Listwise Reward Estimation for Offline Preference-based Reinforcement Learning

Heewoong Choi [1] **Sangwon Jung** [1] **Hongjoon Ahn** [1] **Taesup Moon** [1 2]

Abstract

In Reinforcement Learning (RL), designing precise reward functions remains to be a challenge, particularly when aligning with human intent. Preference-based RL (PbRL) was introduced to address this problem by learning reward models from human feedback. However, existing PbRL methods have limitations as they often overlook the *second-order* preference that indicates the relative strength of preference. In this paper, we propose Listwise Reward Estimation (LiRE), a novel approach for offline PbRL that leverages second-order preference information by constructing a Ranked List of Trajectories (RLT), which can be efficiently built by using the same ternary feedback type as traditional methods. To validate the effectiveness of LiRE, we propose a new offline PbRL dataset that objectively reflects the effect of the estimated rewards. Our extensive experiments on the dataset demonstrate the superiority of LiRE, *i.e.,* outperforming state-of-the-art baselines even with modest feedback budgets and enjoying robustness with respect to the number of feedbacks and feedback noise. Our code is available at https://github.com/chwoong/LiRE

1. Introduction

Reinforcement Learning (RL) has demonstrated considerable success in various domains such as robotics (Haarnoja et al., 2018; Kalashnikov et al., 2018), game (Silver et al., 2017; Mnih et al., 2013; Vinyals et al., 2019), autonomous driving (Kiran et al., 2021), and real-world tasks (Tan et al., 2018; Chebotar et al., 2019). An essential component of RL is to define suitable and precise reward functions so that an RL agent can be trained successfully (Wirth et al., 2017). However, designing the reward function is time-consuming, especially if we want to align it with human intent (Hejna & Sadigh, 2024).

This shortcoming has led to research on learning the reward model from human feedback without explicitly designing the reward function. While expert demonstration is one type of human feedback (Abbeel & Ng, 2004), recent papers use preference feedback on which of a pair of trajectory segments is preferred since it is a significantly easier type of feedback to collect (Kaufmann et al., 2023; Casper et al., 2023). More specifically, the common approach for the Preference-based RL (PbRL) consists of two steps: (1) learn a reward model using preference feedback from trajectory segment pairs, then (2) apply ordinary RL algorithms with the learned reward model. After successfully training a robot agent with PbRL (Christiano et al., 2017), it was shown that novel behaviors aligned with human intent, *e.g.,* backflips, can also be learned (Lee et al., 2021b), while learning such behavior would be extremely hard from explicitly hand-coded rewards. The PbRL framework has gained popularity in both online (Park et al., 2021; Liang et al., 2021) and offline (Kim et al., 2022; Shin et al., 2022; An et al., 2023; Hejna & Sadigh, 2024) settings, in which the former allows the agents to interact with their environments, while the latter does not.

In this paper, we focus on the offline PbRL setting, in which the goal is to find an optimal policy solely from the *previously collected* preference feedbacks on the pairs of trajectories obtained from some past, fixed policy. This setting is challenging since the preference feedback cannot be actively collected on the trajectories generated by the current, updated policy. Hence, developing effective methods for collecting maximally informative preference feedback data from the past policy as well as devising efficient reward learning schemes is indispensable.

The current norm is to collect ternary preference feedback (*i.e.,* more/less/equally preferred) for *independently* sampled pairs of trajectories, and then employ the standard Bradley-Terry (BT) model (Bradley & Terry, 1952) on the collected data to learn the reward function. While the above approach was shown to be effective to some extent, a critical limitation also exists. Namely, due to the independent sampling of the

[1]Department of Electrical and Computer Engineering, Seoul National University [2]ASRI/INMC/IPAI/AIIS, Seoul National University. Correspondence to: Taesup Moon <tsmoon@snu.ac.kr>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

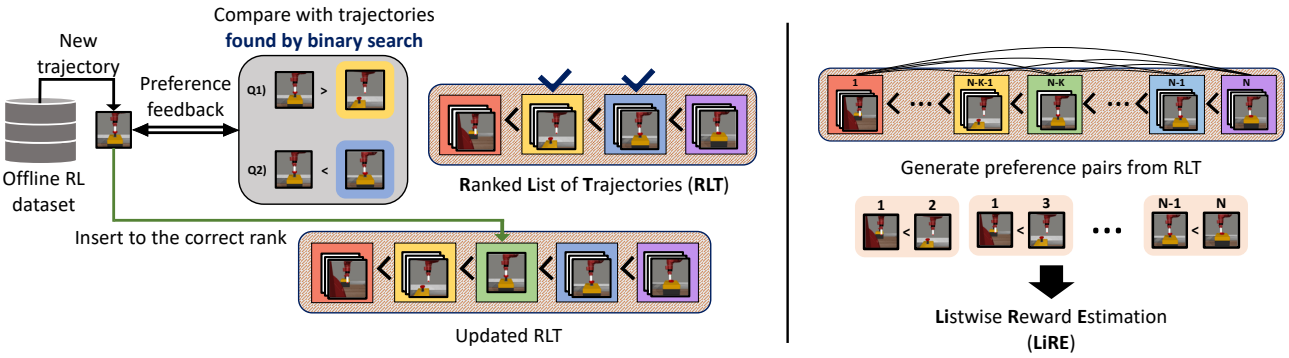


Figure 1: **An overview of LiRE.** The figure shows an example of a *button-press-topdown* task. We sample a trajectory segment and sequentially obtain the preference feedback for existing trajectories in RLT. We use binary search to find the correct rank (left) efficiently. Multiple preference pairs are generated from RLT to learn the reward model (right).

pairs of trajectories and simple ternary feedback, the *second-order* preference, which stands for the *relative* strengths of the preferences, cannot be utilized. There exists a long line of work in several areas asserting that utilizing such second-order preference is indeed effective for more accurate learning (Xia et al., 2008; Touvron et al., 2023; Hwang et al., 2023; Song et al., 2024). However, the majority of these works presume the availability of more sophisticated preference feedback types, which are considerably more laborious and expensive to obtain than the above-mentioned ternary feedback.

To that end, we propose to construct a Ranked List of Trajectories (RLT) while collecting preference feedback data to exploit the second-order preference when learning a reward function. The key novelty and strength of our method is to use *exactly the same* feedback type and budget as before and not require any additional sophistication in collecting the preference feedback. As outlined in Figure 1, the main idea of building such a ranked list is to sample *a* trajectory and *sequentially* obtain the preference feedback by comparing it with existing trajectories in the ranked list multiple times to find its correct rank in the list. Hence, our method ends up sampling *fewer* trajectories for a fixed feedback budget compared to the conventional independent pair sampling. However, once the complete RLT is built, the second-order preference can be extracted and exploited for estimating the reward function, which, as we show in our experiments, results in a significant performance boost of offline PbRL.

The superiority of our method, dubbed as **LiRE** (**Li**stwise **R**eward **E**stimation), is demonstrated through extensive experimental validation. We first created an offline RL dataset using Meta-World (Yu et al., 2020) and DeepMind Control Suite (DMControl) (Tassa et al., 2018) environments that can objectively compare the reward estimation quality of offline PbRL methods. This is motivated by (Li et al., 2023), which pointed out that the offline RL performance can be high in some popular benchmark datasets even with *wrong* or *constant* reward functions. On our proposed datasets, we show that many tasks *cannot* be properly learned with existing offline PbRL methods, even with a large preference feedback budget. In contrast, we showcase our LiRE can outperform those baselines on most of the tasks with significant margins even with a modest preference feedback budget. We conduct comprehensive experimental analyses to investigate the impact of several factors, including the score function of the BT model, the number of preference feedbacks, and the number of trajectories in the RLT. The experimental results show that the degree to which second-order information is utilized has a significant positive impact on the performance of offline PbRL. Furthermore, the results of the real human preference feedback experiments, along with experiments on the level of preference feedback noise and feedback granularity, demonstrate the effectiveness of LiRE in practical scenarios. These analyses provide substantial evidence supporting the strength and robustness of LiRE.

2. Related Works

2.1. Offline Preference-based RL

Due to the difficulty of defining rewards in reinforcement learning (Sutton & Barto, 2018; McKinney et al., 2023), PbRL uses comparison information between trajectories to learn a reward function (Christiano et al., 2017; Fürnkranz et al., 2012; Wilson et al., 2012; Akrour et al., 2012; Ouyang et al., 2022; Stiennon et al., 2020). However, the human preference feedback required for PbRL is expensive to obtain. Thus, several PbRL approaches have been devised to reduce the number of expensive human feedbacks, such as using additional expert demonstrations (Ibarz et al., 2018), meta-learning (Hejna III & Sadigh, 2023), semi-supervised learning or data augmentation (Park et al., 2021), unsupervised pre-training (Lee et al., 2021b), exploration based on reward uncertainty (Liang et al., 2021), and using sequen-

tial preference ranking (Hwang et al., 2023). Offline PbRL assumes a more challenging problem setting where agents cannot interact with the environment, unlike online PbRL where preference feedback can be obtained while interacting with the environment.

In offline PbRL, the two kinds of data are provided: offline data obtained from an unknown policy and preference feedbacks on the pairs of trajectories. Also, traditional offline PbRL methods have two phases; they train a reward model using the preference feedback and then perform RL with the trained reward model without interacting with the environment. On the other hand, recent works propose performing offline PbRL without the reward model by directly optimizing policies (An et al., 2023; Kang et al., 2023), or learning state-action value function or regret from preference labels (Hejna & Sadigh, 2024; Hejna et al., 2023). However, due to the constraint of no interaction with the environment, obtaining the most informative preference feedback from the offline dataset is as important as developing a new training method without the reward model or designing the structure of the reward model well (*e.g.,* non-Markovian reward modeling (Kim et al., 2022)). An active query selection method has been proposed to obtain informative preference pairs (Shin et al., 2022), but their method did not attempt to obtain second-order preference.

Most offline PbRL papers have validated their algorithms on the D4RL dataset (Fu et al., 2020). However, it has been shown that typical offline RL algorithms can produce good policies on D4RL even with a completely wrong reward (*e.g.,* zero, random, negative reward) due to the pessimism and survival instinct of offline RL algorithms (Shin et al., 2022; Li et al., 2023). Hence, to properly evaluate how well offline PbRL algorithms learn the reward model, we need to validate them on a new dataset, on which the policy cannot be easily learned due to survival instincts.

2.2. Second-order Preference Feedback

While typical approaches in PbRL only focus on the first-order preference (*i.e.,* ternary labels including bad, equal, and good), several approaches in the NLP and RL domains have recently been proposed to utilize second-order preference about the relative difference between preferences. One approach is to directly obtain a relative rating for each trajectory pair (*e.g.,* significantly better or slightly better) or an absolute rating for each trajectory (*e.g.,* very good or good) (Touvron et al., 2023; Cao et al., 2021; White et al., 2023). However, the more granular the preferences are, the more expensive they are than just ternary labels.

There is a rich Learning-to-Rank literature that learns the ranking given second-order preference feedback in the form of absolute ratings (Burges et al., 2005; Xia et al., 2008; Xu & Li, 2007; Swezey et al., 2021), but they do not address how to obtain second-order preference only with ternary labels. Another approach is to obtain the second-order preference between samples from a fully-ranked list for multiple trajectories (Chen et al., 2022; Palan et al., 2019; Zhu et al., 2023; Song et al., 2024; Myers et al., 2022; Bıyık et al., 2019; Brown et al., 2019). However, they do not address how to efficiently obtain the fully-ranked list in terms of the number of feedbacks. Since naively constructing a fully-ranked list would require a large number of feedbacks that increase quadratically with the number of trajectories, developing a more efficient list construction method is crucial.

Accordingly, some recent studies have developed how to obtain partially-ranked lists that only know the rankings among a few trajectories (Zhao et al., 2023; Hwang et al., 2023). Perhaps, one of the closest research to ours is Sequential Preference Ranking (SeqRank) (Hwang et al., 2023), which sequentially collects the preference feedback between a newly observed segment and a previously collected segment. However, since their method builds partially-ranked lists rather than fully-ranked lists, the short length of the lists limits the ability to fully utilize second-order information.

3. Preliminaries

An RL algorithm considers a Markov decision process (MDP) and aims to find the optimum policy that maximizes the cumulative discounted rewards. MDP is defined by a tuple (S, A, P, r, γ) where S, A are state, action space, $P = P(\cdot|s,a)$ is the environment transition dynamics, $r = r(s,a)$ is reward function, and γ is discount factor. In offline PbRL, we assume that we do not know the true reward r, but we have a pre-collected dataset that is a set of tuples, $D_o := \{(s,a,s')|(s,a) \sim \mu, s' \sim P(\cdot|s,a)\}$. In general, the policy μ from which the data was collected is unknown. We are allowed to ask for preference feedbacks to obtain preference labels for two distinct trajectory segments sampled from $D_s := \{\sigma \,|\, \sigma = (s_0, a_0, s_1, a_1, \cdots, s_{T-1}, a_{T-1}), (s_t, a_t, s_{t+1}) \in D_o\}$. Annotators assign a ternary label l given a pair of segments $\sigma_1, \sigma_2 \in D_s$; $l = 0$ indicates that σ_1 is preferred over σ_2 (*i.e.,* $\sigma_1 \succ \sigma_2$), $l = 1$ indicates the opposite preference (*i.e.,* $\sigma_1 \prec \sigma_2$), and $l = 0.5$ indicates that σ_1 and σ_2 are equally preferred (*i.e.,* $\sigma_1 = \sigma_2$).

The goal of acquiring preference labels is to learn the unknown reward function. Conventional offline PbRL methods use a preference model that defines the probability that one segment is better than the other as

$$P_\theta(\sigma_1 \succ \sigma_2) = \frac{\phi\big(r_\theta(\sigma_1)\big)}{\phi\big(r_\theta(\sigma_1)\big) + \phi\big(r_\theta(\sigma_2)\big)} \tag{1}$$

in which $r_\theta(\sigma_i) = \sum_{(s_t,a_t)\in\sigma_i} r_\theta(s_t, a_t)$ and θ is the parameter of the reward model. The score function $\phi(x) = \exp(x)$ is commonly used in the BT model (Bradley & Terry,

1952). Given the trajectory segment preference dataset, $D_{pref} \coloneqq \{(\sigma_{i_1}, \sigma_{i_2}, l_i)\}_{i=1}^{K}$, the parameter θ is learned by minimizing following cross-entropy loss:

$$L(\theta) = - \mathop{\mathbb{E}}_{\substack{(\sigma_{i_1},\sigma_{i_2},l_i) \\ \in D_{pref}}} \Big[(1 - l_i) \log P_\theta \left(\sigma_{i_1} \succ \sigma_{i_2}\right) + l_i \log P_\theta \left(\sigma_{i_1} \prec \sigma_{i_2}\right) \Big]. \quad (2)$$

4. LiRE: Listwise Reward Estimation

As mentioned in Section 1, the conventional offline PbRL approaches cannot utilize the second-order information of the preference feedback. In order to describe our method, we begin by stating the mild assumptions we make.

Assumption 4.1. (Completeness) For any two segments σ_i, σ_j, the human feedbacks are provided in the following three ways, $\sigma_i \succ \sigma_j$ or $\sigma_i \prec \sigma_j$ or $\sigma_i = \sigma_j$.
(Transitivity) For any three segments σ_i, σ_j, and σ_k, if $\sigma_i = \sigma_j$ and $\sigma_j = \sigma_k$, then $\sigma_i = \sigma_k$. Also, if $\sigma_i \succ \sigma_j$ and $\sigma_j \succ \sigma_k$, then $\sigma_i \succ \sigma_k$.
Remarks: These assumptions are a generalization of SeqRank (Hwang et al., 2023) to include equal labels. While the transitivity assumption may not always hold in practice, we demonstrate that our method is robust both in the presence of feedback noise (Section 5.4.3) and in real human experiments (Section 5.5), even when the transitivity assumption may not hold.

4.1. Constructing a Ranked List of Trajectories (RLT)

Our goal is to obtain an RLT in which the segments σ are ordered by their level of preference. We represent RLT, L, in the following form:

$$L = [g_1 \prec g_2 \prec \cdots \prec g_s],$$

in which $g_i = \{\sigma_{i_1}, \cdots, \sigma_{i_k}\}$ is a *group* of segments with the same preference level and s is the number of groups in the list. Namely, if $m > n$, we note any segment $\sigma_i \in g_m$ is preferred over any segment $\sigma_j \in g_n$.

Since we assume to have exactly the same type of ternary feedback defined in Section 3, we cannot build RLT by obtaining the listwise feedback at once. Hence, we construct by sequentially obtaining the labels as we describe below.

We start with an initial list $[\{\sigma_1\}]$ by selecting a random segment σ_1 from D_s. We then repeat the process of sequentially sampling the new segment $\sigma_2, \sigma_3, \cdots \in D_s$ and placing it in the appropriate position in the list until the feedback budget limit is reached. To place a newly sampled σ_i in the RLT, we compare it with a segment $\sigma_k \in g_m$ for some group g_m in the list and obtain the ternary preference feedback. Depending on the feedback, we proceed as follows:

Table 1: Feedback efficiency and sample diversity of independent pairwise sampling, SeqRank, and RLT.

	Independent	SeqRank	RLT
Feedback efficiency	1	1.392	$\mathcal{O}(M/\log M)$
Sample diversity	2	$\mathcal{O}(1)$	$\mathcal{O}(1/\log M)$

- If $\sigma_i = \sigma_k$, add σ_i to the group g_m.
- If $\sigma_i \prec \sigma_k$, find the position within $g_1, \cdots, g_{m-1}$.
- If $\sigma_i \succ \sigma_k$, find the position within $g_{m+1}, \cdots, g_s$.

For the latter two cases, we use a binary search so that we can recursively find the correct group for each segment. Namely, the RLT construction algorithm is based on a *binary insertion sort* and the pseudocode is summarized in Algorithm 1 (Appendix). We note that while we can also adopt merge sort or quick sort to construct an RLT after collecting multiple segments, if we already have a partially constructed RLT, binary insertion sort would be more feedback-efficient.

Feedback efficiency and sample diversity Note that by design, we need to obtain *multiple* preference feedbacks for each new segment σ_i. Therefore, for a fixed feedback budget, our method samples fewer segments. However, from the constructed RLT, we can generate *many* preference pairs by exploiting the second-order information encoded in the list; namely, σ_i is preferred to *all* the segments in the groups that rank lower than the group that σ_i belongs to.

To that end, we analyze the feedback efficiency and sample diversity of RLT. **Feedback efficiency** is defined in SeqRank (Hwang et al., 2023) as the ratio of the number of total preference pairs generated to the number of preference feedbacks obtained. We also define **sample diversity** as the ratio of the total number of sampled segments to the number of preference feedbacks obtained. Suppose we obtain preference feedbacks until we collect a total of M segments in the preference dataset. Constructing an RLT with M segments requires $\mathcal{O}(M \log M)$ feedbacks because we use an efficient sorting method based on binary search. In this case, the number of all possible preference pairs (including ties) that can be generated from the RLT is $\binom{M}{2}$. Table 1 summarizes the feedback efficiency and sample diversity of independent pairwise sampling, SeqRank, and RLT. Note our method has a faster rate of increase in the feedback efficiency even with diminishing sample diversity as the number of segments M in RLT increases.

Constructing multiple RLTs Algorithm 1 places all the segments in a single ranked list. Instead of constructing one long list, we devise a variant that generates multiple lists by setting a limit (Q) on the feedback budget for each list. The reason for generating multiple lists is that as the length of the list increases, the number of preference feedbacks required by the binary search process increases. Hence, we increase the sample diversity within the total feedback budget by generating multiple RLTs.

4.2. Listwise Reward Estimation from RLT

Once the RLT is constructed, we construct the preference dataset, $D_l = \{(\sigma_{i_1}, \sigma_{i_2}, l_i)\}_{i=1}^{K}$ with all the pairs we can obtain from the RLT. Specifically, when $\sigma_{i_1} \in g_m$ and $\sigma_{i_2} \in g_n$, the preference label l_i is as follows: $l_i = 0.5$ if $m = n$, $l_i = 0$ if $m > n$, and $l_i = 1$ if $m < n$. The key difference from traditional pairwise PbRL methods is that, instead of independently sampling segment pairs, we derive preference pairs from the RLT. To compare with the independent sampling, suppose that the RLT has segments with the relationship, $\sigma_a < \sigma_b < \sigma_c$. If we sample all pairs from the RLT, then $(\sigma_a, \sigma_b, 1), (\sigma_b, \sigma_c, 1), (\sigma_a, \sigma_c, 1) \in D_l$. From these preference pairs, it can be inferred that the degree to which σ_c is preferred over σ_a is stronger than the degree to which σ_c is preferred over σ_b. Consequently, the reward model trained with pairwise loss in (2) can learn second-order preference between each pair of segments. In contrast, the reward model learned from independent sampling cannot learn second-order preference because each segment is not compared to multiple other segments.

We use pairwise loss in our main experiments, but we can also train the reward model with listwise loss since the segments are ranked in the RLT. To train the reward model with listwise loss, we assume the segments follow a Plackett-Luce model (Plackett, 1975) which defines the probability distribution of objects in a ranked list. We discuss listwise loss more in detail in Appendix A.3 — but, our experimental results show that training with pairwise loss performs better than listwise loss in most cases.

Our proposed LiRE trains the reward model with linear score function $\phi(x) = x$ in (1). The choice of linear score function has the same effect as setting the reward to be the exponent of the optimal reward value obtained through training with an exponential score function $\phi(x) = \exp(x)$. Therefore, the linear score function amplifies the difference in reward values, particularly in regions with high reward values, compared to the exponential score function.

Bounding reward model If $\phi(x) = \exp(x)$, then *adding* a constant to the reward function $\hat{r}_\theta$ does not affect the resulting probability distribution. To align the scaling of the learned $\hat{r}_\theta$ in ensemble reward models, a common choice for the reward model is using the Tanh activation, *i.e.,* $\hat{r}_\theta(\sigma) = \sum_t \hat{r}_\theta(s_t, a_t) = \sum_t \tanh(f_\theta(s_t, a_t))$ (Lee et al., 2021b; Hejna & Sadigh, 2024), to bound the output of the reward model.

In the case of $\phi(x) = x$, *scaling* the reward function by a constant does not affect the probability distribution. Similarly, we use the same Tanh activation function for $\phi(x) = x$ to bound the output of the reward model. Specifically, we set $\hat{r}_\theta(\sigma) = \sum_t \big(1 + \tanh(f_\theta(s_t, a_t))\big) > 0$ to ensure that the probability defined in (1) is positive.

5. Experimental Results

5.1. Settings

Dataset Previous offline PbRL papers are evaluated mainly on D4RL (Fu et al., 2020), but D4RL has the problem that RL performance can be high even when wrong rewards are used (Li et al., 2023; Shin et al., 2022). To that end, we newly collect the offline PbRL dataset with Meta-World (Yu et al., 2020) and DeepMind Control Suite (DMControl) (Tassa et al., 2018) following the protocols of previous work: `medium-replay` dataset, *e.g.,* (Yu et al., 2021a; Mazoure et al., 2023; Gulcehre et al., 2020) and `medium-expert` dataset, *e.g.,* (Yu et al., 2021b; Sinha et al., 2022; Hejna & Sadigh, 2024; Li et al., 2023).

The `medium-replay` dataset collects data from replay buffers used in online RL algorithms, such as the SAC (Haarnoja et al., 2018), and the `medium-expert` dataset collects trajectories generated by the noisy perturbed expert policy. We experiment on both datasets while our main analyses are done on `medium-replay`; see Appendix C.2 for complete details on constructing them. The prior works (Shin et al., 2022; Zhang, 2023) have created datasets that consider survival instinct. However, their dataset was evaluated with only 100 or fewer preference feedbacks, whereas we use 500, 1000, or more feedbacks.

Baselines In our experiments, we consider five baselines: Markovian Reward (MR), Preference Transformer (PT) (Kim et al., 2022), Offline Preference-based Reward Learning (OPRL) (Shin et al., 2022), Inverse Preference Learning (IPL) (Hejna & Sadigh, 2024), and Direct Preference-based Policy Optimization (DPPO) (An et al., 2023). MR refers to the method trained with the MLP layer with the Markovian reward assumption, which is the baseline model used in PT. OPRL learns multiple reward models to select the query actively with the highest preference disagreement. Lastly, IPL and DPPO are algorithms that learn policies without the reward model.

All the above five baselines belong to pairwise PbRL because they all train based on the BT model given first-order preference feedbacks sampled as independent pairs. In addition to pairwise PbRL, we also compare with the sequential pairwise comparison method proposed by SeqRank (Hwang et al., 2023).

Implementation details For LiRE, we use the linear score function and set $Q = 100$ as the default feedback budget for each list. Therefore, if the total number of feedbacks is 500, then five RLTs will be constructed. All baseline methods, including ours, can be applied to any offline RL algorithm, but, as in previous works, we use IQL (Kostrikov et al., 2021). The hyperparameters for each algorithm and the criteria for the equally preferred label threshold of scripted teacher can be found in the Appendix C.4.

Table 2: Average success rates on `medium-replay` dataset over six random seeds. We use 500 and 1000 preference feedbacks and report the average performance of the last five trained policies. The yellow and gray shading represent the best and second-best performances, respectively.

# of feedbacks	Algorithm	button-press -topdown	box-close	dial-turn	sweep	button-press -topdown-wall	sweep-into	drawer-open	lever-pull
-	IQL with GT rewards	88.33 ± 4.76	93.40 ± 3.10	75.40 ± 5.47	98.33 ± 1.87	56.27 ± 6.32	78.80 ± 7.96	100.00 ± 0.00	98.47 ± 1.77
500	MR	9.60 ± 5.74	10.33 ± 8.23	50.20 ± 8.51	79.80 ± 13.36	0.13 ± 0.50	24.80 ± 5.28	98.07 ± 3.20	50.53 ± 8.55
	PT (Kim et al., 2022)	22.87 ± 9.06	0.33 ± 1.16	**68.67** ± 12.39	43.07 ± 24.57	0.87 ± 1.43	20.53 ± 8.26	88.73 ± 11.64	82.40 ± 22.69
	OPRL (Shin et al., 2022)	12.13 ± 5.75	4.73 ± 3.24	54.33 ± 11.47	**94.13** ± 5.95	0.20 ± 0.60	25.87 ± 8.58	94.13 ± 6.41	54.67 ± 12.79
	DPPO (An et al., 2023)	3.93 ± 4.34	10.20 ± 11.47	26.67 ± 22.23	10.47 ± 15.84	0.80 ± 1.51	23.07 ± 7.02	35.93 ± 11.18	10.13 ± 12.19
	IPL (Hejna & Sadigh, 2024)	**34.73** ± 13.92	5.93 ± 5.81	31.53 ± 12.50	27.20 ± 23.81	**8.93** ± 9.84	**32.20** ± 7.35	19.00 ± 13.63	31.20 ± 15.76
	SeqRank (Hwang et al., 2023)	17.6 ± 11.94	**13.2** ± 12.72	65.6 ± 12.84	**83.4** ± 9.76	1.73 ± 1.98	25.67 ± 11.02	**99.53** ± 0.36	**95.67** ± 4.04
	LiRE (ours)	**67.20** ± 18.97	**51.53** ± 18.48	**79.07** ± 10.96	77.53 ± 10.50	**79.13** ± 15.19	**49.13** ± 15.85	**99.40** ± 1.65	**95.67** ± 6.26
1000	MR	9.27 ± 5.30	17.07 ± 9.56	59.07 ± 7.57	**90.80** ± 9.74	0.60 ± 1.87	26.07 ± 8.57	96.47 ± 4.02	50.87 ± 10.89
	PT (Kim et al., 2022)	18.27 ± 10.62	2.27 ± 2.86	68.80 ± 5.50	29.13 ± 14.55	2.13 ± 2.96	20.27 ± 7.84	95.40 ± 7.27	72.93 ± 10.16
	OPRL (Shin et al., 2022)	11.00 ± 7.84	15.07 ± 11.19	51.33 ± 10.08	**85.53** ± 5.43	0.33 ± 0.75	28.27 ± 6.40	**99.20** ± 1.42	53.20 ± 6.67
	DPPO (An et al., 2023)	3.20 ± 3.04	9.33 ± 9.60	36.40 ± 21.95	8.73 ± 16.37	0.27 ± 0.85	23.33 ± 7.80	36.47 ± 7.30	8.53 ± 9.96
	IPL (Hejna & Sadigh, 2024)	**36.67** ± 17.40	6.73 ± 8.41	43.93 ± 13.37	38.33 ± 24.87	**14.07** ± 11.47	**30.40** ± 7.74	28.53 ± 18.37	40.40 ± 17.38
	SeqRank (Hwang et al., 2023)	13.93 ± 8.11	**46.60** ±12.53	**70.67** ± 7.58	74.93 ±22.35	2.47 ±2.67	29.33 ± 11.59	98.6 ± 3.92	**95.47** ± 3.86
	LiRE (ours)	**83.07** ± 6.38	**89.13** ± 6.02	**76.93** ± 7.55	75.87 ± 6.81	**81.47** ± 10.04	**57.73** ± 13.11	**99.73** ± 0.85	**99.47** ± 1.15

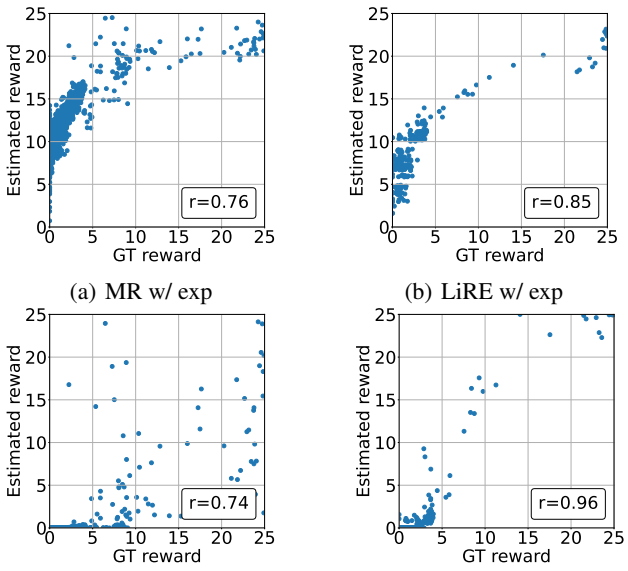
Table 3: Average success rates on `medium-replay` with 500 preference feedbacks.

RLT	$\phi(x)$	button-press -topdown	box-close	dial-turn	lever-pull
✗	$\exp(x)$	9.60 ± 5.74	10.33 ± 8.23	50.20 ± 8.51	50.53 ± 8.55
✓	$\exp(x)$	12.87 ± 7.86	22.73 ± 10.40	65.87 ± 9.46	57.87 ± 11.28
✗	x	36.87 ± 13.75	11.27 ± 14.91	77.27 ± 11.90	70.20 ± 18.03
✓	x	**67.20** ± 18.97	**51.53** ± 18.48	**79.07** ± 10.96	**95.67** ± 6.26

5.2. Evaluation on the Offline PbRL Benchmark

We compare LiRE with the baselines mainly on the Meta-World `medium-replay` dataset. Table 2 summarizes the results of offline RL performance using ground-truth (GT) rewards and preference feedbacks respectively. For many tasks, such as *button-press-topdown* and *box-close*, MR performs poorly compared to training with GT rewards, even with 1000 preference feedbacks. The problem of poor performance remains even if we replace the reward model with a more complex transformer architecture, PT. PT improves performance in *dial-turn* and *lever-pull* tasks, but for other tasks, the performance worsens. OPRL generally performs better than MR due to the increased consistency of the reward models, but the performance improvement is small. Lastly, DPPO and IPL perform better than MR on only a few tasks. We note that existing offline PbRL methods are rarely better than MR when validated on our new dataset.

In contrast, LiRE shows a significant performance improvement over MR except for the *sweep* task. We demonstrate the importance of RLT and the linear score function by achieving high performance even when compared to SeqRank, which is also not an independent pairwise method. In addition, policies trained with preference data outperform policies trained with GT rewards on the *button-press-topdown-wall* task, suggesting that reward models trained with preference data may be more effective, as also reported in prior works (Christiano et al., 2017; Kim et al., 2022; An et al., 2023). The results of the Meta-World `medium-expert` dataset and full learning curves are shown in the Appendix A.



(a) MR w/ exp (b) LiRE w/ exp (c) MR w/ linear (d) LiRE w/ linear

Figure 2: Scatter plots of the estimated rewards for the segments used for *box-close* task. The reward models are trained with MR or LiRE using the exp or linear score function. The Pearson correlation coefficient, r, is presented.

5.3. Ablation Studies of LiRE

5.3.1. Factors of Performance Improvement

We conduct an ablation study to verify if the performance improvement of LiRE is due to two factors: the linear score function and the RLT construction. Table 3 demonstrates that using the linear score function (bottom two rows) clearly

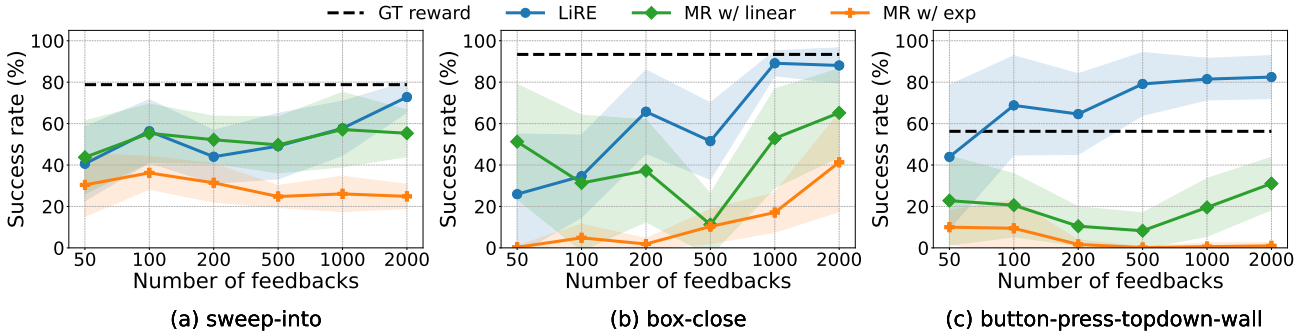


Figure 3: Average success rates of each method while varying the number of preference feedbacks. The black dotted line represents the average success rates when trained with GT reward.

Table 4: Average success rates of LiRE when adjusting the Q budget. We use a total of 500 preference feedbacks.

Dataset	Q=1 (MR w/ linear)	Q=2	Q=10	Q=20	Q=50	Q=100	Q=500	SeqRank w/ linear
button-press-topdown	36.87 ± 13.75	59.47 ± 2.18	53.60 ± 10.82	65.13 ± 14.24	**71.26** ± 12.95	67.20 ± 18.97	**77.67** ± 18.13	54.87 ± 9.89
lever-pull	70.20 ±18.03	69.80 ± 3.79	70.47 ±18.19	92.7 ±7.16	93.4 ±7.90	**95.67** ± 6.26	**99.33** ± 1.18	74.33 ± 18.50

brings a significant performance improvement compared to the default exponential score function (first two rows). Additionally, the bottom two rows of Table 3 show that constructing RLT improves performance by constructing the preference list and exploiting the second-order information. In particular, using the linear score function with RLT has a synergistic effect, resulting in even greater improvement.

5.3.2. Effect of RLT and score function on reward estimation

We examine the estimated reward values of the learned reward models. Figure 2 scatter plots the estimated rewards (y-axis), learned with 500 preference feedbacks, of the segments in *box-close* task against the GT rewards (x-axis). Note our LiRE uses fewer segments to train the reward model, so Figure 2(b) contains fewer dots than Figure 2(a). Each segment has a length of 25 and both GT and the estimated rewards are normalized to values between $[0, 25]$.

From the figure, we clearly observe that the estimated rewards in Figure 2(b) are more highly correlated than those in Figure 2(a). Namely, by constructing the RLT, LiRE exploits the second-order preference, and the high and low reward segments are more clearly distinguished by the reward estimates than vanilla MR. Additionally, when training the reward model with the linear score function, there is a larger gap in the estimated rewards within the reward region for higher GT rewards, as shown in Figure 2(d). We speculate that using the linear score function and RLT makes the estimated reward discern the optimal and suboptimal segments (with respect to the GT rewards) more clearly, hence, the policy learned with the estimated reward turns out to perform much better.

5.4. Additional Analyses of LiRE

5.4.1. Varying the number of feedbacks

We evaluate how the performances of the offline PbRL algorithms are affected by the number of feedbacks. Namely, we measure the average success rate of the *sweep-into*, *box-close*, and *button-press-topdown-wall* tasks of the `medium-replay` dataset while varying the number of the preference feedbacks from 50 to 2000. We note that most previous works (Kim et al., 2022; Hejna & Sadigh, 2024; An et al., 2023) using D4RL only use up to 500 preference feedbacks. As shown in Figure 3, we observe that the typical baseline, MR with exponential score function (denoted as MR w/ exp), cannot achieve a success rate higher than 50% for all three tasks even with 2000 preference feedbacks. When we instead use the linear score function, we observe that MR w/ linear performs much better than MR w/ exp, but the success rates sometimes still remain to be low (*e.g.,* *box-close* with 500 feedbacks and *button-press-topdown-wall* for most of the times). In contrast, it is evident that LiRE mostly surpasses the two baselines with large margins, even with fewer number of preference feedbacks. Specifically, for the *button-press-topdown-wall* task, LiRE with only 100 feedbacks outperforms not only the baselines with 2000 feedbacks but also the policy learned using the GT reward. Again, we can confirm that the high feedback efficiency enabled by RLT makes LiRE very effective even with a smaller number of feedbacks.

5.4.2. Varying Q budget

In Section 4.1, we described that multiple RLTs can be constructed by putting the budget limit (Q) in order to increase the sample diversity. In this subsection, we show the effect of Q. Table 4 shows the performance change of LiRE

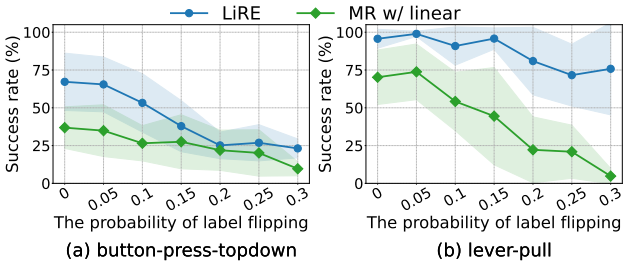


Figure 4: Robustness of LiRE w.r.t the feedback noise.

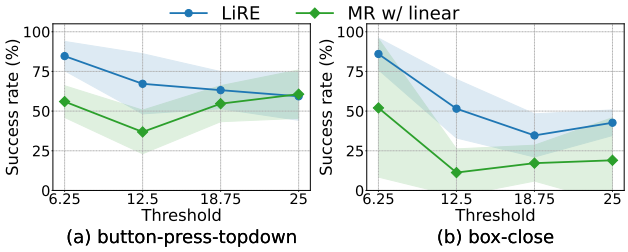


Figure 5: Effect of the granularity of preference feedback.

Table 5: Comparison results between SeqRank and LiRE on Meta-World `medium-replay` dataset.

# of feedbacks	Algorithm	Avg success rate	# of groups in the list	Feedback efficiency	Sample diversity
500	SeqRank w/ linear	61.84 ± 15.80	2.3 ± 0.09	2.20 ± 0.78	**1.002**
	LiRE	**74.83** ± 12.23	**9.3** ± 1.83	**11.33** ± 3.39	0.474 ± 0.069
1000	SeqRank w/ linear	67.49 ± 13.56	2.3 ± 0.09	2.18 ± 0.75	**1.001**
	LiRE	**82.92** ± 6.48	**9.3** ± 1.84	**11.33** ± 3.28	0.474 ± 0.067

when varying the Q budget to $1, 2, 10, 20, 50, 100$, and 500 while setting the total number of preference feedbacks to 500. Hence, for example, $Q = 100$ results in five lists, and $Q = 500$ results in a single list. Table 4 also shows the result of SeqRank (with linear score function). From the table, we observe that since the utilization of the second-order information increases with higher values of Q, the offline PbRL performance correspondingly improves, as expected. We also note that the performance of SeqRank is similar to that of LiRE with $Q = 2$ since SeqRank creates approximately 2.3 groups in the ranked list, as detailed in Table 5. This result indicates that SeqRank does not fully utilize second-order preference due to only building partially-ranked lists. A more in-depth comparison with SeqRank is given in Section 5.4.5.

5.4.3. Robustness to feedback noise

If the preference feedback used in PbRL models human preference labeling, it would be reasonable to assume that the preference feedback may be noisy. To that end, we experiment to assess the robustness of the offline PbRL performance of LiRE with respect to the preference feedback noise. We assume that the preference feedback can be noisy with probability p (*i.e.,* if $l_i = 0$ or 1, the label is flipped to $1 - l_i$ with probability p, and for tie labels, we flip to $l_i = 0$ or 1 with probability $p/2$, respectively). We varied the noise probability p from 0 to 0.3, and Figure 4 compares the success rates of MR w/ linear and LiRE. From the figure, we confirm that the performance of LiRE does not drop as severely as MR w/ linear when p increases. In particular, for *lever-pull* task, we observe that LiRE with feedback noise of $p = 0.3$ even results in a higher success rate than MR w/ linear with no noise, highlighting the robustness of LiRE with respect to feedback noise.

5.4.4. Impact of feedback granularity

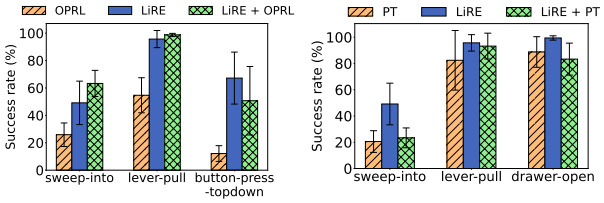
In Figure 5, we compare the performance of LiRE based on the threshold that determines the *tie* between the segments. Specifically, we adjust the threshold value for the reward difference that indicates whether two segments are *equally preferred*. Namely, a higher threshold value means that more segment pairs are labeled as equally preferred, resulting in less granular preference feedback. We note that the threshold value used in Table 2 is 12.5 (see Appendix C.4 for details). Figure 5 shows that using a smaller threshold (*i.e.,* more granular feedback) improves the performance of LiRE, while the performance becomes similar to that of MR w/ linear (*e.g., button-press-topdown* task with threshold 25) when the threshold increases. Thus, we confirm that the more granular preference labels generate additional second-order preference information, which would positively affect the performance of LiRE.

5.4.5. Comparison with SeqRank

Here, we compare LiRE with SeqRank, which also utilizes partially-ranked lists. We also employed the linear score function for SeqRank since it gave better results than using the exponential function and led to a fair comparison with LiRE. We evaluate the average success rates of SeqRank and LiRE on the Meta-World `medium-replay` dataset. The experimental results in Table 5 show that LiRE clearly achieves higher performance than SeqRank. We argue that SeqRank does not fully utilize the second-order information because SeqRank does not construct a fully-ranked list. Indeed, the third column of Table 5 shows that the number of groups in the ranked lists averages less than 3 with the SeqRank, whereas it increases to about 9 on average with LiRE. The last two columns of Table 5 compare feedback efficiency and sample diversity. LiRE achieves a sample diversity of approximately 0.47 through the use of binary search, and the feedback efficiency increases significantly to 11.33 by constructing RLT. Additionally, Table 6 shows the superiority of LiRE over SeqRank on the DM-Control `medium-replay` dataset. We note SeqRank also performs similarly to MR w/ linear on *walker-walk* and *humanoid-walk* tasks, while LiRE achieves much higher performance gains on all three tasks. Thus, we confirm that

Table 6: Average episode returns of MR, SeqRank, and LiRE on DMControl `medium-replay` dataset.

Algorithm	hopper-hop	walker-walk	humanoid-walk
IQL with GT rewards	157.95 ± 9.64	839.6 ± 36.57	250.9 ± 11.62
MR w/ linear	53.96 ± 24.42	677.38 ± 88.14	84.35 ± 23.23
SeqRank w/ linear	80.84 ± 27.67	698.81 ± 91.71	80.68 ± 14.67
LiRE	**99.14** ± 12.28	**822.27** ± 50.83	**104.08** ± 17.45



(a) LiRE with OPRL (b) LiRE with PT

Figure 6: Combining LiRE with other baselines.

constructing RLT and leveraging second-order preference is effective for locomotion as well as manipulation tasks.

5.4.6. Compatibility with other methods

To check the compatibility of LiRE with other methods, we tested the performance of LiRE when combined with OPRL and PT, respectively. First, to apply OPRL and LiRE simultaneously, we trained a reward model each time an RLT was newly constructed, and then actively sampled based on the disagreement of the reward models (following the method of OPRL) when constructing the next RLT. In Figure 6(a), we observe that LiRE+OPRL outperforms LiRE in *sweep-into* and *lever-pull* tasks but performs worse in *button-press-topdown* task. This discrepancy suggests that while the OPRL method enhances the consistency of the reward model, it may lead to oversampling similar segments that are challenging to distinguish depending on the task. Second, as shown in Figure 6(b), LiRE does not necessarily gain improvements when combined with PT. That is, since PT was originally designed to capture temporal dependencies of segments in reward modeling, it seems to struggle in accurately capturing the second-order preference information from RLT possibly due to overfitting to the sequence of past segments.

5.5. Human Experiments

Table 7 presents the results with *real* human preference feedback on the new *button-press-topdown* offline RL dataset, which is distinct from the dataset used in Table 2. Namely, we collected 200 preference feedbacks from one of the authors for each of the three feedback collection methods: MR, SeqRank, and LiRE. For LiRE, we used the feedback budget of $Q = 100$, resulting in two RLTs. The results again indicate that LiRE dominates other baselines and gets stronger when the linear score function is used. We believe this result shows the potential of LiRE that it can be very effective in practical scenarios with real human preference feedback, as in LLM alignment.

Table 7: Average success rates of a *button-press-topdown* task with 200 of real human feedbacks.

$\phi(x)$	MR	SeqRank	LiRE
$\exp(x)$	38.00 ± 8.85	40.93 ± 10.72	**78.27** ± 6.59
x	43.33 ± 25.72	74.13 ± 9.96	**90.67** ± 7.57

6. Limitation

We believe there are two limitations of LiRE. First, LiRE lacks the ability to parallelize the construction of RLT since there are dependencies between the order in which feedbacks are obtained to construct a fully-ranked list. Therefore, in scenarios where parallel feedback collection is feasible, constructing an RLT could be more time-consuming compared to collecting preference feedbacks independently in pairs. Nevertheless, the results presented in Appendix A.2 show that LiRE with only 200 feedbacks outperforms the independent pairwise sampling method using 1000 feedbacks, suggesting the importance of constructing RLT. Second, LiRE relies on the transitivity assumption outlined in Assumption 4.1. Although our experiments with feedback noise indicate LiRE's robustness to noise that violates this assumption, transitivity violations can occur even with noiseless labels in real-world applications. This issue is not unique to LiRE but affects other preference-based RL methods as well. Addressing transitivity violation remains a challenge for scalar reward models, so future research could explore solutions by using multi-dimensional preference feedback to construct RLTs for each dimension.

7. Concluding Remarks

In this paper, we propose a novel Listwise Reward Estimation (LiRE) method for offline preference-based RL. While obtaining second-order preference from a traditional framework is challenging, we demonstrate that LiRE efficiently exploits second-order preference by constructing an RLT using ordinary, simple ternary feedback. Our experiments demonstrate the significant performance gains achieved by LiRE on our new offline PbRL dataset, specifically designed to objectively reflect the effect of estimated rewards. Notably, the reward model trained with LiRE outperforms traditional pairwise feedback methods, even with fewer preference feedbacks, highlighting the importance of second-order preference information. Moreover, our findings suggest that constructing ranked lists can be straightforward without complex second-order preference feedback, indicating the broad applicability of LiRE to more challenging tasks and real-world applications.

Impact Statement

We believe our LiRE can be potentially applied to aligning the RL agent with more fine-grained human intent and preference. Such applications can bring significant societal consequences by enhancing the precision and effectiveness of AI systems in various fields such as health care and education. By ensuring that AI systems more closely reflect and respond to the detailed intentions of their users, LiRE has the potential to foster trust and acceptance of AI technologies, ultimately contributing to their more widespread and ethical adoption.

Acknowledgments

This work was supported in part by the National Research Foundation of Korea (NRF) grant [No.2021R1A2C2007884] and by Institute of Information & communications Technology Planning & Evaluation (IITP) grants [RS-2021-II211343, RS-2021-II212068, RS-2022-II220113, RS-2022-II220959] funded by the Korean government (MSIT). It was also supported by AOARD Grant No. FA2386-23-1-4079.

References

Abbeel, P. and Ng, A. Y. Apprenticeship Learning via Inverse Reinforcement Learning. In *International Conference on Machine Learning (ICML)*, 2004.

Akrour, R., Schoenauer, M., and Sebag, M. APRIL: Active Preference-learning based Reinforcement Learning. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD)*, 2012.

An, G., Lee, J., Zuo, X., Kosaka, N., Kim, K.-M., and Song, H. O. Direct Preference-based Policy Optimization without Reward Modeling. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Bıyık, E., Lazar, D. A., Sadigh, D., and Pedarsani, R. The Green Choice: Learning and Influencing Human Decisions on Shared Roads. In *IEEE Conference on Decision and Control (CDC)*, 2019.

Bradley, R. A. and Terry, M. E. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika*, 39(3/4):324–345, 1952.

Brown, D., Goo, W., Nagarajan, P., and Niekum, S. Extrapolating Beyond Suboptimal Demonstrations via Inverse Reinforcement Learning from Observations. In *International Conference on Machine Learning (ICML)*, 2019.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., and Hullender, G. Learning to Rank using Gradient Descent. In *International Conference on Machine Learning (ICML)*, 2005.

Cao, Z., Wong, K., and Lin, C.-T. Weak Human Preference Supervision For Deep Reinforcement Learning. In *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 2021.

Casper, S., Davies, X., Shi, C., Gilbert, T. K., Scheurer, J., Rando, J., Freedman, R., Korbak, T., Lindner, D., Freire, P., et al. Open Problems and Fundamental Limitations of Reinforcement Learning from Human Feedback. In *arXiv preprint arXiv:2307.15217*, 2023.

Chebotar, Y., Handa, A., Makoviychuk, V., Macklin, M., Issac, J., Ratliff, N., and Fox, D. Closing the Sim-to-Real Loop: Adapting Simulation Randomization with Real World Experience. In *International Conference on Robotics and Automation (ICRA)*, 2019.

Chen, X., Zhong, H., Yang, Z., Wang, Z., and Wang, L. Human-in-the-loop: Provably Efficient Preference-based Reinforcement Learning with General Function Approximation. In *International Conference on Machine Learning (ICML)*, 2022.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep Reinforcement Learning from Human Preferences. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. In *arXiv preprint arXiv:2004.07219*, 2020.

Fürnkranz, J., Hüllermeier, E., Cheng, W., and Park, S.-H. Preference-based Reinforcement Learning: A Formal Framework and a Policy Iteration Algorithm. In *Machine Learning*, volume 89, pp. 123–156. Springer, 2012.

Gulcehre, C., Wang, Z., Novikov, A., Paine, T., Gómez, S., Zolna, K., Agarwal, R., Merel, J. S., Mankowitz, D. J., Paduraru, C., et al. RL Unplugged: A Suite of Benchmarks for Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft Actor-Critic: Off-policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. In *International Conference on Machine Learning (ICML)*, 2018.

Hejna, J. and Sadigh, D. Inverse Preference Learning: Preference-based RL without a Reward Function. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Hejna, J., Rafailov, R., Sikchi, H., Finn, C., Niekum, S., Knox, W. B., and Sadigh, D. Contrastive Preference Learning: Learning from Human Feedback without RL. In *arXiv preprint arXiv:2310.13639*, 2023.

Hejna III, D. J. and Sadigh, D. Few-Shot Preference Learning for Human-in-the-Loop RL. In *Conference on Robot Learning (CoRL)*, 2023.

Hwang, M., Lee, G., Kee, H., Kim, C. W., Lee, K., and Oh, S. Sequential Preference Ranking for Efficient Reinforcement Learning from Human Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward Learning from Human Preferences and Demonstrations in Atari. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation. In *Conference on Robot Learning (CoRL)*, 2018.

Kang, Y., Shi, D., Liu, J., He, L., and Wang, D. Beyond Reward: Offline Preference-guided Policy Optimization. In *arXiv preprint arXiv:2305.16217*, 2023.

Kaufmann, T., Weng, P., Bengs, V., and Hüllermeier, E. A Survey of Reinforcement Learning from Human Feedback. In *arXiv preprint arXiv:2312.14925*, 2023.

Kim, C., Park, J., Shin, J., Lee, H., Abbeel, P., and Lee, K. Preference Transformer: Modeling Human Preferences using Transformers for RL. In *International Conference on Learning Representations (ICLR)*, 2022.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. In *arXiv preprint arXiv:1412.6980*, 2014.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., and Pérez, P. Deep Reinforcement Learning for Autonomous Driving: A Survey. In *IEEE Transactions on Intelligent Transportation Systems (TITS)*, 2021.

Kostrikov, I., Nair, A., and Levine, S. Offline Reinforcement Learning with Implicit Q-Learning. In *Advances in Neural Information Processing Systems Workshop (NeurIPS Workshop)*, 2021.

Lee, K., Smith, L., Dragan, A., and Abbeel, P. B-Pref: Benchmarking Preference-Based Reinforcement Learning. In *Neural Information Processing Systems (NeurIPS)*, 2021a.

Lee, K., Smith, L. M., and Abbeel, P. Pebble: Feedback-Efficient Interactive Reinforcement Learning via Relabeling Experience and Unsupervised Pre-training. In *International Conference on Machine Learning (ICML)*, 2021b.

Li, A., Misra, D., Kolobov, A., and Cheng, C.-A. Survival Instinct in Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Liang, X., Shu, K., Lee, K., and Abbeel, P. Reward Uncertainty for Exploration in Preference-based Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Loshchilov, I. and Hutter, F. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations (ICLR)*, 2018.

Mazoure, B., Eysenbach, B., Nachum, O., and Tompson, J. Contrastive Value Learning: Implicit Models for Simple Offline RL. In *Conference on Robot Learning (CoRL)*, 2023.

McKinney, L., Duan, Y., Krueger, D., and Gleave, A. On The Fragility of Learned Reward Functions. In *arXiv preprint arXiv:2301.03652*, 2023.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing Atari with Deep Reinforcement Learning. In *arXiv preprint arXiv:1312.5602*, 2013.

Myers, V., Biyik, E., Anari, N., and Sadigh, D. Learning Multimodal Rewards from Rankings. In *Conference on Robot Learning (CoRL)*, 2022.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training Language Models to Follow Instructions with Human Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Palan, M., Shevchuk, G., Charles Landolfi, N., and Sadigh, D. Learning Reward Functions by Integrating Human Demonstrations and Preferences. In *Robotics: Science and Systems (RSS)*, 2019.

Park, J., Seo, Y., Shin, J., Lee, H., Abbeel, P., and Lee, K. SURF: Semi-supervised Reward Learning with Data Augmentation for Feedback-efficient Preference-based Reinforcement Learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Plackett, R. L. The Analysis of Permutations. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 24(2):193–202, 1975.

Shin, D., Dragan, A., and Brown, D. S. Benchmarks and Algorithms for Offline Preference-Based Reward Learning. In *Transactions on Machine Learning Research (TMLR)*, 2022.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. Mastering the Game of Go without Human Knowledge. *Nature*, 550(7676):354–359, 2017.

Sinha, S., Mandlekar, A., and Garg, A. S4RL: Surprisingly Simple Self-Supervision for Offline Reinforcement Learning in Robotics. In *Conference on Robot Learning (CoRL)*, 2022.

Song, F., Yu, B., Li, M., Yu, H., Huang, F., Li, Y., and Wang, H. Preference Ranking Optimization for Human Alignment. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2024.

Stiennon, N., Ouyang, L., Wu, J., Ziegler, D., Lowe, R., Voss, C., Radford, A., Amodei, D., and Christiano, P. F. Learning to Summarize from Human Feedback. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Sutton, R. S. and Barto, A. G. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Swezey, R., Grover, A., Charron, B., and Ermon, S. PiRank: Scalable Learning to Rank via Differentiable Sorting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Tan, J., Zhang, T., Coumans, E., Iscen, A., Bai, Y., Hafner, D., Bohez, S., and Vanhoucke, V. Sim-to-Real: Learning Agile Locomotion for Quadruped Robots. In *arXiv preprint arXiv:1804.10332*, 2018.

Tarasov, D., Nikulin, A., Akimov, D., Kurenkov, V., and Kolesnikov, S. CORL: Research-oriented Deep Offline Reinforcement Learning Library. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks Track*, 2023.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. DeepMind Control Suite. In *arXiv preprint arXiv:1801.00690*, 2018.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. In *arXiv preprint arXiv:2307.09288*, 2023.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

White, D., Wu, M., Novoseller, E., Lawhern, V., Waytowich, N. R., and Cao, Y. Rating-based Reinforcement Learning. In *International Conference on Machine Learning Workshop (ICML Workshop)*, 2023.

Wilson, A., Fern, A., and Tadepalli, P. A Bayesian Approach for Policy Learning from Trajectory Preference Queries. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

Wirth, C., Akrour, R., Neumann, G., Fürnkranz, J., et al. A Survey of Preference-Based Reinforcement Learning Methods. In *Journal of Machine Learning Research (JMLR)*, 2017.

Xia, F., Liu, T.-Y., Wang, J., Zhang, W., and Li, H. Listwise Approach to Learning to Rank: Theory and Algorithm. In *International Conference on Machine Learning (ICML)*, 2008.

Xu, J. and Li, H. Adarank: A Boosting Algorithm for Information Retrieval. In *International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2007.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning. In *Conference on Robot Learning (CoRL)*, pp. 1094–1100. PMLR, 2020.

Yu, T., Kumar, A., Chebotar, Y., Hausman, K., Levine, S., and Finn, C. Conservative Data Sharing for Multi-Task Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021a.

Yu, T., Kumar, A., Rafailov, R., Rajeswaran, A., Levine, S., and Finn, C. Combo: Conservative Offline Model-Based Policy Optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021b.

Zhang, J. Efficient Offline Preference-Based Reinforcement Learning with Transition-Dependent Discounting, 2023. URL https://openreview.net/forum?id=7kKyELnAhn.

Zhao, Y., Joshi, R., Liu, T., Khalman, M., Saleh, M., and Liu, P. J. SLiC-HF: Sequence Likelihood Calibration with Human Feedback. In *arXiv preprint arXiv:2305.10425*, 2023.

Zhu, B., Jordan, M., and Jiao, J. Principled Reinforcement Learning with Human Feedback from Pairwise or K-wise Comparisons. In *International Conference on Machine Learning (ICML)*, 2023.

A. Additional Experimental Analysis of LiRE

A.1. Experimental Results on `medium-expert` Dataset

We summarize the experimental results on the Meta-World `medium-expert` dataset in Table 8. LiRE outperforms significantly baselines for *sweep* and *hammer* tasks. While DPPO performs better than LiRE in the case of *box-close* task, DPPO performs poorly compared to basic MR in other tasks.

Table 8: Average success rate of the algorithms on Meta-World `medium-expert` dataset over six random seeds. We use 500 and 1000 preference feedbacks and LiRE significantly outperforms the existing baselines.

# of feedback	Algorithm	box-close	sweep	hammer
-	IQL with GT rewards	65.00 ± 9.98	85.33 ± 5.96	65.00 ± 11.36
500	MR	15.33 ± 6.99	42.67 ± 9.21	5.67 ± 6.97
	PT (Kim et al., 2022)	2.33 ± 3.54	57.33 ± 8.92	1.67 ± 2.92
	OPRL (Shin et al., 2022)	10.00 ± 9.87	30.00 ± 8.87	6.00 ± 4.62
	DPPO (An et al., 2023)	41.00 ± 9.50	12.33 ± 8.44	8.00 ± 10.07
	IPL (Hejna & Sadigh, 2024)	7.00 ± 9.50	15.33 ± 10.37	4.33 ± 4.23
	SeqRank (Hwang et al., 2023)	10.73 ± 7.58	53.8 ± 18.21	4.6 ± 5.27
	LiRE (ours)	25.26 ± 11.70	59.53 ± 26.92	50.20 ± 16.98
1000	MR	13.67 ± 11.57	50.00 ± 8.64	8.00 ± 8.25
	PT (Kim et al., 2022)	13.00 ± 12.26	21.00 ± 15.44	1.33 ± 2.98
	OPRL (Shin et al., 2022)	11.33 ± 6.80	44.33 ± 6.67	6.33 ± 5.47
	DPPO (An et al., 2023)	42.67 ± 15.52	14.33 ± 13.19	5.33 ± 4.85
	IPL (Hejna & Sadigh, 2024)	10.67 ± 6.90	16.67 ± 11.64	9.33 ± 9.14
	SeqRank (Hwang et al., 2023)	13.4 ± 8.89	68.45 ± 14.26	21.27 ± 17.86
	LiRE (ours)	27.53 ± 20.45	73.00 ± 25.68	41.66 ± 32.64

A.2. LiRE with Fewer Preference Feedbacks

If we have pre-collected independent pairwise preference data, MR can use the entire preference data. However, LiRE has the disadvantage of requiring additional feedbacks between segments for constructing RLT. Nevertheless, Table 9 shows the importance of RLT to obtain second-order preference. The performance of LiRE with 200 feedbacks is better than the performance using 1000 independent pairwise feedbacks.

Table 9: Average success rates on Meta-World `medium-replay` dataset. We use 1000 preference feedbacks for MR and 200 preference feedbacks for LiRE.

# of feedbacks	Algorithm	button-press -topdown	box-close	dial-turn	sweep	button-press -topdown-wall	sweep-into	drawer-open	lever-pull
-	IQL with GT rewards	88.33 ± 4.76	93.40 ± 3.10	75.40 ± 5.47	98.33 ± 1.87	56.27 ± 6.32	78.80 ± 7.96	100.00 ± 0.00	98.47 ± 1.77
1000	MR	9.27 ± 5.30	17.07 ± 9.56	59.07 ± 7.57	90.80 ± 9.74	0.60 ± 1.87	26.07 ± 8.57	96.47 ± 4.02	50.87 ± 10.89
200	LiRE	36.60 ± 16.30	60.33 ± 23.96	80.73 ± 10.53	76.87 ± 13.86	53.27 ± 24.12	39.33 ± 10.62	99.93 ± 0.36	81.47 ± 13.53

A.3. Training LiRE with Listwise Loss

Section 4.2 describes how to train the reward model with pairwise loss from constructed RLT. However, we can apply listwise loss in addition to pairwise loss since a ranked list is constructed. In this section, we introduce how to train the reward model with listwise loss. Suppose that we have n segments, $(\sigma_1, \sigma_2, \cdots, \sigma_n)$ and denote the rewards of the segments, $\big(r(\sigma_1), r(\sigma_2), \cdots, r(\sigma_n)\big)$. We assume the probability of permutation of n segments follows a Plackett-Luce (PL) model (Plackett, 1975):

$$P(\boldsymbol{\pi}) = \prod_{i=1}^{n} \frac{\phi\big(r(\sigma_{\pi_i})\big)}{\sum_{j=i}^{n} \phi\big(r(\sigma_{\pi_j})\big)} \tag{3}$$

where ϕ is an increasing and strictly positive function and $\boldsymbol{\pi} = (\pi_1, \pi_2, \cdots, \pi_n)$ is a permutation of $(1, 2, \cdots, n)$. Here, $P(\boldsymbol{\pi})$ is the probability distribution in which n segments are ranked in order of permutation $\boldsymbol{\pi}$, indicating the likelihood of segment σ_i being ranked π_i-th.

Since we do not know the true probability of permutation, we set the score of the segment based on the ranks. Specifically, given k ranks in the list, let $s(\sigma) = (k + 1 - m)R/k$ be the score of the segment σ that belongs to the m-th preferred rank (*i.e.,* $\sigma \in g_{k+1-m}$) where $m \in \{1, \cdots, k\}$ and R is constant. In our implementation, the constant R is set to the maximum boundary of the output of the reward model, which is bounded by $[0, R]$ by the Tanh function.

Our goal is to minimize the following objective:

$$D_{KL}\left(P_s(\boldsymbol{\pi})\|P_\theta(\boldsymbol{\pi})\right) = D_{KL}\left(\prod_{i=1}^{n}\frac{\phi(s(\sigma_{\pi_i}))}{\sum_{j=i}^{n}\phi(s(\sigma_{\pi_j}))}\middle\|\prod_{i=1}^{n}\frac{\phi\big(r_\theta(\sigma_{\pi_i})\big)}{\sum_{j=i}^{n}\phi\big(r_\theta(\sigma_{\pi_j})\big)}\right). \tag{4}$$

Since the number of permutations grows by $n!$, computing the permutation probability demands a high computational cost. Thus, we minimize the following objective using the top one probability proposed by ListNet (Xia et al., 2008):

$$\sum_{i=1}^{n} D_{KL}\left(P_s(i)\|P_\theta(i)\right) \tag{5}$$

where

$$P_s(i) = P_s(\pi_1 = i) = \frac{\phi(s(\sigma_i))}{\sum_{j=1}^{n}\phi(s(\sigma_j))} \tag{6}$$

and

$$P_\theta(i) = P_\theta(\pi_1 = i) = \frac{\phi(r_\theta(\sigma_i))}{\sum_{j=1}^{n}\phi(r_\theta(\sigma_j))}. \tag{7}$$

We train the reward model by sampling $n = 10$ segments from the RLT at each gradient descent. Table 10 compares the performance of LiRE trained with listwise and pairwise losses. As shown in Table 10, training the reward model with pairwise loss is more stable and performs better in most cases, except for *sweep* and *sweep-into* tasks.

Table 10: Average success rates on `medium-replay` dataset when using the listwise loss for training the reward model.

# of feedbacks	Algorithm	button-press -topdown	box-close	dial-turn	sweep	button-press -topdown-wall	sweep-into	drawer-open	lever-pull
-	IQL with GT rewards	88.33 ± 4.76	93.40 ± 3.10	75.40 ± 5.47	98.33 ± 1.87	56.27 ± 6.32	78.80 ± 7.96	100.00 ± 0.00	98.47 ± 1.77
500	LiRE w/ listwise	53.13 ± 10.63	55.07 ± 16.11	63.87 ± 8.39	99.53 ± 1.12	17.73 ± 11.51	63.47 ± 11.47	98.60 ± 3.27	84.53 ± 10.33
	LiRE w/ pairwise	67.20 ± 18.97	51.53 ± 18.48	79.07 ± 10.96	77.53 ± 10.50	79.13 ± 15.19	49.13 ± 15.85	99.40 ± 1.65	95.67 ± 6.26
1000	LiRE w/ listwise	55.73 ± 8.57	68.07 ± 9.06	68.20 ± 9.37	99.07 ± 1.44	23.93 ± 7.31	62.60 ± 12.21	99.40 ± 2.08	83.80 ± 7.97
	LiRE w/ pairwise	83.07 ± 6.38	89.13 ± 6.02	76.93 ± 7.55	75.87 ± 6.81	81.47 ± 10.04	57.73 ± 13.11	99.73 ± 0.85	99.47 ± 1.15

A.4. Increasing Epochs of Reward Model Training

As described in Appendix C.5, the epochs experimented with in Table 2 is 300. Table 11 shows the performance when we increase the epochs to 5000. Both MR and LiRE tend to perform better with more epochs, but LiRE still performs better than MR. The performance gap between using the exponential score function and the linear score function for LiRE is smaller at 5000 epochs than at 300 epochs. However, when the epoch is 5000, the linear score function has a significant performance improvement on the *dial-turn* and *button-press-topdown-wall* tasks and performs better or similar to the exponential score function on other tasks.

Table 11: Average success rates on Meta-World `medium-replay` dataset with increased epochs. There is a performance improvement when training with longer epochs.

Epochs	Algorithm	button-press -topdown	box-close	dial-turn	sweep	button-press -topdown-wall	sweep-into	drawer-open	lever-pull
-	IQL with GT rewards	88.33 ± 4.76	93.40 ± 3.10	75.40 ± 5.47	98.33 ± 1.87	56.27 ± 6.32	78.80 ± 7.96	100.00 ± 0.00	98.47 ± 1.77
300	MR	9.60 ± 5.74	10.33 ± 8.23	50.20 ± 8.51	79.80 ± 13.36	0.13 ± 0.50	24.80 ± 5.28	98.07 ± 3.20	50.53 ± 8.55
	LiRE w/ exp	12.87 ± 7.86	22.73 ± 10.40	65.87 ± 9.46	82.67 ± 19.86	1.33 ± 2.15	24.87 ± 8.39	98.67 ± 1.89	57.87 ± 11.28
	LiRE w/ linear	67.20 ± 18.97	51.53 ± 18.48	79.07 ± 10.96	77.53 ± 10.50	79.13 ± 15.19	49.13 ± 15.85	99.40 ± 1.65	95.67 ± 6.26
5000	MR	32.87 ± 9.94	31.80 ± 12.65	60.33 ± 8.34	93.5 ± 6.61	25.40 ± 10.25	36.00 ± 9.58	98.00 ± 4.00	75.93 ± 6.46
	LiRE w/ exp	68.33 ± 16.33	83.13 ± 10.36	77.53 ± 6.25	91.87 ± 7.02	36.80 ± 14.17	59.53 ± 14.99	99.93 ± 0.36	79.47 ± 8.61
	LiRE w/ linear	77.27 ± 13.52	76.6 ± 17.16	88.33 ± 5.49	87.6 ± 15.45	77.27 ± 13.52	60.67 ± 11.96	97.07 ± 5.63	83.4 ± 6.43

A.5. Applying a Linear Score Function to Other Baselines

Many existing studies utilize the exponential score function for PbRL using human feedback (Christiano et al., 2017; Lee et al., 2021b). Nevertheless, numerous other score functions are also prevalent in the PbRL literature such as Table 1 in

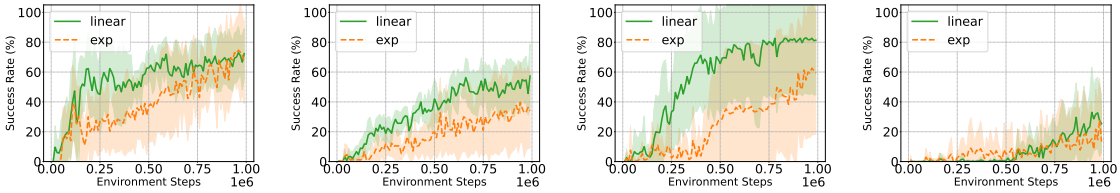
the survey paper of PbRL (Wirth et al., 2017). Additionally, (Song et al., 2024) have demonstrated that alternative score functions are effective in RLHF. We also present the performance of baselines using the linear score function instead of the exponential function across four Meta-World `medium-replay` tasks in Table 12. Table 12 reveals that LiRE, when utilizing the linear score function, surpasses all other baselines, even when these baselines also use the linear score function. For PT or DPPO, there is no performance improvement when using a linear score function. We leave it as future work to analyze which score functions are effective depending on the model structure or training method.

Table 12: Average success rates when applying the linear and exponential score functions to the baselines.

Task	$\phi(x)$	MR	PT	OPRL	DPPO	SeqRank	LiRE
button-press-topdown	$\exp(x)$	9.60 $\pm$ 5.74	22.87 $\pm$ 9.06	12.13 $\pm$ 5.75	3.93 $\pm$ 4.34	20.00 $\pm$ 3.54	12.87 $\pm$ 7.86
	x	36.87 $\pm$ 13.75	17.33 $\pm$ 10.7	30.00 $\pm$ 7.21	1.33 $\pm$ 2.31	54.87 $\pm$ 9.89	67.20 $\pm$ 18.97
box-close	$\exp(x)$	10.33 $\pm$ 8.23	0.33 $\pm$ 1.16	4.73 $\pm$ 3.24	10.20 $\pm$ 11.47	26.8 $\pm$ 3.47	22.73 $\pm$ 10.40
	x	11.27 $\pm$ 14.91	1.33 $\pm$ 2.31	47.33 $\pm$ 39.71	3.33 $\pm$ 4.16	46.67 $\pm$ 36.49	51.53 $\pm$ 18.48
dial-turn	$\exp(x)$	50.20 $\pm$ 8.51	68.67 $\pm$ 12.39	54.33 $\pm$ 11.47	26.67 $\pm$ 22.23	62.27 $\pm$ 5.97	65.87 $\pm$ 9.46
	x	77.27 $\pm$ 11.9	56.67 $\pm$ 10.87	71.33 $\pm$ 9.87	22.0 $\pm$ 21.07	59.80 $\pm$ 17.73	79.07 $\pm$ 10.96
lever-pull	$\exp(x)$	50.53 $\pm$ 8.55	82.40 $\pm$ 22.69	96.0 $\pm$ 4.00	10.13 $\pm$ 12.19	97.07 $\pm$ 1.80	57.87 $\pm$ 11.28
	x	70.20 $\pm$ 18.03	78.8 $\pm$ 13.87	86.67 $\pm$ 12.2	4.00 $\pm$ 5.67	74.33 $\pm$ 18.50	95.67 $\pm$ 6.26

A.6. Online PbRL

Figure 7 depicts the experimental results of online PbRL. We compare the online PbRL performance by using a linear score function and an exponential score function. The increase in performance when using the linear score function suggests that the BT model using the exponential score function may not be the optimum choice for PbRL. We used the code implemented in PEBBLE (Lee et al., 2021b).



(a) dial-turn (b) hand-insert (c) sweep-into (d) lever-pull

Figure 7: Learning curves on online PbRL experiments. Using linear score function in online PbRL improves performance.

B. Details of the Main Experimental Results

B.1. Full Learning Curves of Each Method

Full learning curves for the Meta-World `medium-replay` dataset are shown in Figure 8 and for the Meta-World `medium-expert` dataset are shown in Figure 9. We plot the results for MR, PT (Kim et al., 2022), OPRL (Shin et al., 2022), DPPO (An et al., 2023), IPL (Hejna & Sadigh, 2024), and LiRE. The average success rates reported in Table 2 are obtained with the last 5 trained policies. Although the performance of LiRE for the `medium-replay` *sweep* task is relatively low, the full learning curve shows that the performance of the best-trained policy is competitive.

B.2. Ablation Study of LiRE

We evaluate the success rate by the following: (1) with MR or with LiRE and (2) exponential or linear score function. Table 13 shows that both constructing RLT and using linear function improve offline PbRL performance.

B.3. Effect of RLT and Score Function on Reward Estimation

Similar to Figure 2, in *button-press-topdown* task, Figure 10 shows that constructing RLT and using a linear score function can better distinguish the rewards between segments with relatively high preference.

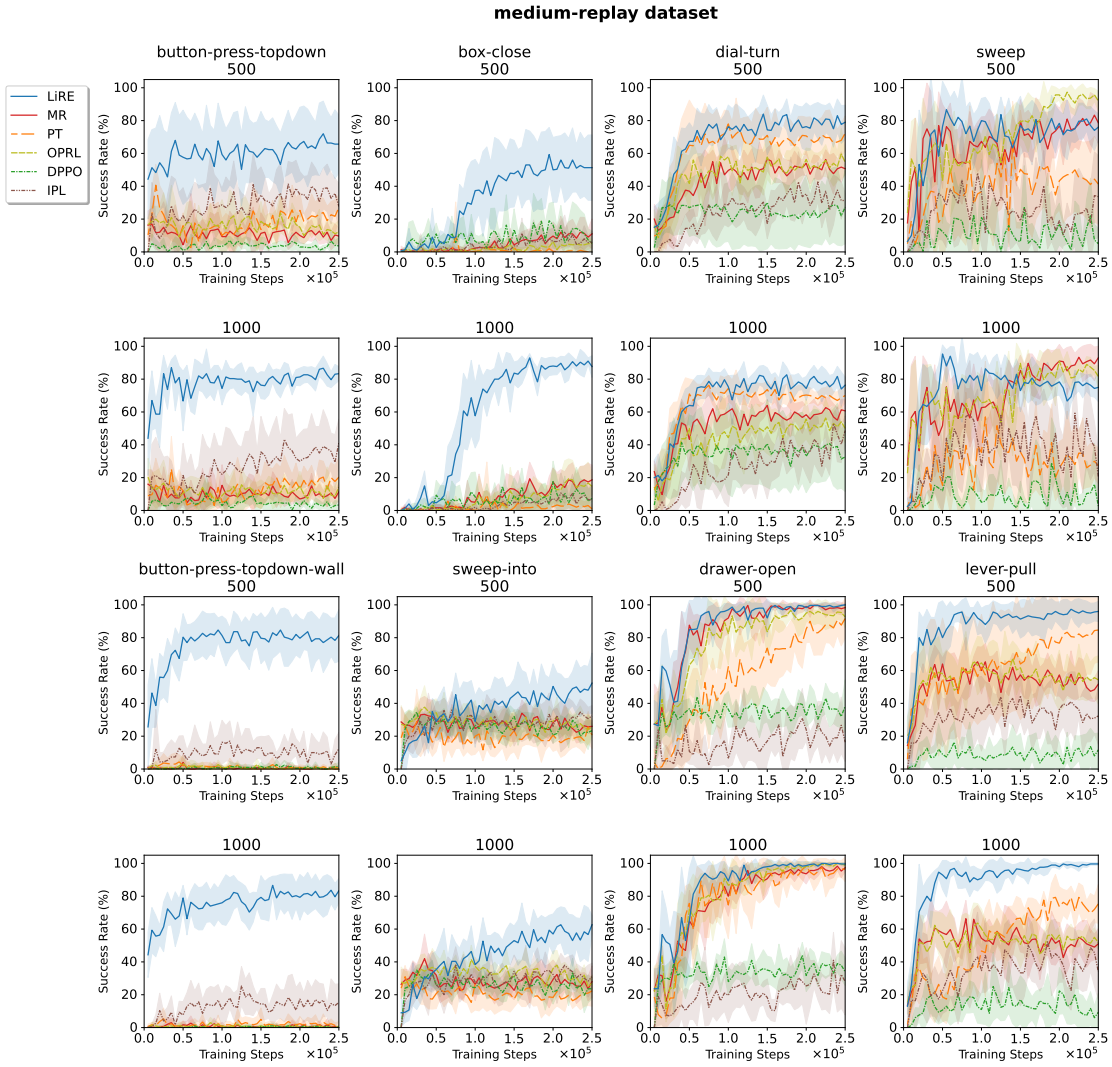


Figure 8: Full learning curves of baselines and LiRE for the Meta-World `medium-replay` dataset. We use 500 and 1000 preference feedbacks and LiRE significantly outperforms existing algorithms for many tasks.

B.4. Comparison with SeqRank

Table 14 shows the success rate of each task in Table 5. SeqRank (Hwang et al., 2023) improves feedback efficiency but constructs a shorter length of the ranked list, so LiRE is better at utilizing second-order preference.

C. Experimental Details

C.1. RLT Construction

To construct RLT, We can use any sorting method such as binary insertion sort, mergesort, or quicksort. However, if the RLT is already partially constructed, a binary insertion sort is an efficient way to find the rank of each segment. The pseudocode for the binary insertion sort we use to construct the RLT is summarized in Algorithm 1.

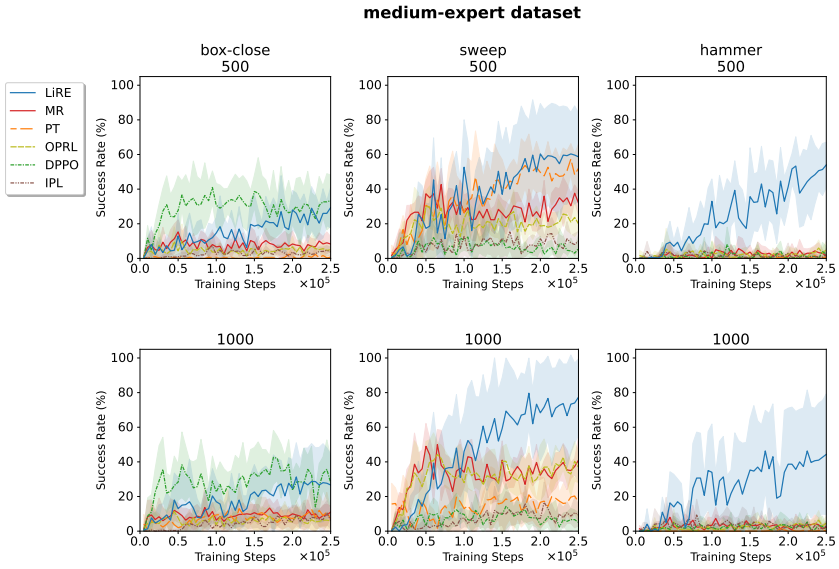
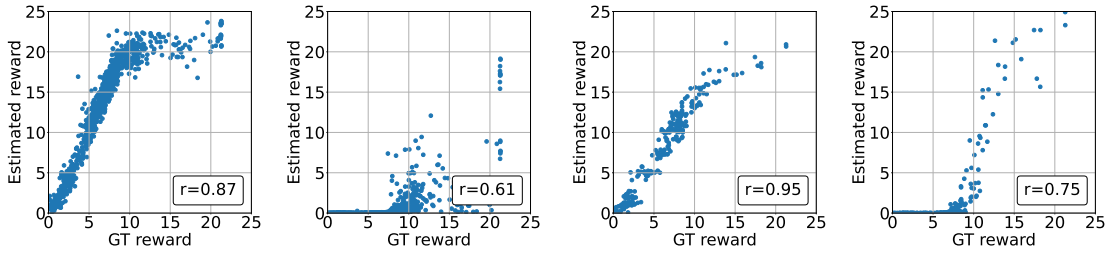


Figure 9: Full learning curves of baselines and LiRE for the Meta-World `medium-expert` dataset. We use 500 and 1000 preference feedbacks. LiRE significantly outperforms existing algorithms for *sweep* and *hammer* tasks.

Table 13: Average success rates on Meta-World `medium-replay` dataset with 500 preference feedbacks. We train the reward model with MR or LiRE using exponential or linear score functions.

Task	$\phi(x) = \exp(x)$		$\phi(x) = x$	
	MR	LiRE	MR	LiRE
button-press-topdown	9.60 ± 5.74	12.87 ± 7.86	36.87 ± 13.75	67.20 ± 18.97
box-close	10.33 ± 8.23	22.73 ± 10.40	11.27 ± 14.91	51.53 ± 18.48
dial-turn	50.20 ± 8.51	65.87 ± 9.46	77.27 ± 11.90	79.07 ± 10.96
sweep	79.80 ± 13.36	82.67 ± 19.86	78.47 ± 10.88	77.53 ± 10.50
button-press-topdown-wall	0.13 ± 0.50	1.33 ± 2.15	8.27 ± 8.64	79.13 ± 15.19
sweep-into	24.80 ± 5.28	24.87 ± 8.39	49.73 ± 13.52	49.13 ± 15.85
drawer-open	98.07 ± 3.20	98.67 ± 1.89	97.20 ± 5.88	99.40 ± 1.65
lever-pull	50.53 ± 8.55	57.87 ± 11.28	70.20 ± 18.03	95.67 ± 6.26



(a) MR w/ exp (b) MR w/ linear (c) LiRE w/ exp (d) LiRE w/ linear

Figure 10: Estimated rewards for the segments used in preference learning for *box-close* task. We train the reward model with MR or LiRE using the exp or linear score function. The Pearson correlation coefficient, r is presented.

C.2. Creating Offline PbRL Dataset

Following offline RL data collection approach, we collect offline RL data from different policies in two ways: `medium-replay` dataset and `medium-expert` dataset.

`medium-replay` dataset We use the replay buffer collected while training online RL as an offline RL dataset. We train with 3 seeds using the online SAC (Haarnoja et al., 2018) implemented in PEBBLE (Lee et al., 2021b) with ground-truth

Table 14: Average success rate and the number of ranks in the list of SeqRank and LiRE.

Task	Number of feedacks			
	500		1000	
	SeqRank w/ linear	LiRE	SeqRank w/ linear	LiRE
button-press-topdown	54.87 ± 9.89	67.20 ± 18.97	60.13 ± 11.43	83.07 ± 6.38
box-close	46.67 ± 36.49	51.53 ± 18.48	65.00 ± 32.90	89.13 ± 6.02
dial-turn	59.80 ± 17.73	79.07 ± 10.96	45.67 ± 16.47	76.93 ± 7.55
sweep	86.33 ± 8.25	77.53 ± 10.50	94.27 ± 3.92	75.87 ± 6.81
button-press-topdown-wall	24.07 ± 16.39	79.13 ± 15.19	37.07 ± 16.96	81.47 ± 10.04
sweep-into	48.80 ± 18.72	49.13 ± 15.85	61.53 ± 13.57	57.73 ± 13.11
drawer-open	99.87 ± 0.50	99.40 ± 1.65	100.00 ± 0.00	99.73 ± 0.85
lever-pull	74.33 ± 18.50	95.67 ± 6.26	76.27 ± 13.27	99.47 ± 1.15
Avg success rate	61.84 ± 15.80	**74.83** ± 12.23	67.49 ± 13.56	**82.92** ± 6.48
Avg # of ranks in the list	2.3 ± 0.09	**9.3** ± 1.83	2.3 ± 0.09	**9.3** ± 1.84

rewards. We stop collecting replay buffers when the average success rate of the online RL's performance is near 50. (For the DMControl dataset, collect until the episode returns are about in the middle of the convergence value.) We measure the online RL performance every 50,000 steps, so depending on the training speed of the online RL, the average success rate of the online RL may be less or more than 50 at the end of the replay buffer collection. Table 15 shows the average success rate when the collection of the replay buffers ends.

Table 15: Average success rate of online RL when replay buffer collection ends.

button-press-topdown	button-press-topdown-wall	box-close	dial-turn
47.0	36.0	46.0	49.3
sweep-v2	sweep-into	drawer-open	lever-pull
32.0	55.3	61.3	78.0

`medium-expert` dataset We collect `medium-expert` dataset following approaches by prior works (Hejna & Sadigh, 2024; Zhang, 2023): collect 50 trajectories from the expert policy provided by Meta-World (Yu et al., 2020), collect 50 trajectories from the expert policy for a different randomized object and goals positions, collect 100 trajectories from the expert policy for a different task out of 50 Meta-World tasks, collect 200 trajectories from a random policy, and finally, collect 200 trajectories from the ϵ-greedy policy that samples an action from the expert policy with 50% probability and from the random policy with the remaining 50% probability. We also add Gaussian noise with a mean of 0 and a standard deviation of 1 for each policy.

C.3. RL Performance between GT Reward and Wrong Rewards

For each dataset, we verify that there is a difference in RL performance when trained with GT reward versus wrong rewards because if offline RL achieves high performance with wrong rewards, the dataset is not appropriate for offline PbRL. We use the three wrong rewards chosen by (Li et al., 2023): zero rewards, where all rewards $r(s, a) = 0$; random rewards, where all reward values are sampled from a uniform distribution $U(0, 1)$; and negative rewards, set to $-r(s, a)$. The performance of offline RL with GT reward and wrong rewards on each dataset is shown in Table 16 and Table 17.

C.4. Preference Label

We set the length of segment σ used in the preference label to 25, denoted as $T = 25$ in $\sigma = (s_0, a_1, \cdots, s_{T-1}, a_{T-1})$. We use the GT reward to label the preference between segment pairs. Considering that GT reward in Meta-World ranges from 0 to 10, segments with GT reward differences less than 12.5 are labeled as equally preferred segments. This threshold is equivalent to the threshold provided by B-pref (Lee et al., 2021a), which is used as an online PbRL benchmark, when the policy has an average return of 5 (that is, medium performance).

C.5. Hyperparameters

Reward model The reward model used in our method and the standard pairwise PbRL and MR reward model use the same reward model structure. We ensemble three reward models and finally predicted the reward in the offline RL dataset by averaging the estimated reward values from the three reward models. The details of the hyperparameters are shown in

Algorithm 1 RLT Construction

function BINARYSEARCH $(\sigma, \texttt{low}, \texttt{high}, L)$:
  if $\texttt{low} = \texttt{high}$ **then**
    insert a new group $\{\sigma\}$ to L right behind to $g_{\texttt{low}+1}$
    (i.e., $g_{\texttt{low}} \prec \{\sigma\} \prec g_{\texttt{low}+1}$)
  else
    /* Human Feedback */
    compare σ to $\sigma_s \in g_{\texttt{mid}}$ where $\texttt{mid} = \lceil \frac{\texttt{low}+\texttt{high}}{2} \rceil$
    if $\sigma_s \prec \sigma$ **then**
      BINARYSEARCH$(\sigma, \texttt{mid}, \texttt{high}, L)$
    else if $\sigma \prec \sigma_s$ **then**
      BINARYSEARCH$(\sigma, \texttt{low}, \texttt{mid} - 1, L)$
    else
      $g_{\texttt{mid}} \leftarrow g_{\texttt{mid}} \cup \{\sigma\}$
Init: List $L = [\,]$
repeat
  sample $\sigma_1, \sigma_2, \cdots \in D_s$
  if L is empty **then**
    $L \leftarrow [\{\sigma_i\}]$
  else
    BINARYSEARCH$(\sigma_i, 0, l, L)$
until end of feedback
Output: L

Table 16: Average success rate of each dataset on GT rewards and wrong rewards with IQL (Kostrikov et al., 2021).

	Task	GT	Zero	Random	Negative
medium-replay dataset	button-press-topdown	88.33 ± 4.76	12.07 ± 5.76	13.00 ± 5.36	0.00 ± 0.00
	box-close	93.40 ± 3.10	0.53 ± 0.88	0.13 ± 0.50	0.13 ± 0.50
	dial-turn	75.40 ± 5.47	16.07 ± 6.44	13.93 ± 7.70	2.40 ± 3.32
	sweep	98.33 ± 1.87	0.20 ± 0.60	0.40 ± 0.80	0.00 ± 0.00
	button-press-topdown-wall	56.27 ± 6.32	1.67 ± 1.64	1.13 ± 1.77	0.00 ± 0.00
	sweep-into	78.80 ± 7.96	24.73 ± 7.26	23.40 ± 7.23	0.07 ± 0.36
	drawer-open	100.00 ± 0.00	25.67 ± 10.65	22.33 ± 11.66	0.00 ± 0.00
	lever-pull	98.47 ± 1.77	1.27 ± 1.50	1.20 ± 1.51	0.00 ± 0.00
medium-expert dataset	box-close	65.00 ± 9.98	3.67 ± 4.68	2.67 ± 3.77	1.00 ± 1.53
	sweep	85.33 ± 5.96	5.00 ± 10.31	2.00 ± 3.65	0.00 ± 0.00
	hammer	65.00 ± 11.36	2.33 ± 5.22	1.67 ± 3.73	1.33 ± 2.21

Table 17: Episode returns of each DMControl `medium-replay` dataset on GT rewards and wrong rewards with IQL (Kostrikov et al., 2021).

Task	GT	Zero	Random	Negative
hopper-hop	157.95 ± 9.64	18.9 ± 7.5	19.79 ± 7.47	0.01 ± 0.02
walker-walk	839.6 ± 36.57	189.58 ± 28.15	234.14 ± 37.22	28.79 ± 2.2
humanoid-walk	250.9 ± 11.62	60.36 ± 10.56	65.13 ± 10.16	1.38 ± 0.21

Table 18.

In our experiments, MR, PT, and OPRL are two-step PbRL methods that first train the reward model and learn the offline RL with the trained reward model. We use the trained reward model to estimate the reward for every (s, a) in the offline RL dataset and apply min-max normalization to the reward values in the dataset so that the minimum and maximum values are 0 and 1. We also apply min-max normalization to the experiments with GT rewards and wrong rewards for a fair comparison.

Implementation details We choose IQL for the default offline RL algorithm and CORL (Tarasov et al., 2023) for the

implementation code[1]. We use IQL because it is the default offline RL algorithm in previous offline PbRL papers, and IQL is also one of the strongest offline algorithms according to CORL. We use the same hyperparameters that were used to train Gym-MuJoCO in CORL. For PT, we follow their implementation[2] for the training reward model and use the CORL library for training offline RL. We follow the official implementations of DPPO[3] and IPL[4] with the hyperparameters they use in the Gym-MuJoCo and Metaworld dataset. The hyperparameters for each baseline, including IQL, are listed in Table 18. The total number of gradient descent steps in the offline RL is 250,000 and we evaluate the success rate for 50 episodes every 5000 steps. We run six seeds for all baselines and our method. We then report the average success rate of the last 5 trained policies. We use a single NVIDIA RTX A5000 GPU and 32 CPU cores (AMD EPYC 7513 @ 2.60GHz) in our experiments.

[1] https://github.com/tinkoff-ai/CORL
[2] https://github.com/csmile-1006/PreferenceTransformer
[3] https://github.com/snu-mllab/DPPO
[4] https://github.com/jhejna/inverse-preference-learning

Table 18: Hyperparameters of the reward model and the baselines.

	Hyperparameter	Value
Reward model	Optimizer	Adam (Kingma & Ba, 2014)
	Learning rate	1e-3
	Batch size	512
	Q	100
	Hidden layer dim	128
	Hidden layers	3
	Activation function	ReLU
	Final activation	Tanh
	Epochs	300
	# of ensembles	3
	Reward from the ensemble models	Average
IQL (Kostrikov et al., 2021)	Optimizer	Adam (Kingma & Ba, 2014)
	Critic, Actor, Value hidden dim	256
	Critic, Actor, Value hidden layers	2
	Critic, Actor, Value activation function	ReLU
	Critic, Actor, Value learning rate	0.5
	Mini-batch size	256
	Discount factor	0.99
	β	3.0
	τ	0.7
PT (Kim et al., 2022)	Optimizer	AdamW (Loshchilov & Hutter, 2018)
	# of layers	1
	# of attention heads	4
	Embedding dimension	256
	Dropout rate	0.1
IPL (Hejna & Sadigh, 2024)	Optimizer	Adam (Kingma & Ba, 2014)
	Regularization λ	3e-4
	Q, V, π arch	3x256d
	β	4.0
	τ	0.7
	Subsample s	16
DPPO (An et al., 2023)	Preference predictor	The same as PT (Kim et al., 2022)
	Smoothness regularization ν	1.0
	Smoothness sigma m	20
	Regularization λ	0.5
OPRL (Shin et al., 2022)	# of ensembles	7
	Initial preference labels	30% of feedback budget
	Every 50 epochs	10% of feedback budget
	Total epochs	500