that VPL can infer the latent representation of the user's preferences z from a few annotated samples, and successfully adapt the reward model. In Figure 13, we observe that the encoder can implicitly cluster users based on their preferences. In contrast, the baselines are unable to fit the datasets because they are unable to account for divergent preferences. Because the datasets are imbalanced, the baselines can sometimes perform better than random guessing by fitting only the preferences of the majority group.

Table 1: We compare the accuracy of different reward models trained on the two datasets. We report the mean and standard deviation of performance of GPT2-based models on three seeds, and one seed for Llama models.

	GPT2				Llama2-7b
	Pets (Divergent)	Pets (Full)	UF-P-2	UF-P-4	UF-P-2
BTL [42]	63.27 ± 0.57	94.92 ± 0.00	49.84 ± 0.14	53.48 ± 0.03	47.17
DPL [47]	70.62 ± 1.13	95 ± 0.00	49.57 ± 0.42	52.92 ± 0.06	49.51
VPL (Ours)	$\mathbf{100 \pm 0.00}$	$\mathbf{100 \pm 0.00}$	$\mathbf{74.75 \pm 2.01}$	$\mathbf{61.49 \pm 0.03}$	**76.41**

5 Conclusion

In this work, we presented VPL, a technique for pluralistic alignment of preference-based RLHF models via variational inference. We show that VPL can capture diverse preferences, and can be used for steerable personalized model learning while capturing uncertainty and divergence in preferences. We discussed practical considerations for enabling VPL to scale up for LLMs and policy learning and showed results across simulated control problems and LLM-based RLHF, significantly outperforming current RLHF techniques.

References

[1] R. Akrour, M. Schoenauer, and M. Sebag. April: Active preference learning-based reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012, Bristol, UK, September 24-28, 2012. Proceedings, Part II 23*, pages 116–131. Springer, 2012.

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

[3] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan. Learning from physical human corrections, one feature at a time. In *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*, pages 141–149, 2018.

[4] P. BehnamGhader, V. Adlakha, M. Mosbach, D. Bahdanau, N. Chapados, and S. Reddy. LLM2Vec: Large language models are secretly powerful text encoders. *arXiv preprint*, 2024. URL `https://arxiv.org/abs/2404.05961`.

[5] K. Bhatia, A. Narayan, C. M. De Sa, and C. Ré. Tart: A plug-and-play transformer module for task-agnostic reasoning. *Advances in Neural Information Processing Systems*, 36:9751–9788, 2023.

[6] E. Biyik and D. Sadigh. Batch active preference-based learning of reward functions. In *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*, volume 87 of *Proceedings of Machine Learning Research*, pages 519–528. PMLR, 2018.

[7] E. Bıyık, M. Palan, N. C. Landolfi, D. P. Losey, and D. Sadigh. Asking easy questions: A user-friendly approach to active reward learning. *arXiv preprint arXiv:1910.04365*, 2019.

[8] E. Bıyık, D. P. Losey, M. Palan, N. C. Landolfi, G. Shevchuk, and D. Sadigh. Learning reward functions from diverse sources of human feedback: Optimally integrating demonstrations and preferences. *The International Journal of Robotics Research*, 41(1):45–67, 2022.

[9] D. Blei, A. Kucukelbir, and J. McAuliffe. Variational inference: A review for statisticians. *arXiv preprint arXiv:1601.00670*, 2016.

[10] A. Bobu, D. R. R. Scobee, J. F. Fisac, S. S. Sastry, and A. D. Dragan. LESS is more: Rethinking probabilistic models of human behavior. In T. Belpaeme, J. E. Young, H. Gunes, and L. D. Riek, editors, *HRI '20: ACM/IEEE International Conference on Human-Robot Interaction, Cambridge, United Kingdom, March 23-26, 2020*, pages 429–437. ACM, 2020. doi: 10.1145/3319502.3374811. URL `https://doi.org/10.1145/3319502.3374811`.

[11] R. Boldi, L. Ding, L. Spector, and S. Niekum. Pareto-optimal learning from preferences with hidden context. *ArXiv*, abs/2406.15599, 2024. URL `https://api.semanticscholar.org/CorpusID:270703216`.

[12] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324, 1952. doi: 10.2307/2334029.

[13] S. Chakraborty, J. Qiu, H. Yuan, A. Koppel, F. Huang, D. Manocha, A. S. Bedi, and M. Wang. Maxmin-rlhf: Towards equitable alignment of large language models with diverse human preferences. *arXiv preprint arXiv:2402.08925*, 2024.

[14] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. In *Neural Information Processing Systems (NeurIPS)*, 2017.

[15] G. Cideron, S. Girgin, A. Raichuk, O. Pietquin, O. Bachem, and L. Hussenot. vec2text with round-trip translations. *arXiv preprint arXiv:2209.06792*, 2022.

[16] V. Conitzer, R. Freedman, J. Heitzig, W. H. Holliday, B. M. Jacobs, N. Lambert, M. Mossé, E. Pacuit, S. Russell, H. Schoelkopf, et al. Social choice for ai alignment: Dealing with diverse human feedback. *arXiv preprint arXiv:2404.10271*, 2024.

[17] G. Cui, L. Yuan, N. Ding, G. Yao, W. Zhu, Y. Ni, G. Xie, Z. Liu, and M. Sun. Ultrafeedback: Boosting language models with high-quality feedback. *arXiv preprint arXiv:2310.01377*, 2023.

[18] J. Dai, X. Pan, R. Sun, J. Ji, X. Xu, M. Liu, Y. Wang, and Y. Yang. Safe rlhf: Safe reinforcement learning from human feedback. *arXiv preprint arXiv:2310.12773*, 2023.

[19] V. Dumoulin, D. D. Johnson, P. S. Castro, H. Larochelle, and Y. Dauphin. A density estimation perspective on learning from pairwise human preferences. *ArXiv*, abs/2311.14115, 2023. URL `https://api.semanticscholar.org/CorpusID:265445749`.

[20] K. Ethayarajh, W. Xu, N. Muennighoff, D. Jurafsky, and D. Kiela. Kto: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

[21] S. Feng, T. Sorensen, Y. Liu, J. R. Fisher, C. Y. Park, Y. Choi, and Y. Tsvetkov. Modular pluralism: Pluralistic alignment via multi-llm collaboration. *ArXiv*, abs/2406.15951, 2024. URL `https://api.semanticscholar.org/CorpusID:270703169`.

[22] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[23] J. Fürnkranz, E. Hüllermeier, W. Cheng, and S.-H. Park. Preference-based reinforcement learning: a formal framework and a policy iteration algorithm. *Machine learning*, 89:123–156, 2012.

[24] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In J. G. Dy and A. Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *Proceedings of Machine Learning Research*, pages 1856–1865. PMLR, 2018. URL `http://proceedings.mlr.press/v80/haarnoja18b.html`.

[25] M. Hessel, H. Soyer, L. Espeholt, W. Czarnecki, S. Schmitt, and H. Van Hasselt. Multi-task deep reinforcement learning with popart. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019.

[26] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

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

[28] J. Ji, M. Liu, J. Dai, X. Pan, C. Zhang, C. Bian, B. Chen, R. Sun, Y. Wang, and Y. Yang. Beavertails: Towards improved safety alignment of llm via a human-preference dataset. *Advances in Neural Information Processing Systems*, 36, 2024.

[29] K. Kang, E. Wallace, C. Tomlin, A. Kumar, and S. Levine. Unfamiliar finetuning examples control how language models hallucinate. *arXiv preprint arXiv:2403.05612*, 2024.

[30] C. Kim, J. Park, J. Shin, H. Lee, P. Abbeel, and K. Lee. Preference transformer: Modeling human preferences using transformers for RL. In *International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=Peot1SFDX0`.

[31] D. P. Kingma and M. Welling. Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[32] H. R. Kirk, B. Vidgen, P. Röttger, and S. A. Hale. Personalisation within bounds: A risk taxonomy and policy framework for the alignment of large language models with personalised feedback. *arXiv preprint arXiv:2303.05453*, 2023.

[33] H. R. Kirk, A. Whitefield, P. Röttger, A. Bean, K. Margatina, J. Ciro, R. Mosquera, M. Bartolo, A. Williams, H. He, 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*, 2024.

[34] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=68n2s9ZJWF8`.

[35] C. Laidlaw and A. Dragan. The boltzmann policy distribution: Accounting for systematic suboptimality in human models. *arXiv preprint arXiv:2204.10759*, 2022.

[36] C. Laidlaw and S. Russell. Uncertain decisions facilitate better preference learning. *Advances in Neural Information Processing Systems*, 34:15070–15083, 2021.

[37] X. Li, Z. C. Lipton, and L. Leqi. Personalized language modeling from personalized human feedback. *ArXiv*, abs/2402.05133, 2024. URL `https://api.semanticscholar.org/CorpusID:267547503`.

[38] M. Liu, M. Zhu, and W. Zhang. Goal-conditioned reinforcement learning: Problems and solutions. *arXiv preprint arXiv:2201.08299*, 2022.

[39] N. Muennighoff. Sgpt: Gpt sentence embeddings for semantic search. *arXiv preprint arXiv:2202.08904*, 2022.

[40] R. Munos, M. Valko, D. Calandriello, M. G. Azar, M. Rowland, Z. D. Guo, Y. Tang, M. Geist, T. Mesnard, A. Michi, et al. Nash learning from human feedback. *arXiv preprint arXiv:2312.00886*, 2023.

[41] V. Myers, E. Bıyık, and D. Sadigh. Active reward learning from online preferences, 2023.

[42] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. L. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, J. Schulman, J. Hilton, F. Kelton, L. Miller, M. Simens, A. Askell, P. Welinder, P. Christiano, J. Leike, and R. Lowe. Training language models to follow instructions with human feedback, 2022.

[43] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. *arXiv preprint*, 2019.

[44] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36, 2024.

[45] X. Ran, M. Xu, L. Mei, Q. Xu, and Q. Liu. Detecting out-of-distribution samples via variational auto-encoder with reliable uncertainty estimation. *Neural Networks*, 145:199–208, 2022.

[46] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL `http://arxiv.org/abs/1707.06347`.

[47] A. Siththaranjan, C. Laidlaw, and D. Hadfield-Menell. Distributional preference learning: Understanding and accounting for hidden context in rlhf. *arXiv preprint arXiv:2312.08358*, 2023.

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

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

[50] G. Swamy, C. Dann, R. Kidambi, Z. S. Wu, and A. Agarwal. A minimaximalist approach to reinforcement learning from human feedback. *arXiv preprint arXiv:2401.04056*, 2024.

[51] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*, 2023.

[52] Y. Wang, Q. Liu, and C. Jin. Is rlhf more difficult than standard rl? a theoretical perspective. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 76006–76032. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/efb9629755e598c4f261c44aeb6fde5e-Paper-Conference.pdf`.

[53] A. Wilson, A. Fern, and P. Tadepalli. A bayesian approach for policy learning from trajectory preference queries. *Advances in neural information processing systems*, 25, 2012.

[54] J. Wu, R. Antonova, A. Kan, M. Lepert, A. Zeng, S. Song, J. Bohg, S. Rusinkiewicz, and T. Funkhouser. Tidybot: Personalized robot assistance with large language models. *Autonomous Robots*, 2023.

[55] Z. Wu, Y. Hu, W. Shi, N. Dziri, A. Suhr, P. Ammanabrolu, N. A. Smith, M. Ostendorf, and H. Hajishirzi. Fine-grained human feedback gives better rewards for language model training. *ArXiv*, abs/2306.01693, 2023. URL `https://api.semanticscholar.org/CorpusID:259064099`.

[56] P. Xavi, U. Eric, A. Szot, M. D. Cote, R. Partsey, J. Yang, R. Desai, A. W. Clegg, M. Hlavac, T. Min, T. Gervet, V. Vondrus, V.-P. Berges, J. Turner, O. Maksymets, Z. Kira, M. Kalakrishnan, J. Malik, D. S. Chaplot, U. Jain, D. Batra, A. Rai, and R. Mottaghi. Habitat 3.0: A co-habitat for humans, avatars and robots, 2023.

[57] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.

[58] S. Zhao, J. Dang, and A. Grover. Group preference optimization: Few-shot alignment of large language models. *arXiv preprint arXiv:2310.11523*, 2023.

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

[60] D. M. Ziegler, N. Stiennon, J. Wu, T. B. Brown, A. Radford, D. Amodei, P. Christiano, and G. Irving. Fine-tuning language models from human preferences. *arXiv preprint arXiv:1909.08593*, 2019. URL `https://arxiv.org/abs/1909.08593`.

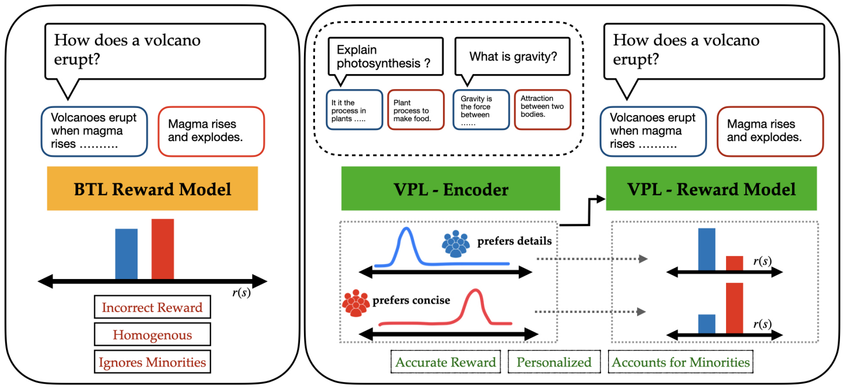


Figure 3: Current RLHF approaches [42] incorrectly assume an unimodal reward model for a diverse population of users. In this example, users have diverging preference over the length of the responses from a large language model. Without additional context, the BTL model considers both responses to be equally likely. In contrast, our method, VPL, is a personalized approach to RLHF. Using a few samples from a particular user, we infer the distribution over their distinct preferences. Based on this distribution, we condition the reward model to more accurately predict rewards, and enable steering the resulting policy to personalize to the specific user.

A Scaled Rewards for Multi-Task Learning

In practice, optimizing latent-conditioned reward functions learned with the VPL objective poses unique challenges. The pairwise preferences used to train the reward model in Section 3 do not have information about the scale of rewards, but only their relative ordering. As a simple illustration, if we have a pair of states s_A, s_B, where the users prefer s_A i.e. $s_A \succ s_B$, two different reward functions: $r(s_A) = 100, r(s_B) = 50$ or $r(s_A) = 50, r(s_B) = 0$ have the same likelihood under the BTL model. Empirically, we observe that this poses problems for optimizing Equation 2; different values of the latent variable z result in learned reward functions of vastly different scales. This is an issue for several reasons: 1) varying reward scales adversely affect the landscape of multi-user policy optimization (often observed in multi-task RL) [25], and 2) it is challenging to identify states where user preferences diverge across the population as differently scaled rewards cannot be directly compared.

To address this issue, we experiment with several different techniques for scaling the learned reward functions (see Appendix D.3). Our key insight in solving this challenge lies in the observation that while raw rewards from BTL are not scaled equally across z, probabilities from the preference likelihood model $p(y \mid s_A, s_B, z)$ are appropriately scaled. This suggests that an effective solution to the reward scaling issue is to replace the raw rewards from the BTL model ($r(s, z)$) with likelihoods suggested by the pairwise preference likelihood model $p(y \mid s_A, s_B, z)$. In particular, a natural choice of scaled rewards for a state s_A is the expected likelihood that the state s_A is "preferred" to all other states (or a sampled set of states) s_B observed in the data - $r_\phi(s_A, z) = \mathop{\mathbb{E}}_{s_B \in \mathcal{S}} [p_\phi(y = 1 \mid s_A, s_B, z)]$.

Since these are probabilities, normalized in the $[0, 1]$ range, the scaling of rewards is consistent across latents z. Note that these expected likelihood rewards can easily be obtained from the objective in Equation 2 since we already train a latent-conditional preference classifier via maximum likelihood. While proposed from a very different perspective, we note the similarity of this reward scaling approach to recent work [50, 40], in particular, Self-Play Preference Optimization (SPO) [50], which was originally proposed to address the issue of intransitive preferences. Similar to [50] we assume that the oracle / user providing preference labels is Non-Markovian. Due to this similarity[2], we use VPL-SPO to indicate this approach of preference likelihood-based reward scaling throughout our experiments (See Algorithm 3 for details).

[2]There are some differences in setup with SPO likelihoods being computed against on-policy samples, while VPL-SPO likelihoods are computed against a fixed offline dataset of comparator states.

B Active Learning

In VPL, the probabilistic modeling of the variational encoder naturally allows for active selection of the most informative query set based on maximal information gain, following prior work [6, 8, 41].Here the provision of preference labels $\{y^i\}_{i=1}^N$ will provide the maximum information about the latent distribution (and indirectly, the user preferences). This active query selection procedure can be expressed as the following optimization problem, maximizing the mutual information between the labels and the latent distribution.

$$\{(s_A^i, s_B^i)\}_{i=1}^N \leftarrow \underset{\{(s_A^i, s_B^i)\}_{i=1}^N}{\arg\max}\ \mathcal{I}\left(z; \{y^i\}_{i=1}^N \mid q_\psi, \{(s_A^i, s_B^i)\}_{i=1}^N\right) \tag{3}$$

The posterior q_ψ is a multivariate Gaussian, and assuming a uniform distribution over the set of labels, $q_\psi(z \mid \{(s_A^i, s_B^i)\}_{i=1}^N)$ allows for closed form solution for mutual information $\mathcal{I}$. To solve the maximization objective, we chose the query set $(s_A^i, s_B^i)_{i=1}^N$ with the maximum information gain, across samples from the preference dataset. Finally, we elicit user labels on this maximal query set, infer the latent, and condition the policy on this latent at deployment.

In our active inference technique, we use a sampling-based method inspired by [6] to generate the active queries for the model. Given a dataset of D queries $(s_A^i, s_B^i)_{i=1}^{|K|}$, we sample S query batches of size Q, where Q is number of annotations per batch we get from a user (total possible combinations are ${}^K C_N$). Here, $Q \in [2, 8]$, so we need to perform O(S * Q) passes over the model with batch size $2^Q \sim [4, 256]$. Furthermore, this process only needs to be performed once after the model is trained to obtain the most discriminative set of queries for the given model. Finally, whenever a new user interacts with the system, we need to get labels on the actively inferred queries (usually 2-4) but do not require any additional passes over the query dataset. In our experiments (Figure 4), we show that using active learning allows the model to achieve comparable performance with fewer queries (~ 2), as compared to randomly sampled larger (~ 8) queries.

B.1 Can VPL enable active query selection for latent estimation?

In Appendix B, we present an objective to actively query users at test time to efficiently infer user preferences. Figure 4 shows that this technique leads to better performance of the learned policy across varying numbers of queries $\|N\|$. This implies that the active learning objective 3 which maximizes information gain over the latent distribution generates queries that are more discriminative and provides a more informative posterior for user identification. This results in a more efficient adaptation of the downstream policy to the distinct user preferences, achieving the same performance with only half the queries. These methods can be potentially transferred to LLMs to query and identify user preferences with minimal questions.

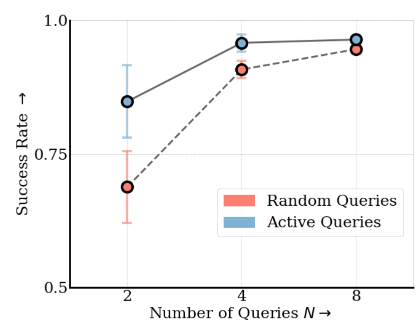


Figure 4: Active learning enables personalizing policies to user preferences with fewer queries.

C Scaling VPL for Reward Learning in Large Language Models (LLMs)

VPL can be used to train pluralistic reward models for LLMs, accounting for diverse human preferences and values. Here we discuss the key details that are essential to scale our method to LLMs. The architecture of our LLM reward model is shown in Figure 5. Unlike prior work which attempted to insert summary embedding layers into LLMs (see e.g. [15]), we find that we can successfully compress user preference information into a concise, probabilistic embedding vector z without sacrificing reward model performance. Further details and hyperparameters are discussed in Appendix E.

Prompt and Response Embeddings. Since using raw representations of the prompt and responses can increase the context length significantly, we use a pre-trained LLM to encode prompt and response pairs together [5] (to be consistent with previous notation, we assume a preferred state s^A contains both a prompt and response $[x, r]$, and we obtain $e^A = LLM(s^A)$). For efficient training, we pre-compute and freeze the encoded embeddings.

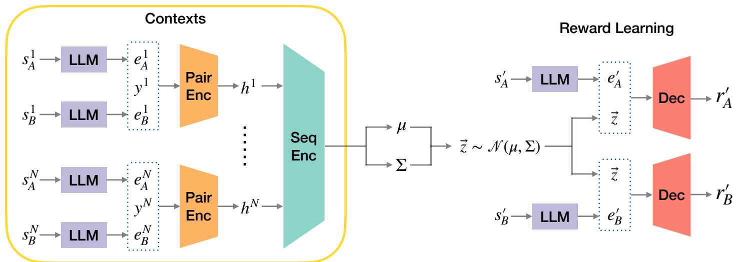


Figure 5: VPL LLM architecture for reward learning. The left and right parts denote the encoder q_ψ and the reward model $r(s, z)$ respectively.

Latent Encoder. Given a set of multiple encoded preference queries from the same user, $\{(e_i^A, e_i^B, y^i)\}_{i=1}^N$, we pass each through the same pair encoder to obtain $h_i = enc(e_i^A, e_i^B)$. The latent representation z is generated using a self-attention layer over the set of encoded pairs, $\{h_i\}_{i=1}^N$.

Reward learning. Here, the representation $e^{A'}$ of a new state $s^{A'}$ is concatenated with a z sampled from the posterior distribution which is passed into an MLP to predict the rewards. The LLM is fine-tuned using low-rank adaptation (LoRA) [26], and unlike typical RLHF settings, we find that we need to train the reward model for ≥ 1 epochs to fit the encoder and the reward model.

Data augmentation. As we scale VPL to larger datasets with more users, augmenting the training dataset with multiple context samples from the same user for each new data point is essential to learning an effective encoder. At training time, given a prompt and response pair s'_A, s'_B from a particular user, we generate $M = 8$ duplicates of this labeled response with different contexts, i.e., annotated prompt and response pairs $(\{(s_A^i, s_B^i, y^i)\}_{i=1}^N)_{j=1}^M$; where each context $(\{(s_A^i, s_B^i, y^i)\}_{i=1}^N)_j$ is sampled from a user annotated subset of size K ($K > N$).

D Experiments

D.1 Didactic example on a toy reward learning problem.

To more carefully understand the behavior of VPL empirically, we construct a didactic example [19] as shown in Figure 6. In this problem, let us consider a mixture of M different annotators providing preferences, where each annotator i has a reward function specified by $\mathcal{N}(\mu_i, \sigma_i)_{i=1}^M$ that they use to assign binary preferences. Mathematically, we sample the preferences from a mixture of Gaussians:

$$p(s_A \succ s_B \mid i) = \frac{e^{r_i(s_A)}}{e^{r_i(s_A)} + e^{r_i(s_B)}}; \text{ where } e^{r_i} \sim \frac{1}{\sigma_i\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu_i}{\sigma_i}\right)^2}$$

Multi-annotator preferences are simulated by sampling an annotator from this mixture distribution and then assigning binary preferences according to the chosen reward function. We train VPL as described in Section 3 to recover the underlying distribution over reward functions. As expected in Figure 6, standard RLHF [14] averages over the different modes since it can only represent a single reward function. While prior work accounts for hidden context in RLHF (DPL [47]) and learns uncertainty in the reward functions due to hidden context, it cannot accurately disambiguate different modes. Meanwhile, VPL is able to infer the underlying context using the approximate posterior q_ψ and the recover the individual reward modes through the latent-conditional reward function $r(s, z)$.

D.2 Does VPL scale with the number of diverse users?

In order to test the effectiveness of VPL in scaling to a control problem with larger modes of underlying preferences, we create a task with ten underlying locations that the users could prefer. So, the challenge here is to disambiguate the user preference among a larger space of possible goals and condition the policy to navigate successfully to the goal. Figure 7a shows that our method is able to navigate to the individual goals with a higher success rate, whereas the baseline DPL model [14]

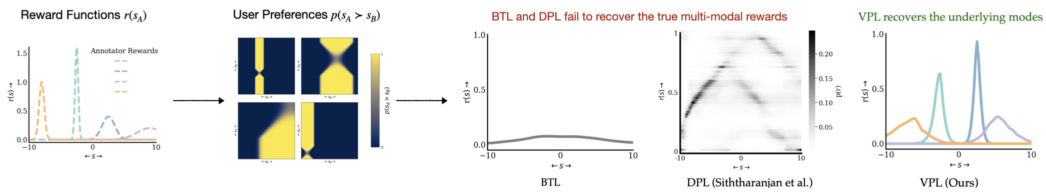
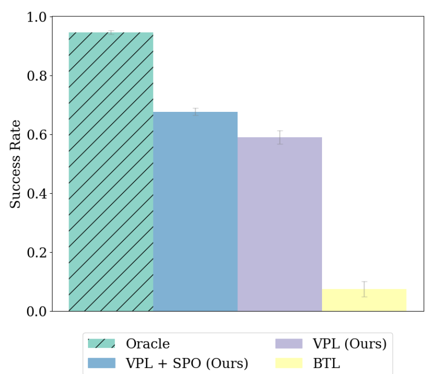
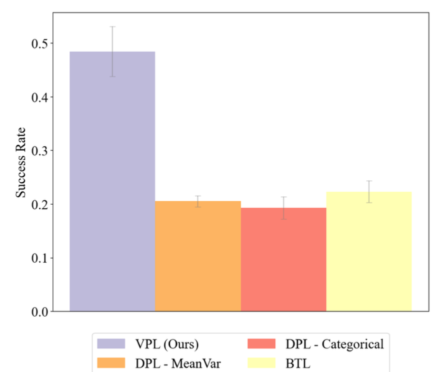


Figure 6: Didactic experiments comparing standard BTL [14], DPL [47] and VPL (Ours). Four Gaussian reward functions generate different binary preference data. The traditional BTL approach [14] averages the different modes, and DPL [47] captures the uncertainty in the rewards due to the multi-modality but cannot accurately predict the true modes. VPL (ours) infers the hidden latent as described in Section 3 and recovers the individual distribution of reward functions.



(a) VPL scales to Maze-Navigation task with ten modes of user preferences. BTL expectedly averages the modes and fails to learn. We also see the benefits of scaling rewards across this domain as well, where VPL +SPO performs better than VPL.



(b) We compare the performance of baselines and VPL on a Habitat-Rearrange environment with 100 users. VPL can scale to a much larger set of diverse users, complementing the real-world capabilities shown in Table 1 and Figure 2.

Figure 7: Comparison of VPL's scalability on different tasks and user bases.

collapses to a single user mode. This demonstrates the benefits of scaling VPL to a setting with a large population of diverse users. To test the method at a larger scale, we increase the number of users in the Habitat-Rearrange tasks to $\sim$100. It is a combinatorial problem as the users provide rankings over five locations, so all possible users/orderings are 5!. We observe in Figure 7b that VPL significantly outperforms the baselines in inferring the user preference and steering the robot policy.

D.3 Does scaling rewards help improve performance?

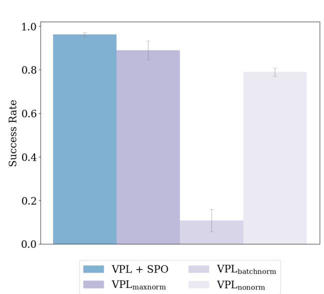


Figure 8: Comparing scaling methods on Maze-Navigation.

To avoid the problem of high variance rewards (Section 3), we compare the performance of VPL$_{\text{no-norm}}$ with VPL + SPO. We further compare against two normalizing schemes: VPL$_{batchnorm}$ where the rewards for each latent is normalized by the mean rewards across a set of state samples i.e. $r'(s, z) = \frac{r(s,z)}{\frac{1}{M}\sum_{s' \in \mathcal{S}} r(s',z)}$, and VPLmax-norm where all the rewards in the offline dataset are normalized by the maximum reward for any latent.

In Section A, we discuss the problem of generating scaled rewards from latent variable-based reward models and compare the performance across multiple baselines discussed above. As shown in Figure 8, the batch norm scaling generates highly biased estimates of the rewards, which is catastrophic for the method. However, VPL methods have decent performance at test-time, but are an unprincipled approach to the scaling problem. Our SPO + VPL presents a general method for estimating normalized rewards. Thus, in Figure 8 we can see that our method outperforms the baseline approaches in terms of success rate. The baselines have an unscaled or a biased estimate of the multi-modal rewards leading to sub-optimal performance. For the ravens-manipulation environment, the dataset doesn't contain sub-optimal trajectories, VPL (with max norm) performs comparably.

D.4 Does VPL scale to real-world settings with larger and noisy users?

A key assumption in our approach is that context questions accurately represent individual users without noise in the underlying dataset. To test VPL's robustness to noisy context labels at test time, we injected noise by flipping the questions answered by each user and evaluated the trained model's accuracy in predicting rewards. This experiment can help us evaluate how well the model would generalize to new users that have similar preferences to those experienced during training, but may not answer questions in exactly the same way. Figure 9 illustrates that VPL is able to outperform prior work even when 25% of preference labels are flipped at test time. Notably, we observed that longer context lengths result in more accurate reward modeling, even with higher noise injection. This is because the encoder can generate more accurate inferences of the latent distribution when provided with more user information through a larger number of context and response labels.

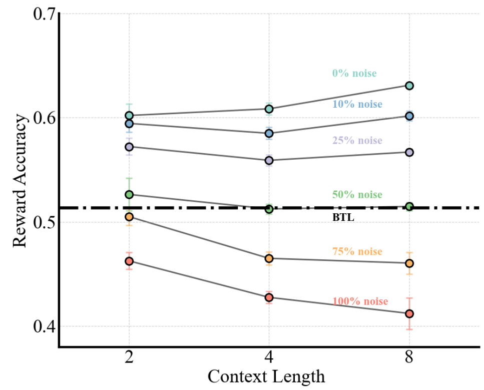


Figure 9: Reward accuracy with varying levels of noisy labels introduced at test time.

We note further that when 50% of the preference labels are flipped, the context preference queries provide no information about the user, and the performance of VPL is equivalent to the baseline BTL model. This mirrors the additional findings presented in Appendix D.7, which demonstrate that when VPL is trained on a unimodal preference language dataset, it gives equivalent performance to BTL. Essentially, when extra preference data is available for a user, VPL can personalize the reward model effectively and attain higher performance. But without that information, it performs just as well as the default BTL model. These findings, as shown in Table 1, Figure 9, and the Appendix, demonstrate VPL's effectiveness in handling multi-modality and noise in large-scale preference datasets, without compromising performance even when no additional preference information is available. This capability suggests the potential for VPL to contribute to the development of next-generation LLMs that are more personalized, inclusive, and efficient.

D.5 How does context length affect VPL?

In Habitat-TidyBot, the robot's task is to relocate an object in the kitchen (fork, knife, spoon, bowl, pitcher) to a specific location based on user preferences. The user can prefer to arrange objects according to the object function (i.e. kitchenware or tableware), or material (metal or plastic). The agent has to query the location of some of the objects, infer the user type, and arrange the requested object accordingly. Fig 2 shows that the baselines converge to the correct location for the objects agreed upon by the different users, but converge to one mode for the divergent choices. Meanwhile, VPL infers and adapts to user preferences and aligns with humans perfectly. However, one caveat to this is the context length i.e. the number of queries to each user. In Figure 10 we show that as the query length increases VPL can identify users with higher accuracy and achieve higher performance. This happens because certain queries are uninformative about the user preferences (such as things users agree on), and thus, it generates a high variance posterior. As a result, longer context length increases the probability

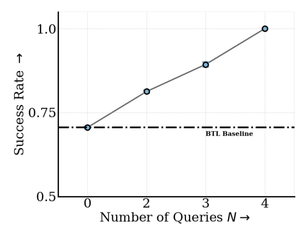


Figure 10: In the Habitat-TidyBot tasks, the agent has to relocate objects in the house based on the user's preferences. We show that the accuracy of user modeling and choosing the right location for the object increases with the number of queries the agent makes to infer the latent distribution.

of useful queries, which enables low variance posterior inference and improved alignment during decision-making. In Appendix B, we also show how to use an active learning approach to generate queries based on a max information gain objective. For LLMs, in Figure 9, we show that a higher context length also provides robustness to noise in the annotated queries.

D.6 Visualizing Learned Rewards and Embeddings

We visualize the rewards generated by the baseline and the latent conditioned rewards models on the diverse domains described in Section 4. We observe that VPL reconstructs the multi-modal reward functions, based on the inferred latent distribution. Figure 1 shows that the BTL models averages the reward over the user-preferred goals, while VPL accurately reconstructs the individual user-specific rewards. Figure 11 shows that for optimal trajectories solving the task, VPL can accurately match the ground rewards for the two modes. At the same time, DPL [47] predicts high-variance rewards for both cases due to the inherent multi-modality. Finally, in Figure 12 we see that the majority of users consider the desk to be the preferred location of the bowl, and standard BTL models converge to the majority population. Meanwhile, VPL can generate user-specific rewards, satisfying all the user groups.

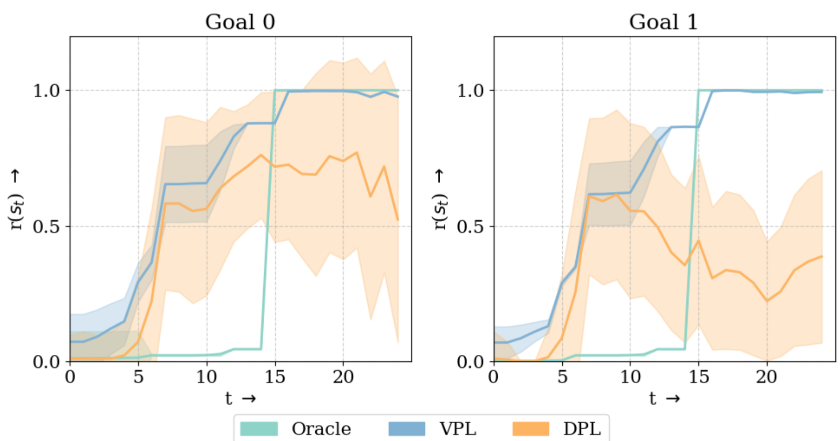
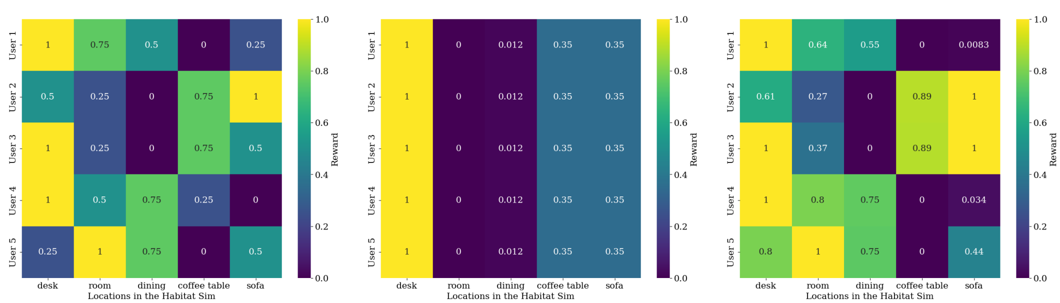


Figure 11: In the Ravens-Manipulation task, we compare the predicted rewards for states s_t along timesteps t in oracle trajectories to either of the goals the user prefers. VPL (Ours) can learn the individual reward functions for the two different (closely matching the ground truth rewards for both users) leading to more performant policies (see Figure 2), while DPL [47] learns a high variance reward function due to the multi-modality.



(a) Ground truth (b) BTL (c) Our method

Figure 12: In the Habitat-Rearrange task, annotators have rankings, i.e. preferences over the different locations they want the robot to place their bowl in their home. Accordingly, (a) shows the rewards associated with a particular location ("column") for each annotator ("row"). We see that a majority of the users rank the desk as the most preferred location. Consequently, in (b), unimodal BTL converges to this majority preference ignoring other users. However, in (c) we see that VPL accurately reconstructs diverse preferences and aligns to all five users.

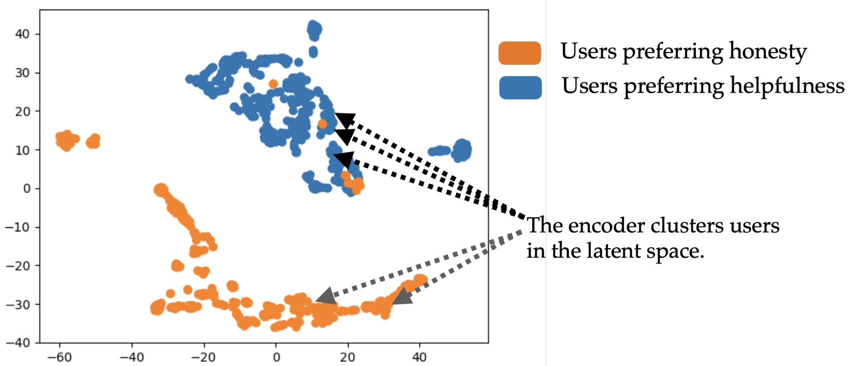


Figure 13: We train GPT2-based VPL, on the UltraFeedback-P dataset. In this plot, we visualize the T-SNE features of the latent distribution z produced by the encoder q_ψ on a set of annotated prompts and responses $\{s_A^i, s_B^i, y^i\}_{i=1}^N$ from the two users in the dataset. We see that the encoder clusters the users in the latent space, allowing the decoder to personalize the reward models according to multiple objectives preferred by the diverse users belonging to a cluster.

In Figure 13, we show that the VPL-Encoder effectively learns a latent space with clusters that correspond to the user types in the dataset. Previous work has shown that attempting to compress information within an LLM into a single bottleneck embedding layer can hinder performance [15]. However, using the architectural design of VPL as well as user-context data augmentation, VPL is able to learn a compressed user representation that accurately separates users according to their preferences, from only a few preference labels.

D.7 Does VPL under uni-modal settings?

To test that the introduction of a variational framework does not decrease performance in settings where all users have single/aligned preferences, we run an experiment on the UF-P-4 dataset, where we considered the preferences of a single user (preferring the model to be "honest" over all other attributes) to analyze the single-modal case as suggested. The standard BTL model gives a 77.04% eval accuracy while our VPL model gives a 77.16% eval accuracy. Our model matches the baseline performance, indicating no drop in performance when using VPL compared to traditional RLHF over an unimodal dataset.

D.8 Limitations and Social Impact

A key limitation of this work is that as yet, realistic preference datasets containing the opinions of diverse users do not yet exist at scale. This limitation necessitated creating our own synthetic preference datasets. Although this was also the approach taken in prior work on personalized RLHF (e.g. [47, 59]), an important direction for future work will be to apply VPL to more realistic preference data from diverse groups of users. Further, our current experiments on the UltraFeedback dataset assume that when adapting to a new user's preference, it is possible to ask them to provide preferences over a sample from a fixed set of survey questions. In the future, it would be good to relax this assumption so that VPL could be applied to preferences obtained naturally during a conversation with the user.

We believe VPL could also provide promising safety benefits, beyond modeling the preferences of diverse users. Because uncertainty detection can be used to prevent jailbreak attacks that arise from conflicting rewards [47], and since VPL could capture uncertainty in the distribution over user preferences, this could potentially be used to improve safety by having the model stop or refuse to answer when uncertainty cannot be reduced [29].

This work has a clear social impact when deployed on user-facing systems like LLMs or household robots. In pluralistic alignment, we assume that some differences in user preferences reflect divergent but equally valid perspectives on which moral or cultural values an LLM should align to; for example, individuals from one culture may hold collectivist moral values, while another culture is more individualist. Both value systems should be respected, and as such LLMs should be able to recognize and adapt to the values held by a particular user. However, the personalized model could potentially either be sycophantic or align with adversarial users, which is undesirable. This raises very interesting questions, such as: At what point should the LLM embrace a more universal set of values? How can we detect when such a point has occurred in a particular conversation? The probabilistic framework of the user distribution could allow us to identify low probability or rare behavior, and also the distributional nature of reward functions can help us point out responses where the users are divergent (maybe signifying disagreement). Additionally, a model could flexibly switch between adhering to the user's personal preferences and conforming to a more universal perspective on topics where it could be biased, or is sensitive to jailbreak [2]. Taking inspiration from Constitutional AI [2], we can allow a system designer to specify the topics for which the LLM should not engage in user personalization. Overall, this presents an exciting future research direction toward building safe personalized LLMs.

E Implementation Details

E.1 Training and Evaluation Details:

We test our hypothesis across evaluation domains in two steps: 1) We train a reward model on a dataset of preferences collected using diverse simulated humans; 2) We train a policy using RL to maximize the learned rewards. For these experiments, we use Implicit Q-Learning [34], an offline RL algorithm that achieves strong performance on offline RL benchmarks [22]. Using the learned reward function $r_\phi(s, z)$, and the prior $p(z)$, we label the reward-free offline RL dataset $D = (s_t, a_t, s_{t+1})$, by sampling a latent $z \in p(z)$, and setting the reward as $r_t = r_\phi(s_t)$, or a one-step look-ahead method, where $r_t = r_\phi(s_{t+1})$ (refer to Algorithm 2 for complete method). We include the hyperparameters and the training details in the Appendix E.

E.2 Baselines:

We compare our method against multiple baselines: 1) **Oracle [38]**: This is a goal-conditioned offline RL method, that presents an oracle with access to the true reward functions for all annotators. 2) **BTL [14]**: This is the standard RLHF method from [14, 42] as a baseline, where the reward model is approximated using the *unimodal BTL* function. 3) **DPL [47]**: Following the work on accounting for hidden context in RLHF, we train a distributional reward model, using both the mean-variance (MeanVar) and categorical (Categorical) approximation for the reward functions. 4) **VPL (Ours)**: We denote two versions of our method, **VPL** and **VPL + SPO**, discussed in Section 3.

E.3 Task Details

We evaluate our methods on three simulated control environments.

Maze-Navigation. This task is adapted from the "maze2d-medium-v2" environment from the D4RL benchmark [22]. The observation space is the position and velocity of the robot (p_x, p_y, p_z), and the pointmass is controlled using torque control. In this environment, point mass doesn't have access to a goal, and diverse users guide the agent to (two or ten) different locations in the maze, marked with their preferred colors. The users label the preferences over two states based on the shortest path to the goal from each state, i.e, the user prefers states closer to their preferred color location. The offline dataset for IQL is collected using the waypoint controllers provided in the D4RL benchmark. For each episode, the agent is spawned at a random location in the maze, interacts with a random user, and navigates to the goal based on the learned reward model and corresponding policy trained on the offline dataset. The oracle reward function is the optimal Q-value of the state, generated using a dynamic programming solution, which is available in D4RL.

Ravens-Manipulation This task is adapted from the ravens benchmark [57]. The observation space is the 3-D position of the object, the 3-D position of the end effector, and the grasp state of the object, i.e., $(ee_x, ee_y, ee_z, p_x, p_y, p_z, grasp)$. The agent commands absolute positions for the 3-DOF robot arm in end-effector space i.e. (ee'_x, ee'_y, ee'_z). This setup resembles how a robot arm would infer user preferences to organize a dining table. The users prefer two different locations for the box spawned at a random location at the beginning of each episode. To collect offline data, we use a motion planning oracle with some added noise, which tries to pick the box and place it randomly at one of the two locations. The oracle reward function is as follows:

$$\text{reward} = \frac{1}{100}\begin{cases} 100 & \text{if goal\_dist} < 0.05 \text{ and not grasped} \\ 5 & \text{if goal\_dist} < 0.05 \text{ and grasped} \\ 2 + \exp(-\text{goal\_dist}) & \text{if not goal\_dist} < 0.05 \text{ and grasped} \\ \exp(-\text{gripper\_dist}) & \text{if not goal\_dist} < 0.05 \text{ and not grasped} \end{cases}$$

Habitat-Rearrange This is a task based on the Meta Habitat simulator [56]. Here, the objective for the Mobile Manipulator is to pick a bowl and place it at the user's preferred location in the home. However, the exact location is underspecified and needs to be inferred from the user-annotated preferences. The robot uses a motion primitive to navigate and place the object at five possible locations ('desk', 'room', 'dining', 'coffee table', 'sofa'). This problem is reduced to a discrete one-step problem, where the robot has to reason about the best possible location to put the bowl. For ranking the states, the users are generated by randomly choosing a five random orderings of the locations, each corresponding to an individual user. At test time, the agent is greedy and chooses the location with the maximum inferred reward from the learned reward model.

Habitat-TidyBot This task is based on Meta Habitat [56] and inspired from the TidyBot task [54]. In this environment, there are 5 objects in the kitchen (spoon, knife, plate, bowl, and spatula). Each user has their preferences for sorting the objects according to particular attributes (material such as metal, plastic or function i.e. tableware or cooking ware). The robot observes or queries the user for the existing location of a subset of objects, then rearranges the misplaced object according to the inferred user preference and greedily selects the goal with the higher reward. The baseline here would converge to sort the objects according to one attribute only, while VPL would infer the latent, and choose the correct location for the given object.

E.4 LLM Preference Learning Dataset Descriptions

Prior RLHF works have focused mainly on unimodal BTL models, and as such there is a lack of publicly available datasets containing annotated preferences with divergent objectives. To evaluate our method on capturing multi-modality in preferences for LLMs, we consider two benchmarks. First, we introduce a synthetic dataset, *Pets*, that directly represents multimodal preferences, and second, we augment the publicly available UltraFeedback [17] dataset.

Pets Here, the dataset is generated to reflect multi-modal user preferences, where each user has a preference ranking over four kinds of animals (in this case cats, dogs, birds, and rabbits). To simulate a setting where users agree on some comparisons and disagree on others, we consider two users who agree on the best and worst pet and disagree on the middle pair of rankings over pets. Preferences here are divergent in certain cases (middle pets), and agree in other instances (best and worst pets), requiring multimodal preference modeling. We evaluate our approach on two versions of the dataset: Pets (Full), and Pets (Divergent) which contains only those prompt and response pairs where the users are divergent (i.e. they have conflicting preferences). For the contexts $\{(s_A^i, s_B^i, y^i)\}_{i=1}^N$, we randomly sample 1-4 other prompts and ranked responses from the same user.

This dataset is generated synthetically to specifically study the ability of models to perform divergent preference modeling. The goal here is to choose between different types of pets. For each animal, including bird, cat, dog, and rabbit, we use GPT-4 to generate 100 sentences that describe these kinds of pets. Then we define two user groups based on their preference order over the pets. So as to have mix of contexts where users agree and disagree, we construct a preference ordering where both groups like birds the most and rabbits the least. One group of users prefers dogs to cats while another group disagrees and prefers cats to dogs. That is to say, among all 6 comparisons between two kinds of pets, only one pair (dogs versus cats) leads to divergent opinions, while the users agree on other comparisons (birds better than dogs, dogs better than rabbits and so on). This tests the ability of the preference models to capture multimodality, even when the users do agree on some set of preferences.

We then construct the Pets dataset by clustering the prompt and ranked responses according to the group of preferences that they align with. To generate the Pets dataset, we randomly sample a pair of different pets as well as two corresponding sentences, and then label them to be “chosen” and “rejected” according to the preference of either “dog group” or “cat group”. The prompt is fixed to be “*Human: Please talk about one kind of pet.*” After all the chosen/rejected pairs are generated, we randomly sample 1-4 pairs of responses from the same user that are comparing dogs and cats for use as the context to the variational encoder. These are informative since a user’s choice over these contexts can clearly express the user’s preference group. In this way, we can generate a “full" Pets dataset, and based on that we filter out a “divergent" split that only contains controversial data points (comparing dogs and cats). This dataset is meant as a didactic test for language modeling capabilities, but scalability is further tested in the following section with the UltraFeedback-P dataset.

Here we show an example data point for Pets.

- Prompt: “Human: Please talk about one kind of pets.”
- Response A: “Cats communicate through vocalizations.” (Rejected)
- Response B: “Birds exhibit complex social behaviors within flocks.” (Chosen)
- Contexts: ["chosen": "Cats have a preference for certain types of litter.","rejected":"Dogs enjoy exploring their surroundings."], ["chosen":"Cats have a preferred scratching substrate.","rejected": "Dogs have a unique personality."]

UltraFeedback-P To construct this dataset *UF-P* (where P stands for personalized), we use the fine-grained scores over different attributes available in the UltraFeedback (UF) [17] dataset to construct different users, taking a similar approach to prior work [47].

The UF dataset contains 64,000 response pairs evaluated across four dimensions: helpfulness, honesty, instruction following, and truthfulness. Instead of using averaged scores, we focus on fine-grained scores to create a dataset with multi-modal and divergent user preferences.

We assume that each user ranks responses based on a single attribute. For instance, one user may prioritize helpfulness while another values honesty, leading to divergent labels for the same response pair. In line with prior work [2, 47], we first create *UF-P-2*, focusing on helpfulness and honesty as the two user preferences. To emphasize multi-modal reward modeling, we filter out response pairs where both users either agree or are indecisive, leaving approximately 4,000 prompt-response pairs per user. Finally, to generate the context $\{(s_A^i, s_B^i, y^i)\}_{i=1}^N$ for inferring latent distributions, for each prompt and response pair, we sample N different data points from a smaller subset of size K from the dataset ($K = 100$ for GPT2 and 16 for Llama2). For a deployed LLM system, this is analogous to having a known set of survey questions from which the user must answer a subset of 2-8 questions in order to personalize the model’s behavior to their needs.

To evaluate VPL ’s ability to handle more users, we introduce *UF-P-4*, which includes all four attributes as distinct user preferences. We filter out response pairs where all four users agree, and within each user subset, we remove pairs where the user gives equal ratings to both responses. This results in approximately 7,500 prompt-response pairs per user. The context generation follows the same approach as UF-P-2, with the number of context samples randomly selected between 2 and 8. However, in UF-P-4 the context can still contain queries where at least two users overlap. Thus, this provides a dataset to evaluate VPL in cases in cases where different users agree on some responses, but not all of them. Overall, this creates a diverse and challenging benchmark for pluralistic alignment, constructed from open-source datasets [17].

E.5 Implementation Details

Learned Prior. While the methods described in the methods section only learn the reward model and the latent encoder, we can also use a prior $p(z)$ as is common in variational inference methods. We assume that our prior is a multi-variate Gaussian with mean μ and covariance $\Sigma = \text{diag}(\sigma\sigma^T)$, where $\mu, \sigma \in \mathrm{R}^d$; where d is the dimension of the latent embedding. In all experiments, they are initialized from a standard Gaussian. However, in our control experiments, we observed that using a learned Gaussian i.e. setting μ and σ to learnable parameters under the ELBO objective in Eq. 2 improved performance and stability during training.

LLM Embeddings. In our experiments, we use the embedding from the last token as our encoding for the prompt+response input. However, we also tried llm2vec [4] and a weighted pooling mechanism [39]. However, we find that using the last token embedding as inputs to the encoder and for predicting the rewards performs the best.

E.6 Hyperparameters

Table 2: Hyperparameters for learning reward models using VPL. We sweep over these values and report the best results on 5 seeds.

Hyperparameter	**Value**
Encoder and Decoder Architecture	MLP
Hidden layers	2 layers of width 256
Optimizer	Adam (Kingma & Ba, 2015)
Learning rate	3.0000×10^{-4}
Latent dimension	$\{8, 16, 32\}$
β	Cosine annealing between 0 and 1 every 25% steps
VAE Prior	Multi-variate gaussian with learnable parameters $\mu, \sigma \in \mathcal{R}^d$
Context set of queries $\|N\|$	2,4,8,16
Comparison set size (for SPO + VPL)	1000
Number of annotated sets	5000 (Maze), 10000(Ravens)

Table 3: Hyperparameters for IQL. We use the same parameters across all experiments.

Hyperparameter	**Value**
Architecture	MLP
Hidden layers	2 layers of width 256 (4 layers of width 1024 for users > 5)
Optimizer	Adam (Kingma & Ba, 2015)
Learning rate	3.0000×10^{-4}
Discount	0.99
Expectile	0.9
Temperature	10
Dataset size	4M steps (Navigation), 5K trajectories (Manipulation)

Table 4: Hyperparameters for LLM experiments

Hyperparameter	**Value**
Pair Encoder Architecture	2 layer MLP with LeakyReLU
Hidden Dimension	512 (GPT2), 1024 (Llama2-7b)
Latent Dimension	512 (GPT2), 1024 (Llama2-7b)
Learning rate	1.0000×10^{-4}
Learning rate scheduler	Cosine with 3% warmup steps
Context set size N	8
Full context sampling set K	100 (GPT2), 16 (Llama2-7b)
Batch size	32 (GPT2), 512 (Llama)
Optimizer	AdamW(with weight decay = 0.001)
β	0.0001 (for Pets), 0.0 (for UltraFeedback-P)
Computational Resources	$2 \times$ RTX4090, $4 \times$ A100

F Algorithms

Algorithm 1 Learning Multimodal Reward Functions using VPL

Require: Preference Data $\{(s_A^i, s_B^i, y^i)\}_{i=1}^N$
Require: Encoder E, Reward Model R, prior $p(z)$
while not done **do**
  Sample a batch $B \sim D$
  Compute $\mu_B, \sigma_B = \mathrm{E}(B)$
  Sample $z \sim \mathcal{N}(\mu_B, \sigma_B)$
  Append z to B: $\{(s_A, s_B, y)\} \rightarrow \{((s_A|z, s_B|z), y)\}$
  Compute rewards: $r_{s_A} = \mathrm{R}(s_A|z)$ and $r_{s_B} = \mathrm{R}(s_B|z)$
  Compute reconstruction loss: $\mathcal{L}_{\text{recon}} = \text{cross entropy}(\sigma(r_{s_A} - r_{s_B}), y)$
  Compute KL-loss: $\mathcal{L}_{KL} = \beta \cdot D_{KL}(\mathcal{N}(\mu_B, \sum_B) \parallel p(z))$
  Compute total loss: $\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{KL}$
  Update E and R by optimizing $\mathcal{L}_{\text{total}}$
end while =0

Algorithm 2 Policy Optimization using IQL and VPL

Require: Offline Dataset $\{\tau_1, \tau_2, \dots\}$
Require: Reward Model $r_\phi(s, z)$
Require: Prior $p(z)$
Require: Policy $\pi(a|s, z)$
for each trajectory $\tau_i = \{(s_t, a_t, s_{t+1})\}_{t=1}^T$ in D **do**
  Sample $z \sim p(z)$
  for each state s_t in τ_i **do**
    Compute reward $r_t = r_\phi(s_t, z)$ # Alternatively, $r_t = r_\phi(s_{t+1}, z)$
    Update dataset with (s_t, r_t, z)
  end for
end for
Train policy $\pi(a|s, z)$ using IQL =0

Algorithm 3 Policy Optimization using IQL and SPO + VPL (Note the changes from Algorithm 2)

Require: Offline Dataset $\{\tau_1, \tau_2, \dots\}$
Require: Preference Model $p_\phi(s_A, s_B, z)$
Require: Prior $p(z)$
Require: Policy $\pi(a|s, z)$
Require: Comparison set $C = \{s_1, s_2, \dots, s_C\}$ # Sampled randomly from the offline dataset
for each trajectory $\tau_i = \{(s_t, a_t, s_{t+1})\}_{t=1}^T$ in D **do**
  Sample $z \sim p(z)$
  for each state s_t in τ_i **do**
    Compute reward $r_t = \frac{1}{\|C\|} \sum_{s' \in C} p_\phi(s_t, s', z)$
    Update dataset with (s_t, r_t, z)
  end for
end for
Train policy $\pi(a|s, z)$ using IQL =0

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We highlight the claims and contributions in the abstract and the introduction.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We highlight the limitations of our approach in the Conclusion.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA] .

Justification: We do not claim any substantial theoretical results that requires a proof.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We describe all the algorithms, datasets, models and hyperparameters in detail in the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We do not provide immediate access to the data and code, but will do so in the future.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide all details in the main paper and the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report standard error across multiple seeds for all results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We indicate the compute resources used in the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We confirm to the code of ethics in every respect.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss the impact of the work in the concluding section.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: We use publicly available datasets and models, and the synthetic toy dataset generated has no risk for misuse.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We cite all the datasets, models and other assets used in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We introduce a new dataset, and provide detailed explanation of the dataset in the Appendix.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: We dont use real humans or crowd sourcing as subjects for any of our experiments.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Our experiments do not require an IRB.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.