Few-Shot Preference Learning for Human-in-the-Loop RL

Anonymous Author(s)
Affiliation
Address
email

Abstract: While reinforcement learning (RL) has become a more popular approach for robotics, designing sufficiently informative reward functions for complex tasks has proven to be extremely difficult due their inability to capture human intent and policy exploitation. Preference based RL algorithms seek to overcome these challenges by directly learning reward functions from human feedback. Unfortunately, prior work either requires an inordinate number of queries implausible for any human to answer or unduly restricts the class of reward functions to guarantee the elicitation of the most informative queries, resulting in models that are insufficiently expressive for realistic robotics tasks. Contrary to most works that focus on query selection to *minimize* the amount of data required for learning reward functions, we take an opposite approach: *expanding* the pool of available data by viewing human-in-the-loop RL through the more flexible lens of multi-task learning. Motivated by the success of meta-learning, we pre-train preference models on prior task data and quickly adapt them for new tasks using only a handful of queries. Empirically, we reduce the amount of online feedback needed to train manipulation policies in Meta-World by 20X, and demonstrate the effectiveness of our method on a real Franka Panda Robot. Moreover, this reduction in query-complexity allows us to train robot policies from actual human users. Videos of our results can be found at https://sites.google.com/view/few-shot-preference-rl/home.

Keywords: Preference Learning, Interactive Learning, Multi-task Learning

1 Introduction

The success of Deep Reinforcement Learning (RL) methods in game-playing and simulated domains [1] has galvanized recent work applying RL-based techniques to real-world robot control to middling success. Integral to the success of deep RL methods is the reward function, which describes the desired behavior of the learning agent. While training robots via trial and error holds great promise, designing suitable reward functions remains challenging. For example, consider teaching a robot to open a door. The simplest reward function would be sparse – providing the robot with a positive reward only when the door has been opened. However, such sparse signals offer little learning signal, hampering exploration and enlarging sampling complexity. Conversely in designing a dense reward function, practitioners are tasked with summarizing multiple objectives like door angle or proximity to the handle into a single scalar. Such reward functions have proven to be difficult to design [2] and can even cause agents to learn unintended behaviors. Hand-designed dense reward functions often do not directly parallel the goal-conditions they aim to capture. This causes RL agents to exploit them, leading to potentially hazardous policies that do not align with human intent [3]. All of these problems are exacerbated in more realistic, multi-task scenarios where we wish to teach agents how to complete a variety of tasks in their environment. A robot that can only open doors provides little

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

utility in the real world. Given the effort required to design a single reward function, constructing reward functions for an entire family of tasks is impractical.

Recent works attempt to circumvent the basic challenges of reward design by learning reward functions directly from human preferences. This paradigm has numerous advantages: learned reward functions are dense, directly aligned with human intent, and easily adjusted. While demonstrations are often difficult to provide due to expensive data collection [4] and large domain gaps [5, 6], human preferences can often be elicited solely through simple pairwise comparisons. However, given the large continuous state and action settings of robotics problems, learning a high-performance reward function from only a handful of noisy user generated binary labels seems hopeless. Consequently, methods from active learning maximize feedback efficiency by attempting to ask the most informative queries with simplistic or linear reward models. The constraints these methods place on the reward function class make them unable to scale to complex domains that necessitate expressive reward models. Moreover, such methods are not significantly more data efficient than random sampling in practice [7, 8]. On the other hand, recent works using general function approximators still require thousands to tens-of-thousands of artificially labeled queries to learn sufficiently accurate reward functions [9, 10, 11]. This is far to onerous for real human labelers to provide, even in the single task setting. In order to train effective reward functions from actual humans, we need need a paradigm shift. Instead of optimizing for the most informative query, we take an orthogonal perspective that maximizes the amount of overall data by leveraging pre-training on realistic multi-task settings, and fine-tuning on a small and manageable amount of human queries online.

In the multi-task setting, significantly more data is available from previously known tasks which can be used to accelerate reward function learning. In fact, the shared structure of many real-world tasks has already been shown to accelerate policy learning [12]. The same structure can be exploited to learn complex reward functions for new tasks with only a handful of queries. This is largely because most tasks have rewards that are non-trivial compositions of other tasks. For example, data collected on opening windows and drawers could help us learn a reward function for door-opening with fewer human queries. Our key insight is to use multi-task data in order to meta-learn reward functions for preference based RL. Pre-training reward functions on a large dataset enables them to quickly adapt to new preferences with only a handful of queries.

Our core contributions are as follows. First we introduce a method for efficiently training RL policies from human-feedback using a meta-learned reward function. Second, we demonstrate its effectiveness across a number of standard robotics benchmarks, reducing query usage by a factor of 20 on robotic benchmarks in comparison to previous state-of-the-art methods. This increase in efficiency allows us to learn manipulation policies from real human feedback unlike prior work. Finally, we demonstrate the effectiveness of our method in the real-world using a Franka Panda robot.

2 Related Work

Our work builds on top of a number of prior works spanning RL, preference-learning, and meta-learning. Here we review the areas most relevant to our method.

Reward Learning. As hand-designed reward functions are difficult to tune, easily mis-specified [3, 13], and challenging to implement in the real world [2, 14], many recent works have leveraged human-collected data in order to learn reward functions. A large body of work focuses on using inverse RL, where a reward function is learned from approximately expert human collected demonstrations [15, 16, 17, 18]. However, demonstration collection is often expensive [19, 4, 20, 5, 21] and collected demonstrations are sometimes not even aligned with true human preferences [22, 23, 24]. Alternative strategies for learning reward functions utilize physical corrections [25], natural language instructions [26], human-provided scalar scores [27, 28] or partial [29] or complete [30**?** , 31] rankings . While physical corrections and language may be easier for the user, it is generally unclear how they translate to reward updates. Stronger signals are provided by scalar scores or multiple rankings, but they are harder for users to provide. We thus use pairwise comparisons as they are the simplest and generally refer to this approach as *preference learning*. Many recent works have stud-

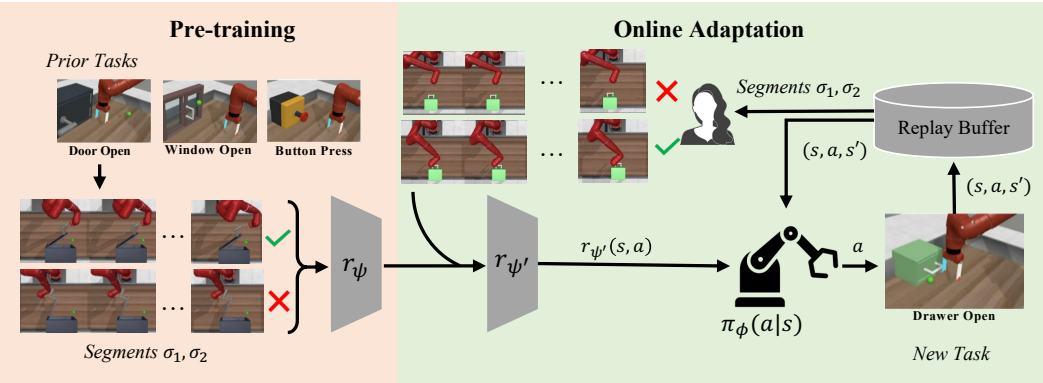

Figure 1: An overview of our method. **Pre-training (left):** In the pre-training phase we generate trajectory segment comparisons using data from a family of previously learned tasks and use it to train a reward model. **Online-Adaptation (Right)**: After pre-training the reward model, we adapt it to new data from human feedback use it to train a policy for a new task in a closed loop manner.

ied active preference-based learning from human feedback, however such approaches often make restrictive assumptions of the reward function, like linearity in predefined features [32, 33, 34, 35, 36]. These assumptions make such methods too inexpressive to scale to modern robot learning with complex objectives. While recent methods combining preference learning with deep RL make no assumptions on the structure of the reward function, they are far too feedback inefficient to be effectively used by humans [11, 9, 37, 38, 39]. Other works that use preferences with deep imitation learning [40] still require demonstrations. Most related to our work, PEBBLE [9] combines the SAC off-policy RL algorithm [41] with an ensemble of learned reward functions for sampling informative comparisons. Unfortunately, PEBBLE still requires an impractical number of queries to learn just a single task (25k for drawer opening). Distinct from prior work, we consider the more realistic multi-task setting that enables us to tap into a large amount of diverse data for for pre-training to increase query-efficiency.

Meta Learning. Meta-learning methods [42, 43] address the few-shot learning problem, where predictions on new tasks are made with a limited amount of data. Inspired by their success in supervised learning problems, we adopt the MAML algorithm [42] for learning new reward functions based on a limited number of human queries. Though supervised meta-learning has been previously used to infer reward classifiers [44] or in learning from heterogeneous demonstrators [45] to our knowledge it has not been applied to reward learning from preferences. Instead of adapting the reward function to new tasks, other related work in meta-RL directly adapts the policy network after a few exploratory episodes [46, 47, 48, 49]. As the RL problem is much more difficult than classification, policy adaptation approaches are likely to be less query efficient.

3 Few-Shot Preference Learning for RL

In this section we formally describe the problem of meta-learning for preference based RL, then detail how our algorithm leverages multi-task pre-training for online few-shot adaptation with preference queries.

Problem Setup. In standard RL, an agent maximizes its cumulative expected reward in a Markov Decision Process (MDP). Unlike standard RL, we assume the reward function $r(s, a)$ to be unknown and instead must be estimated from human feedback. Distinct from prior works in preference based RL, we focus on the multi-task regime and thus additionally assume the existence of a distribution of tasks $p(\mathcal{T})$. Each task τ corresponds to a unique MDP where the state space $\mathcal{S}$, action space $\mathcal{A}$, and discount factor γ are held constant, but the unknown ground-truth reward function $r(s, a)$ varies. Thus, we write that $\tau_i = (\mathcal{P}_i, r_i) \sim p(\mathcal{T})$.

Within this setting, we define the *few-shot preference-based RL* problem. Given access to a dataset of N previous tasks, $\{\tau_i\}_{i=1}^N$, the agents goal is to learn a policy $\pi_{\text{new}}(a|s)$ for a new task $\tau_{\text{new}} \sim p(\mathcal{T})$

from human feedback with as few user queries as possible. We make no explicit assumption on the form of prior data for each of the N prior tasks, only that it contains sufficient information to learn an estimate of the reward r_i. After the pre-training phase, depicted in the left half of Figure 1, we learn policies from online human feedback (right half of Figure 1). This setting is a significant departure from past work in preference-based RL, as we do not assume that tasks are learned in isolation. More realistically, there are multiple tasks we want agents to complete within the same state and action space. Next we explain the major components of our approach.

Preference Learning. In order to learn the policy for a new task $\pi_{\text{new}}(a|s)$ from human preferences, we choose to learn the new tasks' reward function $r_{\text{new}}(s, a)$. While alternative approaches might seek to directly adapt the policy $\pi \rightarrow \pi_{\text{new}}$ using human feedback, such meta-RL style approaches often entail the difficult optimization challenges known to plague policy gradients and dynamic programming [50]. Instead, we directly model reward using supervised learning techniques. We denote $\hat{r}_\psi(s, a)$ to be a learned estimate of an unknown ground-truth reward function $r(s, a)$, parameterized by ψ. As in Wilson et al. [51] we consider preferences over partial trajectory segments $\sigma = (s_t, a_t, s_{t+1}, a_{t+1}, ..., s_{t+k-1}, s_{t+k-1})$ of k states and actions. We then define a preference predictor over segments using the Bradley-Terry model of paired comparisons [52]:

$$P[\sigma_1 \succ \sigma_2] = \frac{\exp \sum_t \hat{r}_\psi(s_t^1, a_t^1)}{\exp \sum_t \hat{r}_\psi(s_t^1, a_t^1) + \exp \sum_t \hat{r}_\psi(s_t^2, a_t^2)}$$

In the above, $\sigma_1 \succ \sigma_2$ indicates the event that segment 1 is preferred to segment 2, as shown in Figure 1. For a given dataset $\mathcal{D}$ comprised of labeled queries (σ_1, σ_2, y) where $y = \{1, 2\}$ corresponds to whether σ_1 or σ_2 is preferred, we optimize the following objective to learn $\hat{r}_\psi$.

$$\mathcal{L}_{\text{pref}}(\psi, \mathcal{D}) = -\,\mathbb{E}_{(\sigma^1,\sigma^2,y)\sim\mathcal{D}} \left[y(1) \log(P[\sigma_1 \succ \sigma_2]) + y(2) \log(1 - P[\sigma_1 \succ \sigma_2]) \right] \tag{1}$$

In practice, this is just the standard binary cross-entropy objective where logits are determined by the sum of the learned reward function σ over k timesteps. Intuitively, this objective seeks to maximize the logits, and consequently predicted reward values, of the preferred segment in comparison to the unpreferred one.

Pre-training for Preference Learning. To estimate the reward function of a new task r_{new} in as few queries as possible, we want to pre-train a reward function $\hat{r}_\psi$ that can quickly adapt to new tasks with only a handful of comparisons (σ_1, σ_2, y). Tapping into offline data can help exploit shared task structure and potential accelerate learning on new tasks. We propose extending the meta-learning framework to preference learning across different tasks. Our approach is agnostic to the choice of meta-learning algorithm, but we choose Model Agnostic Meta-Learning (MAML) [42] for its simplicity. Concretely, MAML searches for parameters ψ that attain high performance on a new task after only a few gradient steps by training on a set of previous tasks. In our setting, data for previous tasks can come from offline datasets, simulated policies, or actual humans. In conjunction with our preference loss from Equation (1), we use the following pre-training update:

Algorithm 1 Few-Shot Preference-based RL

Require: Teacher freq K, Queries per session M
$\psi \leftarrow \arg\min_\psi \sum_i \mathcal{L}\left(\psi - \alpha \nabla_\psi \mathcal{L}(\psi, \mathcal{D}_i), \mathcal{D}_i\right)$
for $t = 1, 2, 3, ...$ **do**
  if $t\%K == 0$ **then**
    for $m = 1, 2, ...M$ **do**
      $(\sigma_1, \sigma_2) \sim$ Disagreement
      $y \leftarrow$ user preference
      $\mathcal{D}_{\text{new}} \leftarrow \mathcal{D}_{\text{new}} \cup (\sigma_1, \sigma_2, y)$
    end for
    $\psi' \leftarrow \psi$ Re-initialize reward model
    for each gradient step **do**
      $\psi' \leftarrow \psi' - \alpha \nabla_{\psi'} \mathcal{L}_{\text{pref}}(\psi', \mathcal{D}_{\text{new}})$
    end for
  end if
  Collect s_{t+1} by taking $a_t \sim \pi(a_t|s_t)$
  Store transition $\mathcal{B} \leftarrow \mathcal{B} \cup (s_t, a_t, s_{t+1})$
  Sample batch $\{(s_t, a_t, s_{t+1})\}_{j=1}^{B} \sim \mathcal{B}$
  Assign rewards $r_t \leftarrow r_{\psi'}(s_t, a_t))$
  Optimize π via SAC with $\{(s_t, a_t, s_{t+1}, r_{\psi'}(s_t, a_t))\}_{j=1}^{B}$
end for

$$\psi \leftarrow \psi - \beta \nabla_\psi \sum_{i=1}^{N} \mathcal{L}_{\text{pref}}(\psi - \alpha \nabla_\psi \mathcal{L}_{\text{pref}}(\psi, \mathcal{D}_i), \mathcal{D}_i). \tag{2}$$

Here α and β are the inner and outer learning rates respectively. Each dataset $\mathcal{D}_i$ is comprised of known queries for each of the N tasks $\tau_i \sim p(\mathcal{T})$. When we start training for a new task, we can quickly adapt the reward function using the new queries as $\psi' \leftarrow \psi - \alpha \nabla_\psi \mathcal{L}_{\text{pref}}(\psi, \mathcal{D}_{\text{new}})$. As ψ is explicitly optimized for performance on $\mathcal{L}_{\text{pref}}$ after only a handful of updates, we significantly reduce query complexity.

Training $\hat{r}_\psi$ using Equation (2) however, requires access to query datasets $\mathcal{D}_i$ for each task. While pre-training can be accomplished through several objectives, like reward regression, we use preference-based pre-training for consistency and its generality. Pairwise comparison data can be extracted from a wide variety of sources. If reward values are present in offline data, artificial labels y for trajectory segments σ_1, σ_2 can easily be generated via the comparison $\sum_t r(s_t^1, a_t^1) > \sum_t r(s_t^2, a_t^2)$ as is common practice in prior works [11]. If reward values for previous tasks are unknown but policies are, reward values can be recovered via inverse-RL, or comparisons can be derived from direct behavior comparison. For example, when generating queries for task i, behaviors from $\pi_i(a|s)$ would be preferred to behaviors generated from $\pi_{\neq i}(a|s)$. The left half of Figure 1 shows the process of extracting query data from offline data for pre-training, which corresponds to line 1 in Algorithm 1. In our experiments, we use the artificial reward labeling scheme described first for consistency with prior work [11].

Few-Shot Preference-based RL. Our pre-trained preference function can then be used for few-shot preference based RL during an online adaptation phase, depicted in the right half of Figure 1. We modify the standard Soft-Actor Critic RL algorithm [41] to relabel transitions using our learned reward function before performing a standard actor-critic update (Algorithm 1 lines 17-18). Every K steps, we ask a user to answer queries and provide feedback labels y as shown in lines 5-7 of Algorithm 1. Informative queries are selected using the disagreement of an ensemble of reward functions over the preference predictors. Specifically, comparisons that maximize $\text{std}(P[\sigma_1 \succ \sigma_2])$ are selected each time feedback is collected. After new feedback is collected, we re-initialize the reward model $\hat{r}_\psi$ to its pre-trained weights. Subsequently, we re-adapt it using the updated dataset $\mathcal{D}_{\text{new}}$ for the new task as shown in Algorithm lines 9-11.

To our knowledge, we are the first to leverage multi-task data for preference-based RL. This paradigm shift necessitates critical algorithmic changes in comparison with prior work. First, we pre-train the reward function from prior data instead of using other warm-start methods like unsupervised exploration used in PEBBLE. Second, we crucially reset the reward model for adaptation. Our setting provides a novel framework that leverages pre-training on a range of tasks for data-efficient adaptation on new tasks enabling human users to provide this data without making any structural assumptions on the reward function.

4 Experiments

In this section we seek to answer the following questions: First, does few-shot preference learning improve the query efficiency of preference-based RL? Second, is our method efficient enough to learn robot policies from real human feedback? Finally, can few-shot preference learning be used in the real world? Full experiment details are available in the Appendix.

4.1 How query-efficient is few-shot preference-based RL?

To test the query-efficiency of few-shot preference based RL for realistic robotic tasks, we adopt the Meta-World benchmark from Yu et al. [12]. Agent tasks include household activities like opening doors or closing windows, and standard manipulation problems like block pushing. Meta-World tasks are particularly difficult for human-in-the-loop learning as they are sequential: feedback on the second part of the task, like where an agent should move a block, can only be provided once the agent learns the first part of the task, like how to grasp a block. Additionally, different objects introducing different manipulation dynamics across tasks. To evaluate the raw-performance of our approach, we use the artificial queries induced by the task ground truth reward function. Previous works in preference based RL have required up to fifty-thousand of these artificial queries in order

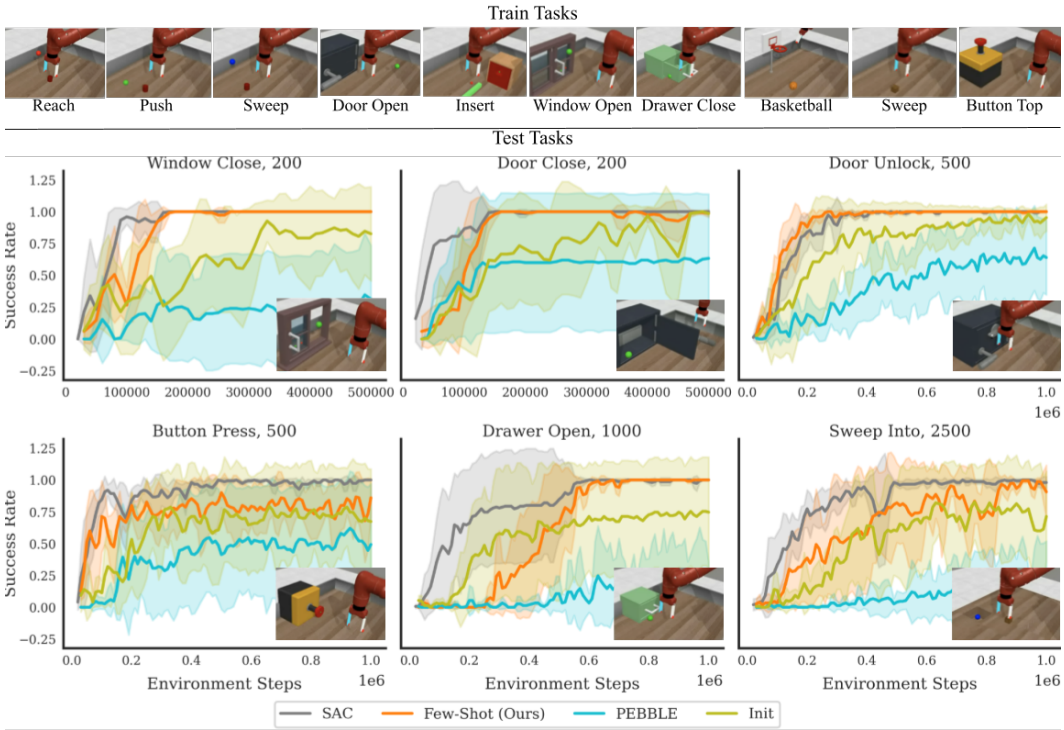

Figure 2: Results on MetaWorld tasks. The title of each subplot indicates the task and number of artificial feedback queries used in training. Results for each method are shown across five seeds.

to solve some of the Meta-world tasks [9]. Our approach generally achieves the same performance using 20X fewer queries. Our reward models are pre-trained using *only 10 prior tasks* and evaluate query-efficiency on six previously unseen tasks. We compare our method, which we refer to as *Few-Shot*, to three baselines:

1. **SAC**: The Soft-Actor Critic RL algorithm trained from ground truth rewards. This represents "oracle" performance.
2. **PEBBLE**: The PEBBLE algorithm from Lee et al. [9], which does not use any prior data.
3. **Init**: This baseline demonstrates the importance of our adaptation procedure during training. Instead of re-adapting the reward model each time new feedback is collected, we initialize the reward model with the pretrained weights, and then perform standard updates with the Adam optimizer [53] as in PEBBLE.

For each environment, we reduce the total feedback budget by a factor of 20 in comparison to the maximum value used in PEBBLE, but keep all other hyper-parameters the same. Full results are shown in Figure 2. Overall, we find that despite the 20X reduction in feedback our method is able to solve almost all of the tasks with a near 100% success rate. In particular Lee et al. [9] use 10k, 50k, and 50k feedback for Button Press, Drawer Open, and Sweep Into respectively. In all tasks, except Button Press, we achieve the same asymptotic performance as SAC. On most tasks, PEBBLE alone is unable to learn a meaningful policy, consistently attaining a sub 50% success rate. While the *Init* baseline generally performs better than PEBBLE, it still falls short of our method, indicating that re-adaptation is important for attaining maximal performance.

4.2 Can few-shot preference learning be used with humans?

While no human could be sensibly be expected to provide thousands of pieces of feedback, around a hundred or less not too daunting a task. Given the lower query-complexity of few-shot preference-based RL, we use it to learn complex robot manipulation policies from real-human feedback for the first time. In the process of doing so, we encountered a few challenges. First, humans often have a

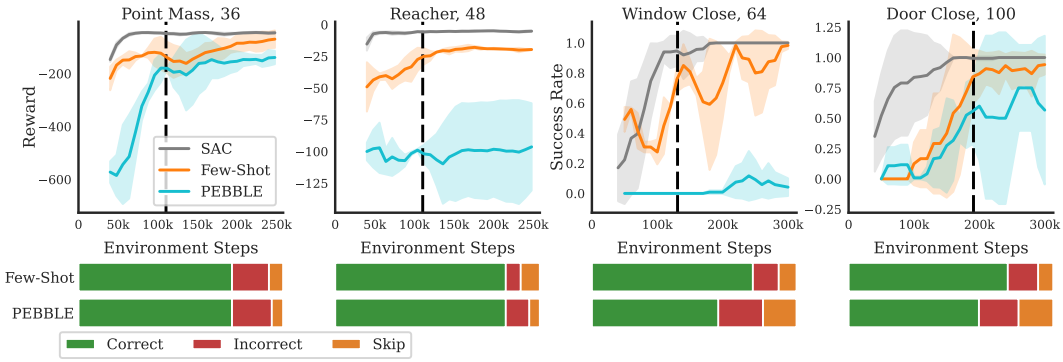

Figure 3: Results on training DM Control and Meta-World tasks from real human feedback. The vertical dashed black line indicates the point at which feedback was stopped. The horizontal bars at the bottom show the proportion of times users provided feedback that "correctly" agreed with the tasks ground-truth reward function, "incorrectly" disagree with the ground-truth reward function, or skipped the comparison.

difficult time answering queries asked by preference based RL algorithms. Queries sampled by maximizing disagreement across an ensemble of reward functions often look identical to humans. Such queries at the margin may be maximally informative, but are more difficult to answer (See Figure 6). For example, it is unlikely that humans can accurately compare two behavior segments that only have slight variations in the robot joint positions. To remedy this, we add the ability for human users to "skip" difficult queries and increased the number of uniform queries before starting to sample with disagreement. Second, humans still make mistakes in labeling, allowing query labels to be inconsistent. We thus allowed policies to train for longer periods of time in between feedback sessions.

After making the aforementioned changes, we examined the performance of few-shot preference-based RL on two of the MetaWorld environments and two additional environments based on the DM Control benchmark [54]. Specifically, we take the point mass and reacher environments from DM Control, and change the reward function to be the negative L2 distance to an unknown goal. We pre-train reward models on random data and evaluate on unknown goal positions. Our full results are shown in Figure 3. As the ground-truth reward value for DM control correspond to the cumulative distance to the goal, the higher reward values of our method indicate that it can better communicate the human's objective with fewer queries. While PEBBLE was completely unable to solve Window-Close from human feedback, it made non-trivial progress on Door Close. This is likely because Door Close can be trivially solved by slamming the robot arm into the door instead of first grasping the handle and then closing the door as is encourage by our reward-function prior. Moreover, we find that in the Meta-World environments, users have an easier time answering queries from our method in comparison to PEBBLE. This is likely because the reward function prior guides agents towards interacting with objects, leading to more distinguishable behaviors.

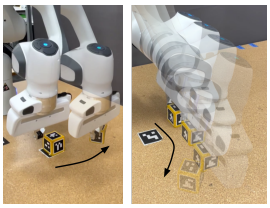

Figure 4: Push rollouts

	Reach		Block Push	
Goal Position	$(.55, .35)$	$(.45, -.3)$	$(.35, .3)$	$(0.35, -0.3)$
Few-Shot	$\mathbf{.061 \pm .041}$	$\mathbf{.056 \pm .009}$	$\mathbf{.188 \pm .175}$	$\mathbf{.056 \pm .035}$
PEBBLE	$.105 \pm .056$	$.129 \pm .067$	$.280 \pm .065$	$.173 \pm .097$

Figure 5: Results for the real-robot tasks. Performance is measured in meters to the desired goal position, lower is better. The z targets for reach were 0.125 and 0.25, respectively. Results are averaged across multiple initial environment configurations. Best method is bolded.

4.3 Can Few-Shot preference-based RL be used in the real world?

Finally, we investigate the use of few-shot preference-based RL in real world settings using a Franka Panda Robot. We design two basic tasks: reaching and block pushing. The robots objective is to move either its end-effector or a block to an unknown goal location communicated only via the learned reward function. We pre-train the reward function using artificially generated queries and transfer policies trained in simulation on unseen goal locations to the real world. More experiment

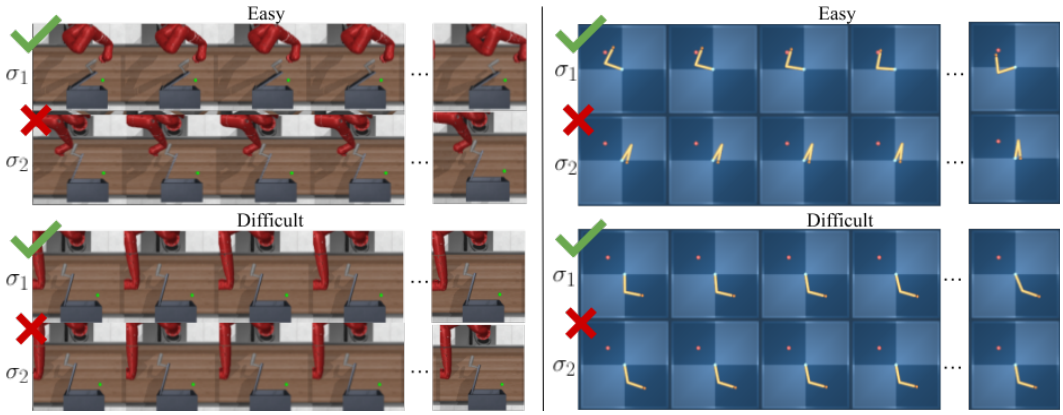

Figure 6: Examples of queries asked on the MetaWorld Door-Close and DM Control Reacher environments. In each figure the top segment, σ_1 had higher cumulative reward.

details and learning curves are included in the Appendix. Table 5 contains our results. Performance is measured in meters to the true goal after transferring to the real world. Again, few-shot preference learning consistently outperforms PEBBLE despite the large sim-to-real gap. One additional benefit of our approach in real-world settings is that it potentially requires less instrumentation, as measurements previously needed functionally compute reward are not required when using human feedback.

5 Discussion

Limitations and Future Work. While few-shot preference learning has several benefits, it has its limitations. Here we list the most salient ones, and possible means of overcoming them:

Query Complexity. Despite our gains in query-efficiency, the most complex tasks still require more feedback than we would like. Future work could examine how to expand set of pre-training data.

Pre-training Methods . We investigate pre-training with artificial query data due to its generality, though our method could be used in combination with other pre-training objectives, like direct reward regression, to further boost performance.

Pre-training Data. While meta-learning methods have proven to be somewhat robust to changing dynamics in the real world [55], the efficacy of reward adaptation under larger distribution shifts induced by sub-optimal users, new tasks, or sim-to-real transfer remains in question. For example, if a new task is significantly out of distribution, we would expect training a reward function from scratch to perform better than adapting. Furthermore, the pre-training can occasionally over-regularize the learned reward model, as exhibited in the Door Close experiment in Section 4.2.

Query Difficulty. Many queries asked by preference learning algorithms are too difficult for humans to answer, as shown in Figure 6. In fact, we find in Section 4.2 that active query schemes often result in queries that are too difficult for users to answer. Future work should explicitly consider how easy it is for a human to answer a query and not just its theoretical information content.

User Inconsistency. Unlike reward oracles, humans will inconsistently label queries. This challenge is only exacerbated when attempting to crowd source data from many users with differing styles. Future work can investigate additionally modeling human users.

Conclusion. We shift the paradigm of human-in-the-loop RL from the single-task to the multi-task setting, unlocking additional data sources that can be used to boost the query-efficiency of preference-based RL Algorithms. We believe our work's change in perspective to be a crucial stepping stone towards training robots with human feedback. Our novel few-shot preference-based RL method is able to effectively minimize the number of human queries required to train complex manipulation policies as demonstrated by our 20X improvement on standard benchmarks and effectiveness at real-human training.

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[2] H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine. The ingredients of real world robotic reinforcement learning. In *International Conference on Learning Representations*, 2020.

[3] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan. Inverse reward design. *Advances in neural information processing systems*, 30, 2017.

[4] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[5] D. P. Losey, K. Srinivasan, A. Mandlekar, A. Garg, and D. Sadigh. Controlling assistive robots with learned latent actions. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 378–384. IEEE, 2020.

[6] L. Smith, N. Dhawan, M. Zhang, P. Abbeel, and S. Levine. Avid: Learning multi-stage tasks via pixel-level translation of human videos. *RSS*, 2020.

[7] S. Karamcheti, R. Krishna, L. Fei-Fei, and C. D. Manning. Mind your outliers! investigating the negative impact of outliers on active learning for visual question answering. *CoRR*, abs/2107.02331, 2021. URL https://arxiv.org/abs/2107.02331.

[8] D. Lowell, Z. C. Lipton, and B. C. Wallace. Practical obstacles to deploying active learning. *arXiv preprint arXiv:1807.04801*, 2018.

[9] K. Lee, L. M. Smith, and P. Abbeel. Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In *International Conference on Machine Learning*, pages 6152–6163. PMLR, 2021.

[10] K. Lee, L. Smith, A. Dragan, and P. Abbeel. B-pref: Benchmarking preference-based reinforcement learning. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=ps95-mkHF_.

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

[12] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/abs/1910.10897.

[13] A. Turner, N. Ratzlaff, and P. Tadepalli. Avoiding side effects in complex environments. *Advances in Neural Information Processing Systems*, 33:21406–21415, 2020.

[14] P. Kormushev, S. Calinon, and D. G. Caldwell. Robot motor skill coordination with em-based reinforcement learning. In *2010 IEEE/RSJ international conference on intelligent robots and systems*, pages 3232–3237. IEEE, 2010.

[15] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1, 2004.

[16] D. Ramachandran and E. Amir. Bayesian inverse reinforcement learning. In *IJCAI*, volume 7, pages 2586–2591, 2007.

[17] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[18] D. S. Brown, W. Goo, and S. Niekum. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pages 330–359. PMLR, 2020.

[19] R. P. Khurshid and K. J. Kuchenbecker. Data-driven motion mappings improve transparency in teleoperation. *Presence*, 24(2):132–154, 2015.

[20] A. D. Dragan and S. S. Srinivasa. *Formalizing assistive teleoperation*. MIT Press, July, 2012.

[21] D. P. Losey, H. J. Jeon, M. Li, K. Srinivasan, A. Mandlekar, A. Garg, J. Bohg, and D. Sadigh. Learning latent actions to control assistive robots. *Autonomous robots*, 46(1):115–147, 2022.

[22] C. Basu, Q. Yang, D. Hungerman, M. Sinahal, and A. D. Draqan. Do you want your autonomous car to drive like you? In *2017 12th ACM/IEEE International Conference on Human-Robot Interaction (HRI*, pages 417–425. IEEE, 2017.

[23] M. Kwon, E. Biyik, A. Talati, K. Bhasin, D. P. Losey, and D. Sadigh. When humans aren't optimal: Robots that collaborate with risk-aware humans. In *2020 15th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 43–52. IEEE, 2020.

[24] L. Chen, R. Paleja, and M. Gombolay. Learning from suboptimal demonstration via self-supervised reward regression. *arXiv preprint arXiv:2010.11723*, 2020.

[25] M. Li, A. Canberk, D. P. Losey, and D. Sadigh. Learning human objectives from sequences of physical corrections. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2877–2883. IEEE, 2021.

[26] J. D. Co-Reyes, A. Gupta, S. Sanjeev, N. Altieri, J. Andreas, J. DeNero, P. Abbeel, and S. Levine. Guiding policies with language via meta-learning. In *International Conference on Learning Representations*, 2019.

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

[28] W. B. Knox and P. Stone. Tamer: Training an agent manually via evaluative reinforcement. In *2008 7th IEEE international conference on development and learning*, pages 292–297. IEEE, 2008.

[29] V. Myers, E. Biyik, N. Anari, and D. Sadigh. Learning multimodal rewards from rankings. In *Conference on Robot Learning*, pages 342–352. PMLR, 2022.

[30] D. Brown, W. Goo, P. Nagarajan, and S. Niekum. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pages 783–792. PMLR, 2019.

[31] E. Bıyık, D. A. Lazar, D. Sadigh, and R. Pedarsani. The green choice: Learning and influencing human decisions on shared roads. In *2019 IEEE 58th conference on decision and control (CDC)*, pages 347–354. IEEE, 2019.

[32] D. Sadigh, A. D. Dragan, S. S. Sastry, and S. A. Seshia. Active preference-based learning of reward functions. In *Proceedings of Robotics: Science and Systems (RSS)*, July 2017. doi: 10.15607/RSS.2017.XIII.053.

[33] X. Wang, K. Lee, K. Hakhamaneshi, P. Abbeel, and M. Laskin. Skill preferences: Learning to extract and execute robotic skills from human feedback. In A. Faust, D. Hsu, and G. Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1259–1268. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/wang22g.html.

[34] M. Cakmak, S. S. Srinivasa, M. K. Lee, J. Forlizzi, and S. Kiesler. Human preferences for robot-human hand-over configurations. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1986–1993, 2011. doi:10.1109/IROS.2011.6094735.

[35] E. Biyik, N. Huynh, M. J. Kochenderfer, and D. Sadigh. Active preference-based gaussian process regression for reward learning. In *Proceedings of Robotics: Science and Systems (RSS)*, July 2020.

[36] J. R. Lepird, M. P. Owen, and M. J. Kochenderfer. Bayesian preference elicitation for multi-objective engineering design optimization. *Journal of Aerospace Information Systems*, 12(10): 634–645, 2015.

[37] J. Park, Y. Seo, J. Shin, H. Lee, P. Abbeel, and K. Lee. SURF: Semi-supervised reward learning with data augmentation for feedback-efficient preference-based reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TfhfZLQ2EJO.

[38] X. Liang, K. Shu, K. Lee, and P. Abbeel. Reward uncertainty for exploration in preference-based reinforcement learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=OWZVD-l-ZrC.

[39] G. Warnell, N. Waytowich, V. Lawhern, and P. Stone. Deep tamer: Interactive agent shaping in high-dimensional state spaces. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[40] D. S. Brown and S. Niekum. Deep bayesian reward learning from preferences. *arXiv preprint arXiv:1912.04472*, 2019.

[41] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

[42] C. Finn, P. Abbeel, and S. Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[43] A. Nichol and J. Schulman. Reptile: a scalable metalearning algorithm. *arXiv preprint arXiv:1803.02999*, 2(3):4, 2018.

[44] A. Xie, A. Singh, S. Levine, and C. Finn. Few-shot goal inference for visuomotor learning and planning. In *Conference on Robot Learning*, pages 40–52. PMLR, 2018.

[45] M. L. Schrum, E. Hedlund-Botti, and M. C. Gombolay. Personalized meta-learning for domain agnostic learning from demonstration. In *Proceedings of the 2022 ACM/IEEE International Conference on Human-Robot Interaction*, pages 1179–1181, 2022.

[46] J. X. Wang, Z. Kurth-Nelson, H. Soyer, J. Z. Leibo, D. Tirumala, R. Munos, C. Blundell, D. Kumaran, and M. M. Botvinick. Learning to reinforcement learn. In *CogSci*, 2017. URL https://mindmodeling.org/cogsci2017/papers/0252/index.html.

[47] R. Agarwal, C. Liang, D. Schuurmans, and M. Norouzi. Learning to generalize from sparse and underspecified rewards. In *International Conference on Machine Learning*, pages 130–140. PMLR, 2019.

[48] Y. Duan, J. Schulman, X. Chen, P. L. Bartlett, I. Sutskever, and P. Abbeel. Rl˜2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[49] K. Rakelly, A. Zhou, C. Finn, S. Levine, and D. Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

[50] W. B. Powell. Perspectives of approximate dynamic programming. *Annals of Operations Research*, 241(1):319–356, 2016.

[51] A. Wilson, A. Fern, and P. Tadepalli. A bayesian approach for policy learning from trajectory preference queries. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/16c222aa19898e5058938167c8ab6c57-Paper.pdf.

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

[53] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015. URL http://arxiv.org/abs/1412.6980.

[54] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa. $\mathrm{dm}_{c}ontrol : Softwareandtasksforcontinuouscontrol. \textit{Software Impacts}, 6 : 100022, 2020. ISSN2665 - 9638. doi : .$ URL https://www.sciencedirect.com/science/article/pii/S2665963820300099.

I. Clavera, A. Nagabandi, S. Liu, R. S. Fearing, P. Abbeel, S. Levine, and C. Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HyztsoC5Y7.