Exploiting Action Distances for Reward Learning from Human Preferences

Mudit Verma [1] **Siddhant Bhambri** [1] **Subbarao Kambhampati** [1]

Abstract

Preference-based Reinforcement Learning (PbRL) with binary preference feedbacks over trajectory pairs has proved to be quite effective in learning complex preferences of a human in the loop in domains with high dimensional state spaces and action spaces. While the human preference is primarily inferred from the feedback provided, we propose that, in situations where the human preferences are goal-oriented, the policy being learned (jointly with the reward model) during training can also provide valuable learning signal about the probable goal based on the human preference. To utilize this information, we introduce an action distance measure based on the policy and use it as an auxiliary prediction task for reward learning. This measure not only provides insight into the transition dynamics of the environment but also informs about the reachability of states under the policy by giving a distance to goal measure. We choose six tasks with goal-oriented preferences in the Meta-World domains to evaluate the performance and sample efficiency of our approach. We show that our approach outperforms methods leveraging auxiliary tasks of learning environment dynamics or a non-temporal distance measure adapted by PbRL baselines. Additionally, we show that action distance measure can also accelerate policy learning which is reaffirmed by our experimental results.

1. Introduction

Preference-based Reinforcement learning (PbRL) is a promising paradigm for training agents to learn from human preferences (Leike et al., 2018; Akrour et al., 2011; Ibarz et al., 2018b; Bakker et al., 2022; Köster et al., 2020). While human feedback can be obtained and incorporated in several ways, the primary objective of PbRL is to distill information from binary preference feedbacks on queried agent behavior trace pairs into the reward function (Wilson et al., 2012; Christiano et al., 2017). Recent advances in PbRL have led to algorithms that are capable of successfully learning human preferences on simpler discrete tasks (Verma & Metcalf, 2022; Soni et al., 2022) to more complex continuous control tasks (Lee et al., 2021a; Park et al., 2022). A key challenge in PbRL has been to reduce human sample feedback complexity and increase agent performance based on the given preference. Typically, prior works have investigated research directions like improving the query sampling strategy (Lee et al., 2021a), performing state or trajectory augmentation in the queries (Park et al., 2022; Guan et al., 2021), unsupervised policy pre-training (Lee et al., 2021a) and learning world models as reward priors (Verma & Metcalf, 2022). However, exploiting the agent policy being learned along with the reward model has not been explored before.

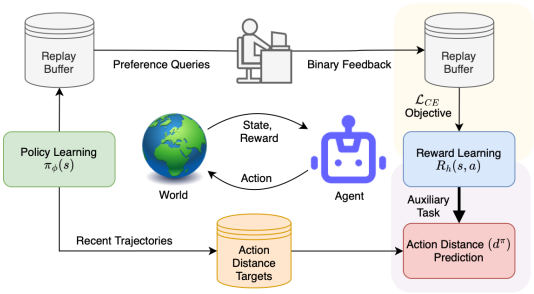


Figure 1. Overview of the proposed approach. Typical to PbRL setups, we have our agent acting in the world & saving state, action, and reward data into its buffer. Every few episodes, the human in the loop is queried for their preference over the agent's behavior trajectory. The aim of the agent is to learn a reward model (reward learning) along with obtaining a good policy (policy learning). We are proposing a method for preference-based reinforcement learning (PbRL) that uses an action distance measure based on the policy being learned as an auxiliary prediction task for reward learning.

We posit that human preferences can either be behavior-oriented, goal-oriented, or both. Behavior-oriented preferences include a preference, say, towards a certain style of walking or a way to swim, and these preferences could themselves be decomposed into tacit and symbolic components (Guan et al., 2022). In this work, we are interested in goal-oriented preferences which could be preferences on walking towards a door or reaching for an object. Con-

[1]SCAI, Arizona State University, USA. Correspondence to: Mudit Verma <mverma12@asu.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

cretely, goal-oriented preferences contain an absorbing state where the agent execution would halt by virtue of successful completion or failure (and not necessarily because of time limit constraints of the environment). Several benchmark domains popularly used by the PbRL community are in fact goal-oriented. These include the robotic manipulation tasks of Meta-World like Button-Press, Door-Open, and Window-Open, to name a few. The oracular approach typically taken by PbRL works for consistent trajectory feedbacks and scalable experimentation over these domains uses the underlying task reward as the human reward function. A closer inspection of these rewards clearly shows their goal-oriented nature (see Appendix F.6).

In the context of goal-oriented preferences, we propose that the learned policy function could very well be utilized to improve the reward learning process since it can inform about the probable goal state indicated by the human preference (or more generally the state space organization and reachability among states). In the existing line of research (Park et al., 2022; Lee et al., 2021a; Christiano et al., 2017), the reward learning process can only distinguish between the states based on the given state representation as temporal features are not provided to the agent. However, for goal-oriented preferences, a more suitable feature would be a distance measure that can not only distinguish between the states based on the organization of the state space but is also in correspondence with the distance-to-goal property.

A policy trained on the reward model (which itself is non-stationary and is updated iteratively), would encapsulate both of our sought-after attributes. For example, a bank of sampled trajectories using the said policy contains information about which states could be reached from which other states, i.e., **reachability**, which actions would lead the agent to be in a certain part of the Markov Decision Process (MDP), i.e., **environment dynamics**. And, finally, since the policy was trained to maximize returns predicted by the reward model, a good policy starting from any state would sample trajectories where successor states, in expectation, progress towards the intended goal, i.e., **distance to goal**. We intend to extract these key pieces of information available to the agent via policy learning to improve the reward learning in PbRL as shown in Figure 1.

We propose a self-supervised method for learning the temporal distance (action distance) between states using a jointly trained policy function, which captures weak learning signals of the intermediate policy function trained on the reward model. To improve PbRL using this action distance measure, we propose that the reward model being learned (particularly an embedding space in the reward model) must be predictive of this action distance. We operationalize this by treating the action distance prediction objective as an additional auxiliary task for the reward model, thus forcing the embedding space to preserve the action distance between any pair of states. Our experiments on six (goal-oriented) continuous control robotic manipulation tasks (Meta-World), commonly used in recent PbRL works (Park et al., 2022; Lee et al., 2021a), show that the use of action distance based auxiliary task in the reward learning process is an effective means of boosting the agent's performance when learning from goal-oriented preferences.

We highlight the main contributions of the work as follows : 1. This is the first work that leverages valuable learning signals from the joint policy being learned to improve reward learning in the context of goal-oriented human preferences and PbRL. 2. We propose an action distance based auxiliary task for the reward model that can be easily incorporated into any PbRL algorithm. 3. We benchmark the proposed work against state-of-the-art PbRL algorithms, as well as adapted PbRL algorithms that share certain characteristics of action distance based auxiliary task.

2. Related Work

Readers are encouraged to read Appendix A.1, A.2 as preliminaries for PbRL and multi-dimensional scaling.

Preference-based Reinforcement Learning. There are several works in the RL literature (Liu et al., 2023; Bewley & Lecue, 2021; Zhang & Kashima, 2023; Liu & Chen, 2022; Wirth et al., 2017) that focus on acquiring ratings or feedback from the human-in-the-loop (Knox & Stone, 2009; Christiano et al., 2017; Ibarz et al., 2018a; Stiennon et al., 2020). (Knox & Stone, 2009) was one of the foremost works to incorporate human-in-the-loop binary feedback to aid the agent's learning for solving the problem of sparse environment reward. However, the framework presented in (Knox & Stone, 2009) and further extensions of it were restricted to querying the user over state preferences. (Christiano et al., 2017) proposed a deep RL framework that queried the human user for trajectory preferences instead, by asking the user to choose the preferred trajectory over the other based on the Bradley-Terry model (Bradley & Terry, 1952), which has further been incorporated in several works that followed (Lee et al., 2021a; Park et al., 2022; Guan et al., 2022; Liu et al., 2023; Liang et al., 2022). While (Lee et al., 2021a) proposed to use unsupervised pre-training to query diverse behaviors to the teacher, (Park et al., 2022) utilized data augmentation techniques to learn the reward model combined with a semi-supervised learning approach to utilize the unlabeled trajectories. Both of these works jointly learn the reward and the policy model for the agent. (Ibarz et al., 2018a) combined using trajectory preferences with expert demonstrations. We extend the PbRL literature in two key ways. In line with (Guan et al., 2022)'s view that preferences contain both tacit and symbolic components (Kambhampati et al., 2022), we rather distinguish between behavior-oriented and goal-oriented preferences. Second,

we show that action distance is a good measure encapsulating the weak learning signal from the jointly trained policy function and use it to improve the reward learning for goal-oriented preferences.

Representation learning using distance measures. Several works within and outside the field of reinforcement learning have focused on learning distance-guided representations that allow for learning structured representations (Florensa et al., 2019; 2018; Pong et al., 2019; Nair et al., 2018; Li et al., 2020; Shen et al., 2018; Roh et al., 2021) which prove to be useful for the downstream tasks.

It is known that guidance or heuristics informing about the goal, or distance-to-goal can help with both known (Hart et al., 1968; Browne et al., 2012; Zhong et al., 2013; Hoeller et al., 2020; Bejjani et al., 2018) and unknown MDPs (Cheng et al., 2021; Wagener et al., 2021; Guan et al., 2021; Garcıa & Fernández, 2015). While several prior works have utilized the notion of action distances or commute times, closest to our work would be research in goal-conditioned RL (Hartikainen et al., 2019) and (Venkattaramanujam et al., 2019). This first work proposes to use action distances for skill learning while the second uses it for learning goal conditioned policies. Although (Hartikainen et al., 2019) does discuss ways of incorporating human preferences to obtain the reward model there are several key distinctions. First, neither of the works provide a clear distinction between behavior-oriented and goal-oriented preferences. Second, both (Hartikainen et al., 2019) and (Venkattaramanujam et al., 2019) require goal proposals, that is they explicitly set states as goals either by using their learned distance measure or by asking humans to label possible goals. In contrast our method does not require explicit goals even though we attempt to learn a goal-oriented preference. Third, both the works learn an explicit distance function to approximate action distance. Our proposed solution instead shows that we can utlize the reward model's embedding space to compute the action distance. Fourth, (Hartikainen et al., 2019) uses the computed action distance directly as the reward value which makes it incompatible to use with other PbRL techniques like reward priors (Verma & Metcalf, 2022) etc. Finally, while (Venkattaramanujam et al., 2019) is interested in learning goal-conditioned policies, in this work, we explore the use of action distances on trajectory based binary preferences.

3. Methodology

In this section, we present our main contribution and discuss the key reasons why action distance measure is helpful for both reward learning and policy learning. Our main idea is to make the reward model being learned aware of the intended absorbing state or the probable goal according to the human preference (distance-to-goal) and the state space structure (i.e. reachability and environment dynamics). We do so by making the reward model solve the auxiliary task of predicting action distances between two states. We will first ground the action distance measure and propose the methodology for incorporating it with any existing PbRL framework that learns the reward model via function approximation and any underlying RL algorithm (both online and offline).

3.1. Action Distances

Definition 3.1. Average passage distance $m^\pi(s_j, a_j|s_i, a_i)$ is given by the expected number of actions required to go from s_i to s_j by taking a_i as the first action and choosing a_j as the final action upon reaching s_j.

$$m^\pi(s_j, a_j|s_i, a_i) = \mathop{\mathbb{E}}_{\tau\sim\pi}|\tau_{[(s_i,a_i)\cdots(s_j,a_j)]}| \tag{1}$$

Definition 3.2. Action distance, or commute distance, d^π between two states (s_i, s_j) and an initial action a_i, and final action a_j under some policy $\pi_\phi(s)$ and transition dynamics $\mathcal{T}(s, a, s')$ is given by the expected number of action steps taken to reach a state s_j from s_i with the first executed action as a_i and final chosen action a_j.

$$\begin{aligned} d^\pi(s_i, a_i, s_j, a_j) = \frac{1}{2}(&m^\pi(s_j, a_j|s_i, a_i) \\ &+ m^\pi(s_i, a_i|s_j, a_j)) \end{aligned} \tag{2}$$

For shorthand, in the context of distances between states, we abuse the notations for states s_i to mean (s_i, a_i) tuple, since we always compute distances in the embedding space for which the input was state-action tuple. Action distances (commute distance) have been largely used in developing theory for time-reversible Markov chains (Aldous & Fill, 1995). A variant of it was proposed in (Fouss et al., 2005) and more recently (Venkattaramanujam et al., 2019) used action distances to generate goals for learning goal-conditioned policies faster. Action distances have been used in variety of applications, but (Fouss et al., 2005) first showed that an embedding space that preserves commute distance must exist, however (Venkattaramanujam et al., 2019) admits obtaining action distance based ground truth targets can be challenging.

While incorporating action distance measure directly into the reward learning process by making the reward model $\hat{r}_t$ target to be a scaled linear combination of the existing reward model $\hat{r}_{t-1}$ and the action distance measure maybe possible, it becomes extrememly non-trivial and challenging for reasons like the scale of the distance measure and the predicted reward would have to be matched. A simpler approach would be to incorporate the action distance measure via mMDS by enforcing the embedding space of the reward model to be predictive of the action distance measure as

an auxiliary task. Not only such a method would not suffer from noise due to explicit goal-proposals (Hartikainen et al., 2019), but it also allows us to use it with other PbRL algorithms as well.

3.2. Action Distances can Improve Reward Learning

The reward learning and the policy learning happen in an iterative manner in our PbRL framework. While the key information regarding the human preference is contained in the binary feedback preference labels given by the human-in-the-loop, we posit that the policy trained over these learned rewards could still provide weak learning signal about the human preference. The policy function (π_ϕ) being trained takes into account the environment dynamics as well as the reward model (R_h) and can hint at the goal state (or atleast the set of states which could be potential goals under the reward model at the time). An adaptation of (Venkattaramanujam et al., 2019) would be to sample these potential goal states based on the action distance measure and try to improve reward model towards these sampled goals, however this requires strong assumptions like a simulator that can be reset and started from any given state. Moreover, such a strategy may be extremely hard to stabilize the learning as the intended goal state of the human is possibly partially known to the agent (because of limited preference queries) followed by setting hypothetical goals (sampled from the policy) to update the reward. Another direction could be to perform an Inverse RL or IRL over (Ng et al., 2000; Arora & Doshi, 2021; Ab Azar et al., 2020) step the learned policy to get another possible reward function, say, (r_{IRL}) and combine R_h and r_{IRL}. But such a method could be very expensive (as PbRL itself is a form of IRL with binary feedbacks over trajectory pairs) and the obtained r_{IRL} could be extremely noisy and may not offer any generalization to exisiting R_h. Another approach could be to learn the full world model (Hafner et al., 2019) and use the reward model to infer goal state (or trajectories leading to it). However, learning the world model may again be very expensive. An approximation to this approach could be to learn a forward dynamics model where an additional objective of the reward model is to predict the next state observation (given the current state and action). While this approach is more feasible than explicitly learning the world model and that such an auxiliary task has been used in the context of improving the policy learning (Nguyen et al., 2021; Zhang et al., 2018), our results demonstrate that this auxiliary task is a research challenge in itself and does not offer any performance improvements.

3.3. Accelerating Policy Learning via Action Distance

Definition 3.3. A Markov Decision Process $\{\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{R}, \gamma\}$ is called strong-reversible if for any action a_{ij} that allows the agent to transition from state s_i to s_j in a single step, there exists an action a_{ji} that allows the agent to transition from s_j to s_i in one-step.

Proposition 3.4. *For any MDP, $\{\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{R}, \gamma\}$, with $\forall s \sim \mathcal{S}, R(s) \leq -1, \gamma = 1$ and an absorbing state s_g, the average passage distance $m^\pi(s_g|s_i)$ from any state s_i to the goal state under a stochastic policy π is a pessimistic heuristic.*

Proposition 3.5. *For a strong-reversible MDP, $\{\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{R}, \gamma\}, \gamma = 1, \forall s \sim \mathcal{S}, R(s) \leq -1$ and an absorbing state s_g, the action distance $d^\pi(s_i, s_g)$ between any state s_i and the goal state under a stochastic policy π is a pessimistic heuristic.*

(Cheng et al., 2021) proved that pessimistic heuristics used with reward function are desirable to accelerate policy learning. This shows that for strong-reversible MDPs with absorbing states and at-least unit action costs, an action distance based heuristic is desirable for accelerating policy learning on the reward model. While we do not directly use the action distance for shaping the rewards from the reward model, the auxiliary task that requires the embedding space to preserve action distance does accelerate policy learning as confirmed empirically in Section 4.

3.4. Utilizing Action Distances for Reward Learning

The central idea is to create an auxiliary objective for the reward learning task where the reward model R_h is also predictive of the action distance between any two states. Under PbRL, the main reward learning objective is given by Equation 5. As shown by (Fouss et al., 2005), we know that an embedding space where the distance between the points (i.e. the embedding of state, action pair) is proportional to the action distance exists. Hence, we resort to performing a metric Multi-dimensional scaling (see Section A.2) using our action distance measure, i.e. ensure that the embedding space in the reward model $\hat{r}$ also reflects action distances.

Therefore, we ensure that the embedding space of the reward model, $R_e(s)$ of the state s, say the penultimate layer if R_h is a neural network, by ensuring that the Euclidean distance between the embedding of two states s_i and s_j reflects the action distance $d^\pi(s_i, s_j)$. This can be achieved by minimizing the Mean Squared Error (MSE) between the computed distance in the embedding space and the action distance as follows:

$$\mathcal{L}^{ad} = \mathop{\mathbb{E}}_{\substack{s_i,a_i \\ s_j,a_j \\ d_y \sim D_{ad}}} (||R_e(s_i, a_i) - R_e(s_j, a_j)||^2 - d_y)^2 \quad (3)$$

where, s_i, a_i, s_j, a_j are state action pairs in the dataset $D_{ad} = ((s_i, a_i), (s_j, a_j), d_y)$ which consists of the computed ground truth action distances between them as d_y.

3.4.1. Collecting Data for Action Distance Loss

Proposition 3.6. *For a stochastic policy π that induces a stationary distribution, under a balanced sampling from the*

dataset obtained by from Algorithm 1, a perfect function approximator can estimate the action distance between the states s_i and s_j.

We leverage the trajectory bank D_τ (agent's replay buffer) to create the dataset with action distance targets, D_{ad}. The key idea is that since the action distance ground truth that we want is an expectation over number of actions taken to reach s_j from s_i, we can approximate this action distance by sampling a state $s_i, s_j \in \tau$ where $j > i, \tau \in D_\tau$ and use the number of action steps taken in the trajectory from s_i to s_j as the ground truth distance $d_y = |j - i|$ as described in Algorithm 1 (see Appendix B). An important note is that the distances d_y in the dataset D_{ad} should be from the agent's current policy π_ϕ. For off-policy RL algorithms where the replay buffer, D_τ, would contain trajectories sampled from a stale policy, we emulate the required behavior of the dataset D_{ad} by ensuring that only the last k trajectories added to the dataset D_τ are used to compute D_{ad}. Practically, instead of picking n samples uniformly within a trajectory, we found it better to generate all combinations of (i, j) s.t. $i < j$ to populate the dataset. Refer to D.3 for the proof of 3.6.

3.4.2. Action Distance based PbRL Objective

We perform a linear combination of the proposed action distance based loss function $\mathcal{L}^{ad}$ the cross entropy loss L_{CE}, as in equation 5, (see Section A) to get our novel reward learning objective:

$$\mathcal{L}^{reward} = \lambda_{CE}\mathcal{L}^{CE}(D_h) + \lambda_{ad}\mathcal{L}^{ad}(D_{ad}) \qquad (4)$$

where $\mathcal{L}^{CE}$ is computed over D_h containing the queried trajectory pairs with human binary feedbacks (mean over the samples), and $\mathcal{L}^{ad}$ is computed over the dataset of state pairs with action distance targets D_{ad} created from the k most recent trajectories added to D_τ.

4. Empirical Evaluation

We design our experiments to investigate the following:

1. How do Action Distances improve the feedback sample efficiency compared to state of the art PbRL baselines? What is it's impact on policy learning?
2. How do Action Distances fare against PbRL baselines adapted to be aware of environment dynamics, or preserve the Eucledian distance between states?
3. How do the various factors like length of query trajectory, number of feedback queries and weight of the action distance loss affect the performance of the PbRL agent?
4. Does combining Action Distances with other Semi-Supervised approaches affect the overall performance?

To validate our proposed method, we conduct our experiments on six domains of Meta-World, namely, Hammer, Door-Open, Drawer-Open, Window-Open, Button-Press, Sweep-Into (Yu et al., 2020). We use the DMControl domains of Walker-Walk to answer the last question posed above. These domains have also been used in prior PbRL literature and the oracular reward setup assumes human preference to be behavior-oriented (that is the tacit task of walking). Following B-Pref (Lee et al., 2021b), we consider a scripted human in the loop (HiL) who provides a binary feedback label of their preference during the agent's training. Since the oracle uses the underlying task reward (henceforth, true reward model) to generate binary feedbacks, it allows us to evaluate the learned policy on the true reward. Finally, since the proposed method can be used with any existing PbRL technique, we use the state-of-the-art approach PEBBLE (Lee et al., 2021a) in our experiments as the backbone algorithm and call the combination of our reward learning objective in Equation 4 and PEBBLE as ADLoss (see Appendix E). The result plots show the mean (solid line) and standard deviation (shaded region) over five random seeds. Refer to Appendix for more details on domains F.6, experimental setup F and PEBBLE algorithm E.

State-of-the-art baselines: Since PEBBLE is the backbone algorithm used in the experiments for showing the benefits of action distance measure, we use PEBBLE as baseline. Additionally, we use the state-of-the-art PbRL algorithm SURF as our baseline. We use the version of SURF without temporal data augmentation, which helps in comparing the effects of semi-supervised learning by additional trajectory samples (SURF) and semi-supervised learning of action distances (ours). Finally, we use SAC trained on underlying oracle rewards as a loose upper-bound of policy performance. Figure 2 shows the performance of action distance auxiliary task (red) to be substantially and consistently better than baselines PEBBLE and SURF. Interestingly, for several domains (Hammer, Door-Open, Button-Press, Drawer-Open, Window-Open), our algorithm reaches performance very close to SAC (Haarnoja et al., 2018). As pointed out in 3.3, we also note that a typical pattern with ADLoss is an early peak towards higher return (indicative of accelerated policy learning).

Adapted PbRL Baselines: While we have discussed that action distance loss provides the reward model about valuable information like reachability, environment dynamics and distance to goal, part of these could also be given by other auxiliary tasks like forward dynamics prediction (Nguyen et al., 2021; Zhang et al., 2018) which were proposed in the context of RL for improved policy learning. We adapt the task of forward dynamics prediction by updating the reward model embedding to be predictive of the next state (for the given state-action pair). We refer to this as "*Rdynamics*".

Apart from action distances, Eucledian distances can also be preserved in the embedding space by the stress loss in mMDS (Equation 8). Hence, we create a baseline

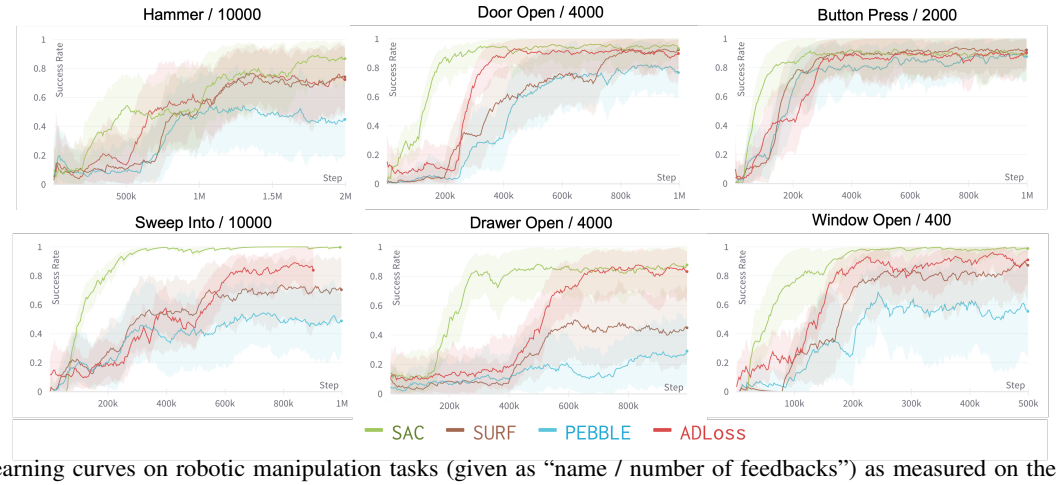


Figure 2. Learning curves on robotic manipulation tasks (given as "name / number of feedbacks") as measured on the success rate comparing ADLoss with PEBBLE, SURF and SAC.

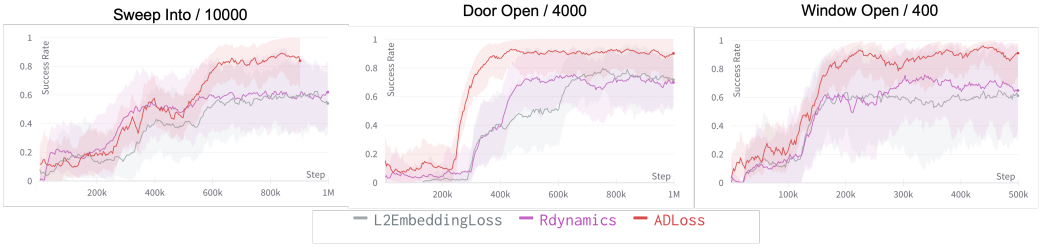


Figure 3. Learning curves on robotic manipulation tasks (given as "name / number of feedbacks") as measured on the success rate comparing ADLoss with adapted PbRL baselines Rdynamics and L2EmbeddingLoss.

"*L2EmbeddingLoss*" that uses L2 distance between the original state representations as the target in mMDS. Fig. 3 show that Rdynamics and L2EmbeddingLoss, although are a slight improvement over PEBBLE, are clearly weaker approaches than action distance based auxiliary task. Since it is known that learning world models can be a challenging (Ha & Schmidhuber, 2018) task in itself, then, even though Rdynamics captures information about the environment dynamics, incorporating it into PbRL is nontrivial. We find that L2EmbeddingLoss provides no additional useful inductive bias (for example goal-oriented nature of preference in our case, SSL over trajectories in SURF, etc.) to provide any basis for improvement, and performs the worst.

Additional Experiments: We conducted additional experiments to get more insignts about the effectiveness of ADLoss such as the effect of query segment length, effect of varying λ_{ad} the impact of ADLoss when used in conjunction with other SSL approaches. (see App. C).

5. Discussion

In this work, we present a Preference-based Reinforcement Learning algorithm for goal-oriented preferences like "reaching for a button to press it" or "reaching for a door to open it" that have an absorbing state. We note that many popular PbRL works have used the underlying task reward in robotic manipulation domains like Meta-World to emulate human preferences, and that these preferences are in fact goal-oriented. In the context of goal-oriented preferences we propose that the policy being learned along with the reward model can also provide valuable information like reachability, environment dynamics and distance-to-goal measures. We extract this learning signal via action distances, which is the expected number of actions taken under a policy to go from one state to another, and incorporate it into the reward learning objective by proposing an auxiliary objective for the reward model to be predictive of these action distances. This auxiliary objective ensures that the Eucledian distance between the embeddings (in the reward model) between two (state,action) tuples reflects the action distance. In addition to aiding the reward learning process, we show that action distance heuristics are pessimistic, and thus, can accelerate the policy learning. Our experiments on six Meta-World robotic manipulation tasks shows the effectiveness of our approach over several PbRL baselines.

Interesting future work includes combining action distance based auxiliary task with approaches that maybe specific to behavior-oriented preferences to create a comprehensive PbRL solution. Additionally, we believe action distances could be a key factor in pushing PbRL approaches to work on long horizon tasks with possible chained goals and would continue our research in that direction.

References

Ab Azar, N., Shahmansoorian, A., and Davoudi, M. From inverse optimal control to inverse reinforcement learning: A historical review. *Annual Reviews in Control*, 50:119–138, 2020.

Abel, D., Dabney, W., Harutyunyan, A., Ho, M. K., Littman, M., Precup, D., and Singh, S. On the expressivity of markov reward. *Advances in Neural Information Processing Systems*, 34:7799–7812, 2021.

Akrour, R., Schoenauer, M., and Sebag, M. Preference-based policy learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 12–27. Springer, 2011.

Aldous, D. and Fill, J. Reversible markov chains and random walks on graphs, 1995.

Arora, S. and Doshi, P. A survey of inverse reinforcement learning: Challenges, methods and progress. *Artificial Intelligence*, 297:103500, 2021.

Bakker, M. A., Chadwick, M. J., Sheahan, H. R., Tessler, M. H., Campbell-Gillingham, L., Balaguer, J., McAleese, N., Glaese, A., Aslanides, J., Botvinick, M. M., et al. Fine-tuning language models to find agreement among humans with diverse preferences. *arXiv preprint arXiv:2211.15006*, 2022.

Barto, A. G. Intrinsic motivation and reinforcement learning. *Intrinsically motivated learning in natural and artificial systems*, pp. 17–47, 2013.

Bejjani, W., Papallas, R., Leonetti, M., and Dogar, M. R. Planning with a receding horizon for manipulation in clutter using a learned value function. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pp. 1–9. IEEE, 2018.

Bewley, T. and Lecue, F. Interpretable preference-based reinforcement learning with tree-structured reward functions. *arXiv preprint arXiv:2112.11230*, 2021.

Borg, I. and Groenen, P. J. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

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.

Browne, C. B., Powley, E., Whitehouse, D., Lucas, S. M., Cowling, P. I., Rohlfshagen, P., Tavener, S., Perez, D., Samothrakis, S., and Colton, S. A survey of monte carlo tree search methods. *IEEE Transactions on Computational Intelligence and AI in games*, 4(1):1–43, 2012.

Cheng, C.-A., Kolobov, A., and Swaminathan, A. Heuristic-guided reinforcement learning. *Advances in Neural Information Processing Systems*, 34:13550–13563, 2021.

Chentanez, N., Barto, A., and Singh, S. Intrinsically motivated reinforcement learning. *Advances in neural information processing systems*, 17, 2004.

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

Devlin, S. M. and Kudenko, D. Dynamic potential-based reward shaping. In *Proceedings of the 11th international conference on autonomous agents and multiagent systems*, pp. 433–440. IFAAMAS, 2012.

Florensa, C., Held, D., Geng, X., and Abbeel, P. Automatic goal generation for reinforcement learning agents. In *International conference on machine learning*, pp. 1515–1528. PMLR, 2018.

Florensa, C., Degrave, J., Heess, N., Springenberg, J. T., and Riedmiller, M. Self-supervised learning of image embedding for continuous control. *arXiv preprint arXiv:1901.00943*, 2019.

Fouss, F., Pirotte, A., and Saerens, M. A novel way of computing similarities between nodes of a graph, with application to collaborative recommendation. In *The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*, pp. 550–556. IEEE, 2005.

Garcıa, J. and Fernández, F. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

Guan, L., Verma, M., Guo, S. S., Zhang, R., and Kambhampati, S. Widening the pipeline in human-guided reinforcement learning with explanation and context-aware data augmentation. *Advances in Neural Information Processing Systems*, 34:21885–21897, 2021.

Guan, L., Valmeekam, K., and Kambhampati, S. Relative behavioral attributes: Filling the gap between symbolic goal specification and reward learning from human preferences. *arXiv preprint arXiv:2210.15906*, 2022.

Ha, D. and Schmidhuber, J. Recurrent world models facilitate policy evolution. *Advances in neural information processing systems*, 31, 2018.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019.

Hart, P. E., Nilsson, N. J., and Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

Hartikainen, K., Geng, X., Haarnoja, T., and Levine, S. Dynamical distance learning for semi-supervised and unsupervised skill discovery. *arXiv preprint arXiv:1907.08225*, 2019.

Hoeller, D., Farshidian, F., and Hutter, M. Deep value model predictive control. In *Conference on Robot Learning*, pp. 990–1004. PMLR, 2020.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31, 2018a.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in atari. *Advances in neural information processing systems*, 31, 2018b.

Kambhampati, S., Sreedharan, S., Verma, M., Zha, Y., and Guan, L. Symbols as a lingua franca for bridging human-ai chasm for explainable and advisable ai systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 12262–12267, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Knox, W. B. and Stone, P. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the fifth international conference on Knowledge capture*, pp. 9–16, 2009.

Köster, R., McKee, K. R., Everett, R., Weidinger, L., Isaac, W. S., Hughes, E., Duéñez-Guzmán, E. A., Graepel, T., Botvinick, M., and Leibo, J. Z. Model-free conventions in multi-agent reinforcement learning with heterogeneous preferences. *arXiv preprint arXiv:2010.09054*, 2020.

Lee, K., Smith, L., and Abbeel, P. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. *arXiv preprint arXiv:2106.05091*, 2021a.

Lee, K., Smith, L., Dragan, A., and Abbeel, P. B-pref: Benchmarking preference-based reinforcement learning. *arXiv preprint arXiv:2111.03026*, 2021b.

Leike, J., Krueger, D., Everitt, T., Martic, M., Maini, V., and Legg, S. Scalable agent alignment via reward modeling: a research direction. *arXiv preprint arXiv:1811.07871*, 2018.

Li, P., Wang, Y., Wang, H., and Leskovec, J. Distance encoding: Design provably more powerful neural networks for graph representation learning. *Advances in Neural Information Processing Systems*, 33:4465–4478, 2020.

Liang, X., Shu, K., Lee, K., and Abbeel, P. Reward uncertainty for exploration in preference-based reinforcement learning. *arXiv preprint arXiv:2205.12401*, 2022.

Liu, M. and Chen, C. Task decoupling in preference-based reinforcement learning for personalized human-robot interaction. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 848–855. IEEE, 2022.

Liu, Y., Datta, G., Novoseller, E., and Brown, D. S. Efficient preference-based reinforcement learning using learned dynamics models. *arXiv preprint arXiv:2301.04741*, 2023.

Nair, A. V., Pong, V., Dalal, M., Bahl, S., Lin, S., and Levine, S. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems*, 31, 2018.

Ng, A. Y., Harada, D., and Russell, S. Policy invariance under reward transformations: Theory and application to reward shaping. In *Icml*, volume 99, pp. 278–287. Citeseer, 1999.

Ng, A. Y., Russell, S., et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 2, 2000.

Nguyen, T., Luu, T. M., Vu, T., and Yoo, C. D. Sample-efficient reinforcement learning representation learning with curiosity contrastive forward dynamics model. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3471–3477. IEEE, 2021.

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. *arXiv preprint arXiv:2203.10050*, 2022.

Pong, V. H., Dalal, M., Lin, S., Nair, A., Bahl, S., and Levine, S. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

Roh, B., Shin, W., Kim, I., and Kim, S. Spatially consistent representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1144–1153, 2021.

Schmidhuber, J. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE transactions on autonomous mental development*, 2(3):230–247, 2010.

Shen, J., Qu, Y., Zhang, W., and Yu, Y. Wasserstein distance guided representation learning for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Soni, U., Sreedharan, S., Verma, M., Guan, L., Marquez, M., and Kambhampati, S. Towards customizable reinforcement learning agents: Enabling preference specification through online vocabulary expansion. *arXiv preprint arXiv:2210.15096*, 2022.

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

Tunyasuvunakool, S., Muldal, A., Doron, Y., Liu, S., Bohez, S., Merel, J., Erez, T., Lillicrap, T., Heess, N., and Tassa, Y. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

Venkattaramanujam, S., Crawford, E., Doan, T., and Precup, D. Self-supervised learning of distance functions for goal-conditioned reinforcement learning. *arXiv preprint arXiv:1907.02998*, 2019.

Verma, M. and Metcalf, K. Symbol guided hindsight priors for reward learning from human preferences. *arXiv preprint arXiv:2210.09151*, 2022.

Wagener, N. C., Boots, B., and Cheng, C.-A. Safe reinforcement learning using advantage-based intervention. In *International Conference on Machine Learning*, pp. 10630–10640. PMLR, 2021.

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

Wirth, C., Akrour, R., Neumann, G., Fürnkranz, J., et al. A survey of preference-based reinforcement learning methods. *Journal of Machine Learning Research*, 18(136): 1–46, 2017.

Young, F. W. and Hamer, R. M. *Multidimensional scaling: History, theory, and applications*. Psychology Press, 2013.

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*, pp. 1094–1100. PMLR, 2020.

Zhang, A., Satija, H., and Pineau, J. Decoupling dynamics and reward for transfer learning. *arXiv preprint arXiv:1804.10689*, 2018.

Zhang, G. and Kashima, H. Learning state importance for preference-based reinforcement learning. *Machine Learning*, pp. 1–17, 2023.

Zhong, M., Johnson, M., Tassa, Y., Erez, T., and Todorov, E. Value function approximation and model predictive control. In *2013 IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*, pp. 100–107. IEEE, 2013.

A. Preliminaries

A.1. Preference-based Reinforcement Learning

Reinforcement learning allows for agents interacting in an environment $\mathcal{E}$ where at each discrete time-step t, the agent receives an observation o_t from the environment and chooses an action a_t based on its policy π. As in conventional RL frameworks, we assume that the underlying system is a Markov Decision Process, i.e. the tuple $< \mathcal{S}, \mathcal{T}, \mathcal{A}, \tilde{\mathcal{R}}_h, \gamma >$ describing the state space $\mathcal{S}$, agent's action space $\mathcal{A}$, the underlying environment transition dynamics $\mathcal{T}$, the discount factor γ where the agent's goal is to maximize the return $\sum_{k=0}^{\infty} \gamma^k \tilde{\mathcal{R}}_h(s_{t+k}, a_{t+k})$ computed over the reward system $\mathcal{R}_h$ in concern. In the PbRL setup that we are interested in, the goal of the agent is twofold: (1) to infer the human's underlying reward model $\tilde{\mathcal{R}}_h$ via binary feedback over trajectory pairs, and (2) to further use the learned reward model R_h to compute a policy π_ϕ parameterized by ϕ to maximize discounted cumulative return over R_h.

We utilize the formulation presented in (Wilson et al., 2012) for the Preference-based Reinforcement Learning problem where the agent queries the human in the loop with a trajectory pair (τ_0, τ_1), where, $\tau_i = \{(s_k, a_k), (s_{k+1}, a_{k+1} \cdots (s_{k+H}, a_{k+H}))\}$, for a binary feedback $y \in \{0, 1\}$ indicating their preferred trajectory. Such feedbacks along with the queried trajectories are stored in a dataset D_τ as tuples (τ_0, τ_1, y). Following the Bradley Terry model (Bradley & Terry, 1952) to compute the probability of one trajectory being preferred over another, recent line of works like (Lee et al., 2021a; Christiano et al., 2017) approximates the human reward function as R_h, parameterized by, say, ψ, by solving a supervised learning problem where the returns computed over the learned reward function are higher for trajectories that were preferred by the human in the loop than the returns computed on the non-preferred trajectory. This is done by minimizing the cross-entropy between the predictions and ground truth human labels as follows:

$$\begin{aligned} \mathcal{L}_{CE} = - \mathop{\mathbb{E}}_{(\tau_0,\tau_1,y)\sim\mathcal{D}} & [y(0) \log P_\psi[\tau_0 \succ \tau_1] \\ & + y(1) \log P_\psi[\tau_1 \succ \tau_0]] \end{aligned} \tag{5}$$

where probabilities P_ψ are computed using the approximated reward function R_h as:

$$P_\psi[\tau_0 \succ \tau_1] = \frac{\exp(\sum_t R_h(s_t^0, a_t^0))}{\sum_{i\in\{0,1\}} \exp(\sum_t R_h(s_t^i, a_t^i))} \tag{6}$$

A.2. Multi-dimensional Scaling

Multi-dimensional Scaling (MDS) (Borg & Groenen, 2005; Young & Hamer, 2013) is a form of non-linear dimensional reduction where dissimilarities between pairs of the data in the original space are mapped to distances and are preserved in the low-dimension space. While MDS has typically been used to visualize similarity or dissimilarity between a set of objects in a low-dimensional space, it has also been used to construct embedding space for a set of objects by finding a set of coordinates in the low-dimensional space that minimize the difference between the distances between the objects in the original high-dimensional space and the distances between the objects in the low-dimensional space.

Classical MDS (or Toegerson-Gower Scaling, or Principal Coordinates Analysis, or cMDS) assumed that the dissimilarities in the original dimensionality are in the Euclidean space and therefore, algorithms for classical MDS preserve the input dissimilarities when these dissimilarities are Euclidean distances. Say the dissimilarity for objects i, j is given by $\delta_{i,j}$, and the embedding space is given by E. Then, cMDS would reduce a loss (also called stress), as follows:

$$\sigma_{cMDS}(E) = \sum_{i<j} w_{ij}(d_{ij}(E) - \delta_{ij})^2 \tag{7}$$

In this work we use a generalization of classical MDS, called Metric MDS (mMDS) where in the stress Equation 7, the distance metric and dissimilarity measures are replaced by $f(x)$ to give the following stress objective:

$$\sigma_{mMDS}(E) = \sum_{i<j} w_{ij}(f(d_{ij}(E)) - f(\delta_{ij}))^2 \tag{8}$$

We note that, as pointed out in (Venkattaramanujam et al., 2019) mMDS does not admit an analytical solution, instead it is solved iteratively and convergence to a global minimum is not guaranteed.

B. Dataset for Action Distance Loss Update

Algorithm 1 Dataset for Action Distance Loss Update

Input: D_τ, Recent window k, Samples per trajectory n
$D_\tau^k = D_\tau[k \cdots N]$ {N is size of D_τ}
Initialize $D_{ad} \leftarrow \varnothing$
for $ix = 0$ **to** $N - k$ **do**
  for $iy = 0$ **to** n **do**
    Uniformly choose i, j s.t. $i < j \leq T$ {T} is length of trajectory τ
    $x, y = ((s_i^\tau, a_i^\tau), (s_j^\tau, a_j^\tau)), |j - i|$
    $D_{ad} = D_{ad} \cup (x, y)$
  end for
end for
return D_{ad}

C. Additional Experiments

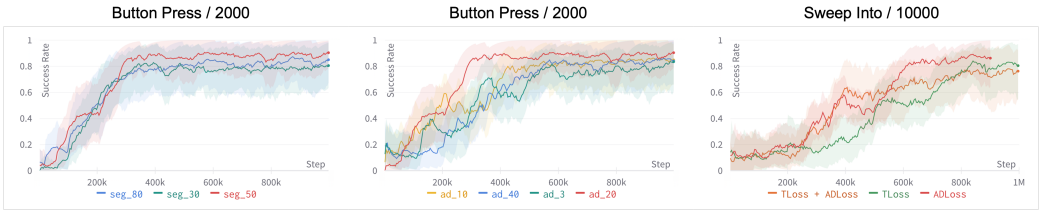


Figure 4. Ablation study on Button-Press (2000 feedbacks). (left) Effect of query segment length on the ADLoss agent performance for lengths $\{30, 50, 80\}$. (center) Effect of varying λ_{ad} in equation 4 in the set $\{3, 10, 20, 40\}$. (right) Additional experiment on Sweep-Into (10000 feedbacks) to evaluate how does combining action distances with other SSL approaches may impact PbRL performance.

To get more insights about the effectiveness of ADLoss, we conducted additional experiments on Meta-World-Button-Press, Sweep-Into and DMControl Walker-Walk. From Fig. 4 (left), we find that minor change to the length of the query segment dose not greatly impact the agent's performance. This is important as although the given query sizes are inconsequential with respect to compute, these size differences can have a huge cognitive impact on the human in the loop. Fig. 4 (center) tests ADLoss performance at different λ_{ad} values in the ADLoss reward learning objective shown in Equation 4. Similar to (Park et al., 2022), we find that tuning this hyperparameter has the most impact on the performance. Next, we test our ADLoss in conjuction with a SSL techqniue (proposed in Appendix I) to study how compatible is ADLoss to other PbRL approaches. As discussed in section 3.3, we find that the SSL approach (TLoss), when used in conjunction with ADLoss benefits from accelerated policy learning. Finally, we performed experiments with DMControl Walker-Walk to realize whether ADLoss has any negative impact on behavior-oriented preferences, We find (Appendix H) that ADLoss performance is comparable to SURF which is a positive indicator for future work for hybrid PbRL algorithms using action distances for behavior-oriented preferences.

D. Proofs

D.1. Proof for Proposition 3.4

Statement: *For any MDP, $\{\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{R}, \gamma\}$, with $\forall s \sim \mathcal{S}, R(s) \leq -1, \gamma = 1$ and an absorbing state s_g, the average passage distance $m^\pi(s_g|s_i)$ from any state s_i to the goal state under a stochastic policy π is a pessimistic heuristic.*

Proof Sketch: As defined by (Cheng et al., 2021), a pessimistic heuristic is one that overestimates the cost to goal (or underestimates the reward). Proving Proposition 3.4 is straightforward, since the average passage distance, by definition, gives an estimate of the number of actions required to reach the goal, which is an overestimate of minimum cost-to-go from that state to the goal.

D.2. Proof for Proposition 3.5

Statement: *For a strong-reversible MDP,* $\{\mathcal{S}, \mathcal{T}, \mathcal{A}, \mathcal{R}, \gamma\}, \gamma = 1, \forall s \sim \mathcal{S}, R(s) \leq -1$ *and an absorbing state* s_g*, the action distance* $d^\pi(s_i, s_g)$ *between any state* s_i *and the goal state under a stochastic policy* π *is a pessimistic heuristic.*

Proof Sketch: To prove, we must show that the commute distance from any state s_i to any state s_j is an overestimate of the shortest distance from s_i to s_j. Since it is a strong-reversible MDP, the shortest distance from s_i to s_j, say d^*, is the same as shortest distance from s_j to s_i. From proposition 3.4, $m^\pi(s_i|s_j) \geq d^*$ and $m^\pi(s_j|s_i) \geq d^*$. Hence, $d^\pi(s_i, s_j) = \frac{1}{2}(m^\pi(s_i|s_j) + m^\pi(s_j|s_i)) \geq d^*$, is a pessimistic heuristic.

D.3. Proof for Proposition 3.6

Statement: *For a stochastic policy* π *that induces a stationary distribution, under a balanced sampling from the dataset obtained by from Algorithm 1, a perfect function approximator can estimate the action distance between the states* s_i *and* s_j*.*

Proof Sketch: Algorithm 1, in the limit, can generate infinitely many samples for action distance targets from a state i to j, and j to i. This is easy to realize as the dataset is constructed by uniformly picking s_i, s_j pairs from trajectories sampled via the policy π. As the action distance estimate is the average of the distance from i to j and back. It is possible that the number of data points sampled from $i \rightarrow j$ are considerably more than $j \rightarrow i$ in which case even a perfect approximator under MLE (Maximum Likelihood Estimation) assumption would predict the mean of the observed samples as the action distance. The mean of the observed samples is guaranteed to be equal to the commute distance or the action distance if the observed samples along each direction $i \rightarrow j$ and $j \rightarrow i$ are balanced (Venkattaramanujam et al., 2019).

E. PbRL Algorithm

We present our PbRL algorithm, as shown in (Lee et al., 2021a), which uses the PEBBLE as a backbone. The integration of action distance loss into the PbRL algorithm requires no change to the model architectures or the learning paradigm and can be easily done so by updating the lines in red in Algorithm 2. SURF updates the same parts of the pseudocode as ours where the SSL (semi-supervised learning) approach in SURF integrates in to the $\mathcal{L}^{Reward}$ and the SSL step populates the feedback buffer with pseudo-labels.

E.1. PEBBLE Algorithm

PEBBLE is a PbRL algorithm that comprises of two key elements: pre-training and relabeling experience buffer. To gather a wide range of experiences, PEBBLE starts by using intrinsic motivation (Chentanez et al., 2004; Barto, 2013; Abel et al., 2021; Schmidhuber, 2010) to pre-train the policy, which optimizes the policy to increase the state entropy in order to explore the environment effectively. Afterwards, PEBBLE uses the SAC algorithm, a state-of-the-art off-policy RL algorithm, to further train the policy. To ensure stability in the learning process, PEBBLE relabels all experiences in the buffer when the reward model is updated.

Algorithm 2 Integrating ADLoss into PEBBLE

Input: feedback frequency K, # queries per feedback session M
Initialize parameters of Q_θ and $\hat{r}_\psi$
Initialize a dataset of preferences $D_h \leftarrow \varnothing$
// EXPLORATION PHASE
$D_\tau, \pi_\phi \leftarrow$ `EXPLORE()` in (Lee et al., 2021a)
POLICY LEARNING PHASE
for each iteration **do**
  // REWARD LEARNING PHASE
  if iteration % K == 0 **then**
    for m in 1...M **do**
      $(\sigma^0, \sigma^1) \sim$ `SAMPLE()` in (Lee et al., 2021a)
      Query instructor for y
    end for
    for each gradient step **do**
      Sample minibatch $\{(\sigma^0, \sigma^1, y)_j\}_{j=1}^{D_h} \sim D_h$
      Perform Semi-Supervised Learning as in Algorithm 1
      Optimize $\mathcal{L}^{Reward}$ w.r.t. ψ
    end for
    Relabel entire replay buffer D_τ using $\hat{r}_\psi$
  end if
  REINFORCEMENT LEARNING PHASE
  for each time-step t **do**
    Collect s_{t+1} by taking $a_t \sim \pi_\phi(a_t|s_t)$
    Store transitions $D_\tau \leftarrow D_\tau\{(s_t, a_t, s_{t+1}, \hat{r}_\psi(s_t))\}$
  end for
  for each gradient step **do**
    Sample random minibatch $\{\tau_j\}_{j=1}^{D_\tau} \sim D_\tau$
    Optimize L_{critic}^{SAC} and L_{actor}^{SAC} w.r.t. θ and ϕ, respectively, as in (Lee et al., 2021a)
  end for
end for

F. Experiment Details

Unless stated otherwise, we have attempted to keep all the hyperparameters & experiments settings as close to that proposed in prior works (Park et al., 2022; Lee et al., 2021a).

F.1. State Space and Action Space

We use the available Meta-world package (Yu et al., 2020) to instantiate our environments. We use the default state space representation given by the package, that contains information about the Cartesian position of the end-effector, positions of objects of concern etc. The details about the action space and the observation space are given in (Yu et al., 2020).

F.2. Reward Architecture and Embedding Space

Following the implementation of (Lee et al., 2021a; Park et al., 2022) we implement reward model via a neural network and bound the final output using a $\tanh$ activation function : [-1, 1]. For all the Meta-world experiments the reward model has three hidden layers with 256 neurons each followed by the prediction layer (with one neuron). The embedding space used for minimizing the metric Multidimensional Scaling stress or the derived ADLoss Mean squared error is the penultimate layer of the network. We use the ADAM optimizer for training the SAC actor-critic as well as the reward model. The hyperparameters used for baseline PEBBLE, SURF and ADLoss (our action distance based auxiliary task loss) are given in tables 3, 4, and 5.

The input to the reward model is (state, action) tuple. As used previously (Park et al., 2022; Lee et al., 2021b), we stack the

state and action vectors and treat them as a single input for the reward model.

F.3. Oracle

While B-Pref (Lee et al., 2021b) explores various types of scripted humans in the loop like myopic, noisy etc, since our primary objective in this work is to evaluate the effectiveness of Action Distance measure for PbRL, we assume an oracle scripted human who uses the underlying reward to correctly provide the binary feedback. The feedback is given as follows :

$$y(\tau_0, \tau_1) = \begin{cases} 0, & \text{if} \quad \sum_i \tilde{R}_h(\tau_0) > \sum_i \tilde{R}_h(\tau_1) \\ 1, & \text{if} \quad \sum_i \tilde{R}_h(\tau_0) < \sum_i \tilde{R}_h(\tau_1) \end{cases} \tag{9}$$

where, $\tilde{R}_h$ is the environment reward being used as the human preference reward function and τ_0, τ_1 are the queried trajectory pairs. Note that we work under the setting that the preference feedback is binary and therefore if the trajectory returns are equal we uniformly pick a preferred trajectory. This does not pose any problems with our chosen benchmark domains as the underlying reward is dense and shaped (Devlin & Kudenko, 2012; Ng et al., 1999).

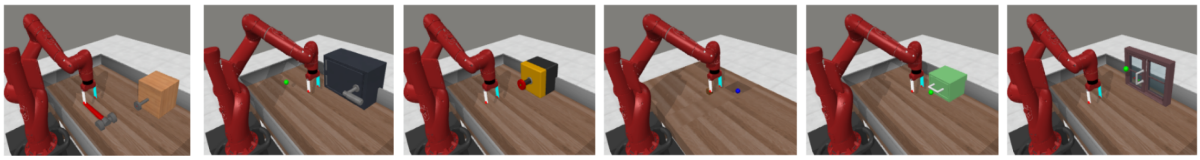
F.4. Sampling Schemes

We refer the readers to (Lee et al., 2021a; Christiano et al., 2017) for the various sampling schemes that have been proposed in prior work. In this work, for all the experiments we use the disagreement based sampling for selecting pair of trajectories to query to the human in the loop.

F.5. Implementation, Code and Compute

We use the publicly available implementation of B-Pref (Lee et al., 2021b) for the implementation of PEBBLE and SAC. We implement the remaining baselines, SURF, Rdynamics, L2EmbeddingLoss (code in supplementary). All the experiments were run on an Intel(R) Xeon(R) Gold 6258R CPU @ 2.70GHz, with Quadro RTX 8000 GPU.

F.6. Evaluation Domains



(a) Hammer (b) Door Open (c) Button Press (d) Sweep Into (e) Drawer Open (f) Window Open

Figure 5. Rendered images of the Meta-world (Yu et al., 2020) evaluation domains.

Figure 6. Rendered image of DMControl Suite (Tunyasuvunakool et al., 2020) - Walker Walk - a benchmark for behavior-oriented preferences.

We present here in Table 1, the description of the Meta-World domains that we have used to show our empirical evaluations, along with the respective environment rewards in Table 2 as have been specified in (Yu et al., 2020). Effectively, all the specified rewards are goal-reaching distance-to-go and therefore an oracle over these rewards has goal-oriented preferences.

Table 1. Meta-World domain descriptions, as in (Yu et al., 2020).

Task	**Description**
Hammer	Hammer a screw on the wall. Randomize the hammer and the screw positions.
Door Open	Open a door with a revolving joint. Randomize door positions.
Button Press	Press a button. Randomize button positions.
Sweep Into	Sweep a puck into a hole. Randomize puck positions.
Drawer Open	Open a drawer. Randomize drawer positions.
Window Open	Push and open a window. Randomize window positions.
Unlock Door	Unlock the door by rotating the lock counter-clockwise. Randomize door positions.
Plate Slide	Slide a plate into a cabinet. Randomize the plate and cabinet positions.

Table 2. Meta-World domain rewards, as in (Yu et al., 2020).

Task	**Description**
Hammer	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 100 \cdot \min\{o_z, z_{target}\} + I_{\lvert o_z - z_{target}\rvert<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$
Door Open	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$
Button Press	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$
Sweep Into	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$
Drawer Open	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$
Window Open	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$
Unlock Door	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$
Plate Slide	$-\parallel h-o \parallel_2 + I_{\parallel h-o\parallel_2<0.05} \cdot 1000 \cdot \exp\{\parallel h-g \parallel_2^2 /0.01\}$

G. Hyperparameters

Table 3. Hyperparameters of backbone PEBBLE in our experiments.

Hyperparameter	Value	Hyperparameter	Value
Initial temperature	0.1	Hidden units per each layer	1024(DMControl), 256(Meta-world)
Length of segment	50	# of layers	2(DMControl), 3(Meta-world)
Learning rate	0.0003 (Meta-world) 0.0005 (Walker) 0.0001 (Quadruped)	Batch Size Optimizer	1024(DMControl), 512(Meta-world) Adam (Kingma & Ba, 2014)
Critic target update freq	2	Critic EMA τ	0.005
β_1, β_2	(0.9, 0.999)	Discount γ	0.99
Frequency of feedback	5000 (Meta-world) 20000 (Walker) 30000 (Quadruped)	Maximum budget / # of queries per session	1000/100, 100/10(DMControl) 10000/50, 4000/20(Meta-world) 2000/25, 400/10 (Meta-world)
# of ensemble models N_{en}	3	# of pre-training steps	10000

Table 4. Hyperparameters of SURF.

Hyperparameter	Value
Unlabeled batch ratio μ	4
Threshold τ	0.95
Loss weight λ	1

Table 5. Additional hyperparameters used in our experiments.

Hyperparameter	Value
Recent trajectories k	5
Adloss weight	20

H. Additional Experiment Details

H.1. Setup for Rdynamics baseline

The Rdynamics baseline requires modification to the reward model. While imposing assumptions that one can easily modify the model architecture should be avoided, predicting forward state dynamics has been used in past RL literature and has been shown to be useful for policy learning. Just as we use the penultimate reward model layer as the embedding space, we attach another layer that predicts the next state and add a forward state prediction loss. Fig. 7 illustrates the reward architecture used for Rdynamics baseline and below is the additional loss used for forward state prediction, where y is the predicted next state. A linear combination of $\mathcal{L}_{forwarddynamics}$ and $\mathcal{L}_{CE}$ (similar to ours, in equation 4) gives the reward learning objective for rdynamics.

$$\mathcal{L}_{forwarddynamics} = MSE(y, s_{t+1}) \tag{10}$$

H.2. Setup for L2EmbeddingLoss baseline

L2EmbeddingLoss uses the same $\mathcal{L}^{Reward}$ objective as in Equation 4, except that in Equation 3 the targets d_y are set to be, $||s_j - s_i||^2$, i.e., the L2 norm of the difference between the two states.

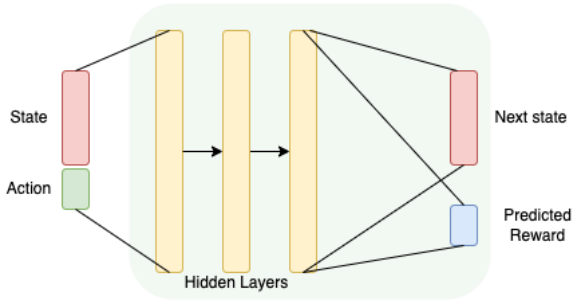


Figure 7. Overview of the reward model architecture for Rdynamics.

H.3. Using ADLoss for Behvaior Oriented tasks

While the use of action distances has been motivated in the context of goal-oriented preferences, we performed an additional experiment to test whether ADLoss has any adverse effects for preferences on behavior-oriented tasks. We test our proposed method on Walker-Walk, a popular DMControl benchmark for PbRL, and find that with just 100 feedback queries it is able to perform similar to SURF (state of the art). Without drawing any scientific conclusions, this experiment atleast motivates us to continue to analyze the impact of action distance measure in PbRL in behavior-oriented preferences.

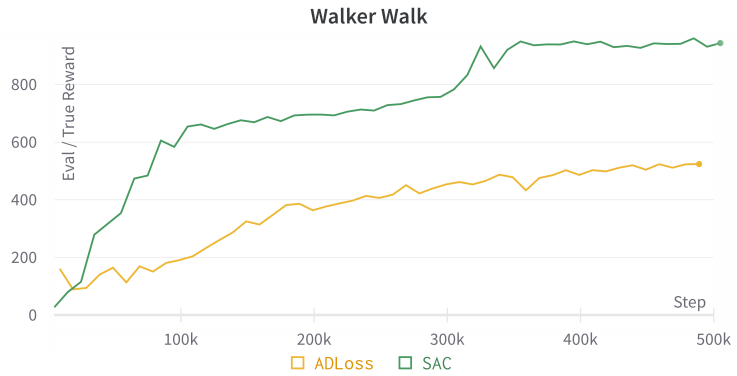


Figure 8. Performance of the proposed action distance based auxiliary task (with backbone PEBBLE) on the behavior oriented task of Walker-Walk in the DMControl suite using 100 feedbacks.

I. Triplet Loss based Semi-Supervised Learning objective

While we have shown the benefits of ADLoss in section 4, we wanted to test how would ADLoss perform in conjunction with another SSL approach.

In pursuit of proposing a comprehensive, yet complementary solution for PbRL, we present a novel semi-supervised learning objective (SSL) that utilizes pseudo-labelling for unlabelled trajectories followed by a triplet loss minimization that can be used in conjunction with the proposed Action Distance Auxiliary task. The proposed combination is to explore one of the tasks we mention as the future work, which is to find a more comprehensive solution for PbRL that can boost agent's performance for both behavior-oriented and goal-oriented preference. While ADLoss takes care of teh goal-oriented parts of a specified preference we look for a technique that can not only help with behavior-oriented parts of the preference but also gels well with ADLoss.

In our limited testing we have found that the combination of ADLoss and the following triplet loss SSL outperforms the PEBBLE / SURF baselines and is complementary across several domains (for DMControl - capturing behavior preferences and Metaworld - capturing goal oriented preferences).

This SSL objective is operationalized by a triplet loss that requires a specified anchor data point, and positive and negative samples. It then ensures that there exists atleast some margin gap between the +sample-anchor distance and -vesample-anchor distance. We first set the sampled trajectory from the unlabelled dataset as our anchor. The minibatch of trajectory pairs sampled from the human feedback replay buffer will serve as positive or negative samples. To do so, we make an Absolute Preference over Minibatch assumption as,

Assumption I.1. In a minibatch of k samples from human feedback buffer as triplets of $(\tau_0, \tau_1, y)_{i=1}^{i=k}$, the preference label can be treated as an absolute preference about the trajectory, i.e. if $y = 1$ then trajectory is a preferred trajectory (not relative to other trajectories).

Assumption I.1 enables the creation of a bipartite set of trajectories, with set g containing all preferred trajectories and set b containing all dispreferred trajectories. If the queries are made from far enough regions of the state space, the assumption holds well, but if the batch size is too large, it is possible to encounter situations where the assumption does not hold. Because of this, we limit the batch size to be in the set $\{8, 16, 32\}$.

We perform a pseudo-labelling step where we identify whether the sampled trajectory τ is closer to the trajectories in set **g** or **b** where the label is:

$$\mathbf{y} = \begin{cases} \mathbf{g} & ; d(\mathcal{R}(\tau), \mathop{\mathbb{E}}_{g\sim\mathbf{g}}[\mathcal{R}(\tau_g)]) < d(\mathcal{R}(\tau), \mathop{\mathbb{E}}_{b\sim\mathbf{b}}[\mathcal{R}(\tau_b)]) \\ \mathbf{b} & ; \text{otherwise} \end{cases} \tag{11}$$

where d is the L_2 distance function. We use the set **y** as the positive set of data points and **1-y** as the negative set for our triplet loss,

$$\begin{aligned} \mathcal{L}^t(\tau) = max(0, ||\mathcal{R}(\tau) - \mathop{\mathbb{E}}_{y\sim\mathbf{y}}[\mathcal{R}(\tau_y)])||^2 \\ -||\mathcal{R}(\tau) - \mathop{\mathbb{E}}_{\tilde{y}\sim\mathbf{1\text{-}y}}[\mathcal{R}(\tau_{\tilde{y}})])||^2 + m) \end{aligned} \tag{12}$$

where m is the margin hyperparameter. We overload the notation for reward to reflect the rewards for the trajectory states as a vector, i.e., $\mathcal{R}(\tau) = \begin{bmatrix} R(s_0) & R(s_1) & \cdots & R(s_{T-1}) & R(s_T) \end{bmatrix}^T$.

Figure 4 Sweep-Into shows results of using TLoss, ADLoss and Cross Entropy together and ablates against each component. We find that while the performance of the agent (ADLoss + TLoss) is quite close to the best performer, the addition of action distance based loss to the triplet loss does accelerate policy learning - a feature which was theoretically motivated for action distances in Section 3.3.