Bayesian Hypothesis Testing Policy Regularization

Sarah Rathnam[1], **Finale Doshi-Velez**[2], **Susan A. Murphy**[2]
sarah_rathnam@g.harvard.edu

[1]**Department of Applied Mathematics, Harvard University, Boston, MA**
[2]**Department of Computer Science, Harvard University, Boston, MA**

Abstract

In reinforcement learning (RL), sparse feedback makes it difficult to target long-term outcomes, often resulting in high-variance policies. Real-world interventions instead rely on prior study data, expert input, or short-term proxies to guide exploration. In this work, we propose *Bayesian Hypothesis Testing Policy Regularization* (BHTPR), a method that integrates a previously-learned policy with a policy learned online to speed up learning in such settings. BHTPR applies the inductive bias that the prior study data matches the current study environment in some states but is incorrect in others. We use Bayesian hypothesis testing to determine, state by state, when to transfer the prior policy and when to rely on online learning.

1 Introduction

In RL settings with sparse or delayed rewards, exploration can be costly or risky—particularly. Mobile health studies are one key example. Outcomes such as a physical fitness assessment, lower blood pressure, or smoking cessation, may only be observed infrequently or at the study's end. Furthermore, excessive exploratory actions (such as sending motivational prompts) can overburden participants, causing disengagement. To reduce variance and speed up learning, clinicians often introduce proximal outcomes or other imperfect mediators that correlate with the distal health goal. For example, in the mobile health algorithm "HeartSteps", the RL agent targeted a "short-term, measurable behavioral or psychosocial effect through which that component is hypothesized to mediate desired distal health outcomes" (Klasnja et al., 2019). Those proximal outcomes induce a policy that can steer the agent toward distal success while limiting patient burden. Throughout this paper we use mobile health as the running application scenario motivating our method and experiments.

One RL mechanism for injecting such prior behavioral knowledge is policy regularization. Policy regularization is commonly used in offline RL to reduce the learned policy's deviation from the behavior policy (Wu et al., 2019). In this setting, policy regularization reduces the bias caused by optimism when extrapolating beyond the observed data (Kumar et al., 2020). In this paper, we instead apply policy regularization to online learning with sparse, distal rewards. In this setting, rather than prevent optimism, policy regularization guides learning using a policy calculated from previous study data.

In this paper, we introduce Bayesian Hypothesis Testing Policy Regularization (BHTPR), an online method that uses Bayesian hypothesis testing to selectively leverage data from a previous study. We assume that the transition dynamics—and hence the optimal policy—induced by this previous data are accurate only in a subset of states. In other states, the dynamics may differ, rendering the previous policy suboptimal. BHTPR addresses this mismatch by learning, via Bayesian hypothesis testing, the states in which the previous policy is likely to be optimal. At each timestep, the agent chooses an action from either the policy calculated from the previous data (using any standard method such as policy iteration) or from the policy learned online.The probability of acting according to each policy is determined by a Bayesian hypothesis test. Because BHTPR chooses between policies rather than combining Q-values, it remains robust even when outcome scales differ across environments. Furthermore, by assigning higher probability to the previous policy in states where the hypothesis test indicates higher similarity, the agent reduces unnecessary exploration.

We demonstrate the performance of BHTPR across a range of experiments varying episode length, quality of prior data, environment stochasticity, and other factors. BHTPR demonstrates good performance both on a simple tabular example and on a more realistic mobile health setting. We compare BHTPR to (1) standard epsilon-greedy exploration, and (2) using the previous study data as a prior on the transition function. We find that our method performs similarly to or better than baselines in variations in the setting and environment and is robust to parameter choices.

2 Related Works

Approaches for Sparse, Distal Rewards The problem of pursuing a distal outcome in the face of sparse feedback is a core problem of RL that has been addressed by many areas of research. Reward design focuses on providing structured signals that guide behavior and accelerate learning (Singh et al., 2009; Sowerby et al., 2022; Hadfield-Menell et al., 2017). Reward design is a key challenge for mobile health studies Trella et al. (2023). Reward shaping augments the environment's reward function with additional feedback to guide the agent to learn more quickly without altering the optimal policy (Ng et al., 1999). Safe RL focuses on problems "in which it is important to ensure reasonable system performance and/or respect safety constraints during the learning and/or deployment process" (Garcıa & Fernández, 2015). Finally, transfer learning leverages experience gained from similar tasks to improve the learning of a novel task (Taylor & Stone, 2009; Lazaric, 2012). In this work, we modify exploration using external knowledge. We build on prior works by using Bayesian hypothesis testing to reflect the inductive bias that the external knowledge is correct at some states and incorrect at others.

Policy Regularization Policy regularization is commonly used to avoid a policy deviating too far from its prior version or from a reference policy, improving stability, sample efficiency, and generalization. In offline settings, policy regularization ensures the learned policy does not deviate far from the behavior policy, reducing extrapolation error (Wu et al., 2019). In online settings, methods such as Trust Region Policy Optimization (TRPO) constrain successive policy updates using KL divergence penalties (Schulman et al., 2015). In a complementary line of work, Lu et al. (2023), combines imitation learning and reinforcement learning. They use an imitation learning objective and modify this with an RL objective when the data are out-of-distribution.

Our method can be viewed as policy regularization, as it constrains the learned policy to use a policy derived from prior study data in certain states. Furthermore, it can be framed within BRAC, as discussed in Sec. 6.

Bayesian Hypothesis Testing Bayesian hypothesis testing (BHT) is a method of comparing two competing hypotheses or models by computing how likely the observed data are under each hypothesis (Fay & Brittain, 2022). We are aware of one other work that explicitly incorporates BHT into an RL algorithm. Zhang et al. (2022) uses BHT to interpolate between MDP and bandit algorithms when the true nature of the environment is not known. In contrast, our method assumes that the true environment is an MDP and the role of BHT is to learn the states in which the given policy is optimal.

3 Methods

3.1 Setting and Notation

We consider an episodic Markov decision process (MDP) setting with, with states $s \in \mathcal{S}$, actions $a \in \mathcal{A}$, reward function $R(s, a)$ and transition function $T(s, a, \cdot)$. In the experiments below, we assume $R(s, a)$ is known. We define $\mathcal{D}$ to be the transition data collected online. We use this data and the reward function to learn a policy π and an MLE transition matrix $\hat{T}$. We define $\mathcal{D}_{prior}$ as the transition data collected from some prior study. We compute the optimal policy based on this transition data and the know reward function $R(s, a)$, and call this policy π_{prior}. We compute $\hat{T}_{prior}$ as the maximum likelihood estimate (MLE) transition matrix using $\mathcal{D}_{prior}$. The transition function generated from $\mathcal{D}_{prior}$ may match that of the new environment or it may differ in certain states. Consequently, we aim to rapidly differentiate between states where π_{prior} can be utilized and states where a new policy must be learned online.

3.2 Algorithm Definition

BHTPR learns weights $w(s)$ that reflects the agent's belief about whether the transition dynamics in the current environment match those from the prior study in each state s. Then $w(s)$ is used within the algorithm to decide whether to select an action in state s according to π (the policy being learned online) or π_{prior} (the policy computed from the prior study data).

3.2.1 Update rule for $w(s)$ using Bayesian hypothesis testing

BHTPR assumes access to data $\mathcal{D}_{prior}$ that comes from an environment with the same state space, but potentially different transition dynamics in some states. In order to differentiate between the states in which the transition dynamics are the same and those that differ, BHTPR learns a state-specific weight $w(s) \in [0, 1]$ that represents the probability that the transition dynamics in the current environment match those implied by the prior data. This is formalized via Bayesian hypothesis testing, which updates $w(s)$ as more transition data are collected.

In each state, we test the null hypothesis that transitions match the prior environment against the alternative hypothesis that they do not. Using observed transition data, we update $w(s)$ to reflect the posterior probability that the prior model is correct in that state. We use Bayes' rule to compute this posterior probability, based on the likelihood of transition data under the prior and learned models. The update equation and further details are provided in Appx. A.

3.2.2 BHTPR Algorithm: Using $w(s)$ to choose between policies

At every step of episode e, the online algorithm chooses the policy from which to select its action based on a draw from a Bernoulli trial, where the probability of acting according to $\pi_{prior}(s)$ is $w_e(s)$. If the resulting value $b \sim Bernoulli(w_e(s))$ equals 1, the agent selects its action based on $\pi_{prior}(s)$, otherwise it selects an action from the current learned policy $\pi(s)$. Hence the agent takes actions based on the previous study data in proportion to the probability that the observed data was generated from the previous study transition function. This procedure is detailed in Algorithm 1.

4 Experiments

We demonstrate the ability of BHTPR to leverage previous study data to speed up learning in settings with sparse, distal rewards. We show how the algorithm can outperform a standard epsilon-greedy approach as well as an epsilon-greedy approach using the previous study data directly as a Dirichlet prior on the transition function. Additional experimental details are in Appx. C.

4.1 Environments

We illustrate the performance of BHTPR on experiments using two environments. The first is a grid in which an agent must navigate from one corner to the opposite corner. This simple example allows us to manipulate different aspects of the environment and observe the effect on the performance of BHTPR. The second environment is the "Chainworld" from Nofshin et al. (2024). This reflects a realistic depiction of the challenges faced in a mobile health setting.

Algorithm 1 Bayesian hypothesis testing policy regularization

```
1: Input: Previous study data $\mathcal{D}_{prior}$
2: Calculate $\hat{T}_{prior}, \pi_{prior}$ using $\mathcal{D}_{prior}$; Initialize $\hat{T}(s, a, \cdot)$, weight $w_0(s)$, policy $\pi$
3: for each episode $e$ do:
4:     Initialize episode data $\mathcal{D}_e(s) = \{\} \forall s$
5:     Initialize state $s$ from starting state distribution
6:     while $s$ not terminal do
7:         Draw $b \sim Bernoulli(w_e(s))$
8:         if $b = 1$ then
9:             Choose action $a \sim \pi_{prior}(s)$
10:        else if $b = 0$ then
11:            Choose action $a$ epsilon-greedily from $\pi(s)$
12:        end if
13:        Take action $a$. Observe next state $s'$. Update data $\mathcal{D}_e(s) = \mathcal{D}_e(s) \cup (s, a, s')$.
14:        Update $\hat{T}, \pi$ (e.g. by Q-learning)
15:        $s \leftarrow s'$
16:    end while
17:    Update $w$ using Bayesian hypothesis testing (Eq. 1)
18: end for
```

Gridworld The state space of this environment is a 7x7 grid with four actions (up/down/left/right). The agent starts at the lower-left of the grid and the episode concludes when either the agent reaches the upper-right corner or after a fixed number of steps. The agent receives a reward of -1 per step until reaching the goal. When the agent reaches the goal, it receives a reward of 100 and the episode terminates. This is depicted in Fig. 4a in Appx. B.

Chainworld A more realistic mobile health example, the "Chainworld" from Nofshin et al. (2024), applies behavioral science literature to model human-AI interaction when an AI assists a human in a frictionful task such as adhering to a physical therapy program. Both the human and the AI are represented by the MDPs, where the AI MDP's actions affect the human's MDP. The human MDP represents progress towards the goal and the AI's MDP uses the human's MDP state and action as its state space. Further details are in Appx. B.

5 Results

BHTPR has good performance across variations in the gridworld and chainworld environments. In the chainworld, the results are greatly shaped by the presence of the absorbing state.

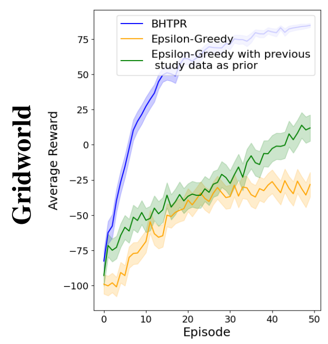
5.1 Impact of an Absorbing State on Exploration

The main element distinguishing the chainworld environment is the consequence of exploration. In contrast to the gridworld, where unnecessary exploration increases the number of steps to reach the goal, exploration in the chainworld can result in disengagement after which the agent can never reach the goal state. Hence exploration is incredibly costly and the best algorithm in this environment is one that minimizes unnecessary exploration. BHTPR can reduce exploration because, once the agent has learned that π_{prior} is optimal, it can stop exploring in that state. In terms of Algorithm 1, this means acting epsilon-greedily only if $b = 0$ (acting according to the policy π learned online, line 11) and not exploring when acting according to π_{prior} (line 9).

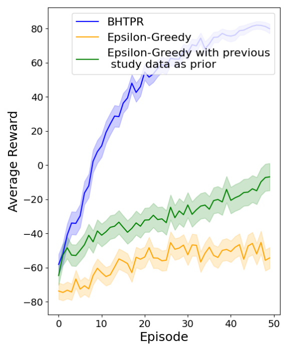
Fig. 5 in Appx. D highlights the impact of exploration on the performance of BHTPR and the baselines in the chainworld. In the right column, equal exploration is enforced for all methods. Here BHTPR performs similarly to the prior on T. This holds in both the case when there is a large negative reward for disengagement (bottom row) and when there is not (top row). Epsilon-greedy is slower to learn without the large disengagement reward but performs similarly to the other two in the presence of a large disengagement reward. In contrast, when BHTPR is not forced to explore equally with the baselines, it outperforms the baselines, as seen in the left column of Fig. 5.

For the sake of comparison between methods, in the gridworld example, we fix the amount of exploration to be equal across BHTPR and the baselines. In the chainworld examples, however, since performance is driven by exploration, BHTPR is implemented in the results that follow with *unequal exploration* in the chainworld environment (i.e. No exploration when $b = 1$.).

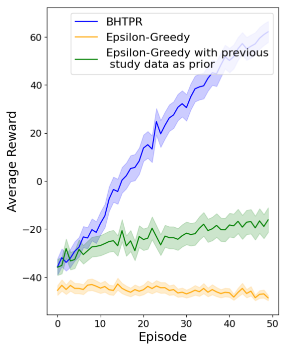
5.2 BHTPR can learn over short episodes.



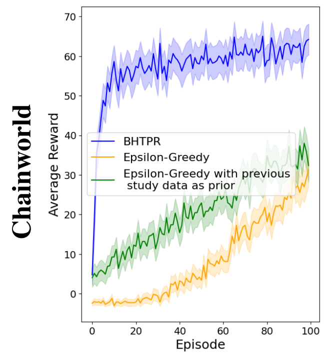
(a) 150-step episodes



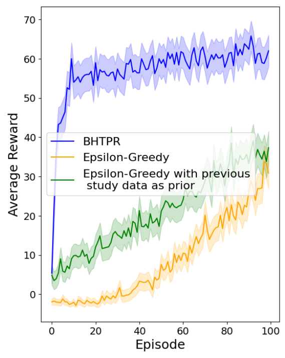
(b) 100-step episodes



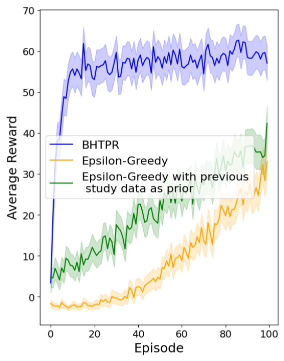
(c) 50-step episodes



(d) 50-step episodes



(e) 30-step episodes



(f) 20-step episodes

Figure 1: **Episode length.** BHTPR learns in gridworld even with short episodes, unlike the baselines. In chainworld, episode length has limited effect due to disengagement.

Gridworld By incorporating correct actions learned from the previous study data, BHTPR is able to learn over short episodes. This is the case even when episodes are too short for the baselines to learn. Fig. 1 illustrates the performance of our method compared to the baselines in environments of different episode lengths. In the gridworld, all three methods are able to learn when the episodes are sufficiently long (Fig. 1a), however, when the episodes are short (Fig. 1c), the epsilon-greedy agent cannot learn at all and epsilon-greedy agent with the prior study data as a prior on T learns very slowly. BHTPR is able to learn with a greatly reduced number of steps per episode compared to standard epsilon-greedy learning because it quickly learns the states in which π_{prior} is the optimal policy and does not have to spend as many steps in the episode taking suboptimal actions.[1] To confirm this intuition, see Fig. 6 in Appx. D. Here, we mix actions from the true optimal policy with the learned policy for a range of fixed percentages (25, 50, and 75 percent). The figure shows

[1]Exploration is equalized across methods in the Gridworld example, however fewer wrong actions are still taken by BHTPR because in the states π_{prior} is optimal, the agent rapidly learns to use this. The baselines take more suboptimal actions while learning the policy online.

that as the proportion of optimal actions increases, the performance approaches that of BHTPR, demonstrating that the reason our method can learn with fewer episode steps is due to incorporating optimal actions.

Chainworld In contrast to the gridworld, the chainworld's absorbing state limits the impact of the number of episode steps on the performance of each method. This is illustrated in Fig. 1(d-f), where we see little impact of the change in maximum steps per episode across the plots in the row. Once the disengagement state is reached, no further learning is possible, even if the agent has not reached the maximum number of steps in the episode.

5.3 BHTPR performs well regardless of the correctness of the prior data.

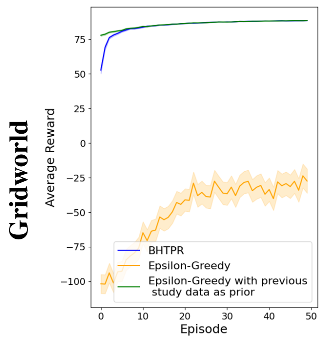
When prior study data are used to form the policy for a new study, this policy will often be suboptimal in certain regions of the state space, for example because the new study is performed on a population that differs from that of the previous study. Since we do not know the number or identity of these states, BHTPR must perform well regardless of the number of states for which the prior study data are incorrect for the new setting. In Fig. 2, we vary the size of the region in which the previous study data matches the current environment. For the states in which the data are corrupted, the environment for the prior study data is randomly generated by selecting the next state for the transition function uniformly at random. In the first column (Figs. 2a and 2d), $\mathcal{D}_{prior}$ matches the environment perfectly, whereas in the next two columns, the prior study data's environment is increasingly different from the target environment. In the gridworld examples, when the prior study data is perfect, using it directly as a prior on the transition function performs best, however BHTPR is very close in performance, greatly outperforming the epsilon-greedy strategy. Note that this is not reflected in the chainworld example (Fig. 2d) because we do not fix exploration to be equal across methods. Hence, BHTPR performs better even when the data is perfect because of the lower amount of exploration, otherwise the baseline in which the previous study data serves as a prior would be favored in this case as well. This highlights how much exploration is driving the performance in the chainworld. When the data are partially correct, our method outperforms the two baselines. Note, however, that with greater regions of incorrect data, the exploration parameter needs to be decayed more slowly otherwise BHTPR will fail to learn the optimal policy.

The performance of BHTPR across the range of data corruption schemes shown here is related to our inductive bias that the data are either entirely correct or entirely wrong in each state. In contrast, in a setting where there is a distribution shift between the prior study environment and the environment of interest, we would expect the baseline in which the previous study data forms a prior to perform well. We confirm this intuition in Fig. 12. In these experiments, we generate an incorrect transition function in the corrupted states by drawing a next state uniformly at random, as before, but then instead of using this transition function directly, we mix the incorrect and correct transition functions together in a range of proportions to create a distribution shift between the previous study environment and the new environment. When the prior is close to the target environment (Fig. 12a), the Bayesian prior baseline performs slightly better; as the mismatch increases, BHTPR outperforms.

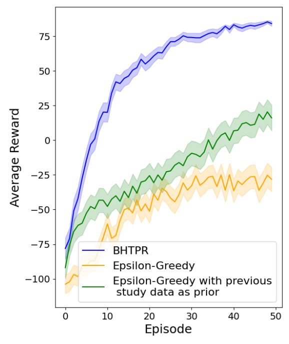
5.4 BHTPR performs well in environments of varying levels of stochasticity.

As illustrated in Fig. 3, reducing the gridworld's stochasticity (making transitions more deterministic) increases the performance advantage of BHTPR over the comparison methods. This behavior reflects the setup of experiments where the amount of prior study data is fixed across examples, so the maximum prior magnitude $T_{prior}(s, a, \cdot)$ is highest for more deterministic environments. To confirm this, we fix the distribution of prior $T_{prior}(s, a, \cdot)$, but vary the magnitude in Fig. 7 (Appx. D). Given the correct prior magnitude, the baseline Bayesian method can outperform; however, our method avoids this parameter tuning and, as demonstrated in the next section, is robust across initializations of $w(s)$.

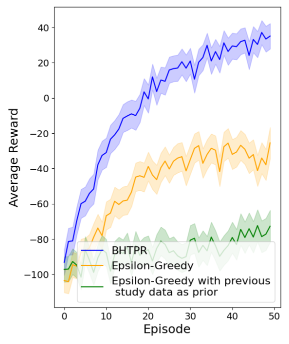
In the chainworld environment, BHTPR performs well across variations in the human's probability of moving, but all methods perform best in the deterministic environment. In the leftmost panel of



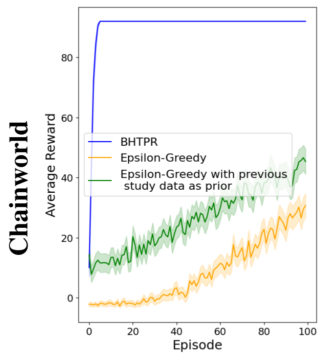
(a) Prior environment matches current (all states).



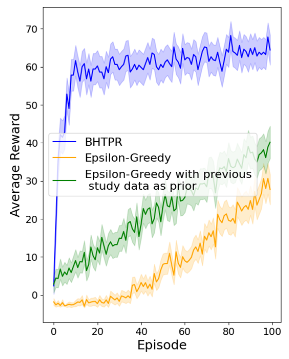
(b) Prior incorrect if x or $y > 4$.



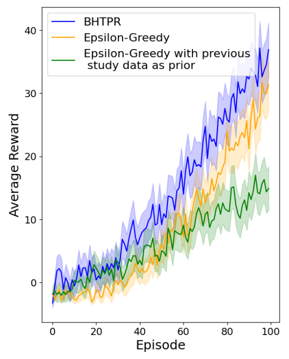
(c) Prior incorrect if x or $y > 2$.



(d) Prior environment matches current (all states).



(e) Prior incorrect for states ≥ 14.



(f) Prior incorrect for states ≥ 8.

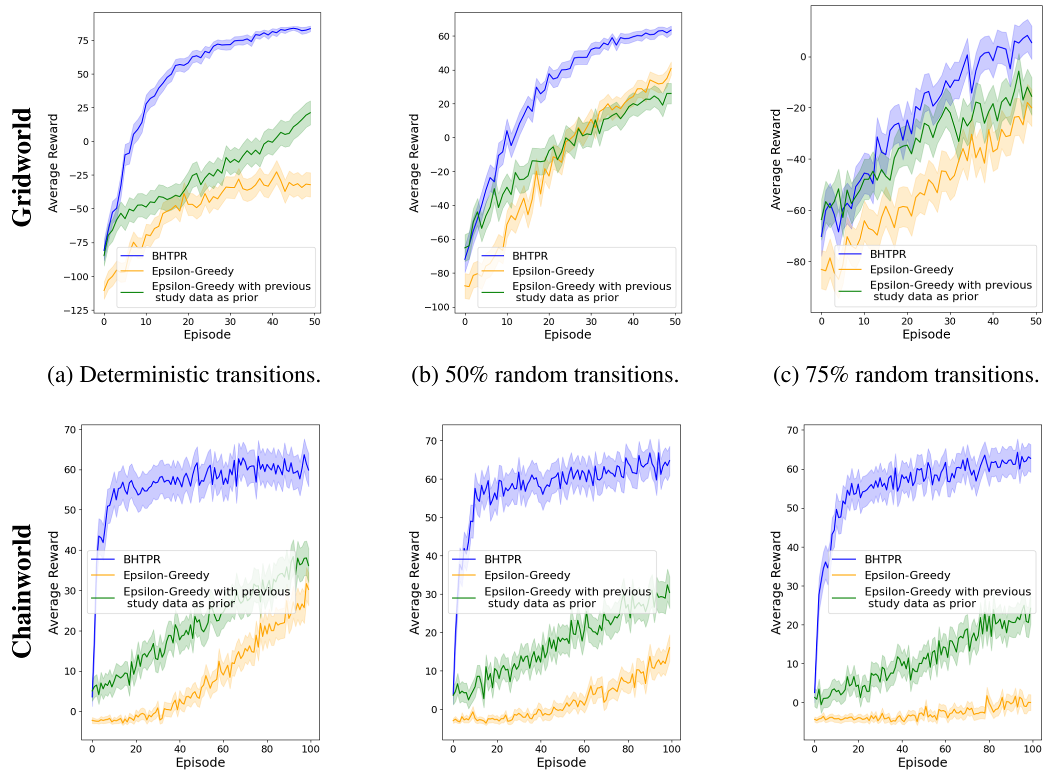
Figure 2: **Correctness of prior study data.** BHTPR performs best when prior and current environments match (a,d), but still outperforms or matches baselines when prior data are partially incorrect.

Fig. 3, the human moves with 100% probability if they take an action, and this probability decreases from left to right. This results in the probability of the human disengaging if they take action 0 decreasing from left to right (d-f). Moving from plot (d) to (e) to (f), the agent will encounter the disengagement state less frequently and consequently receive less feedback about the possibility of disengagement when taking action 0. This makes it harder for the agent to learn from random exploration, and consequently the benefit of injecting outside information is particularly strong in this case.

6 Discussion

Connection to other policy regularization methods As discussed in Sec. 2, BHTPR can be viewed as a form of policy regularization. Wu et al. (2019) define a framework, BRAC, to describe different methods of policy regularization for off-policy RL, and we can view BHTPR as an instance of this framework. BRAC describes policy regularization methods in terms of a penalty on the Q-value objective and/or the policy objective based on how the learned policy diverges from the behavior policy, encouraging similarity between the learned and behavior policies. Similarly, our method also encourages the learned policy to be more similar to π_{prior}, the policy learned from the data of the previous study. Bayesian hypothesis testing provides a principled way to set the strength of the penalty– here the probability of acting according to π_{prior}.

Limitations The main limitation of this method is that it is currently framed in a tabular setting. While it would be easy to use a weight $w(s)$ to interpolate between learned and prior policies where



(a) Deterministic transitions. (b) 50% random transitions. (c) 75% random transitions.

(d) 100% prob. of move. (e) 75% prob. of move. (f) 50% prob. of move.

Figure 3: **Stochasticity.** In gridworld, BHTPR's advantage is largest in deterministic environments. In chainworld, all methods perform better when the human behaves deterministically.

both are defined in a continuous state/action setting, extending the calculation for w itself to a function approximation setting requires further work. We also note that compared to the baselines methods, BHTPR introduces additional computational overhead in terms of updating weight $w(s)$ for all states after each episode. Finally, BHTPR does not account for the transfer of policies between settings where the state spaces are not identical.

7 Conclusion

Learning in a setting with sparse, distal rewards– a situation commonly encountered in mobile health– provides unique challenges. Policies crafted from previous study data or by experts can guide exploration, leading the agent to learn more quickly. Our approach makes explicit the inductive bias that such prior policies are optimal in some parts of the state space and suboptimal in others. By using Bayesian hypothesis testing to determine where this bias holds, our method incorporates prior study data in a principled, adaptive way and speeds up learning across a range of settings.

A Appendix: Bayesian Hypothesis Testing Details

$w(s)$ represents the posterior probability that the observed transition data was generated by $\hat{T}_{prior}$, the current estimate of T_{prior} as opposed to $\hat{T}$. We formalize this idea using Bayesian hypothesis testing, with null an alternative hypotheses defined as follows.

Null hypothesis, H_0 The transition data collected online was generated by the transition function implied by the prior study data, i.e. $T(s,a,\cdot) = T_{prior}(s,a,\cdot)$ for all actions $a \in \mathcal{A}$ in a given state s.

Alternative hypothesis, H_1 The transition data collected online was not generated by the transition function implied by the prior study data; therefore, we calculate the likelihood of the transition data under the transition function being learned online.

Per-episode update for $w(s)$ Using the null and alternative hypotheses above, we set w(s) equal to the posterior probability of the null hypothesis, i.e. the probability that the prior study data is correct in state s.

$$P(H_0|\mathcal{D}) = \frac{P(\mathcal{D}|H_0)P(H_0)}{P(\mathcal{D})} = \frac{P(\mathcal{D}|H_0)P(H_0)}{P(\mathcal{D}|H_0)P(H_0) + P(\mathcal{D}|H_1)P(H_1)}$$

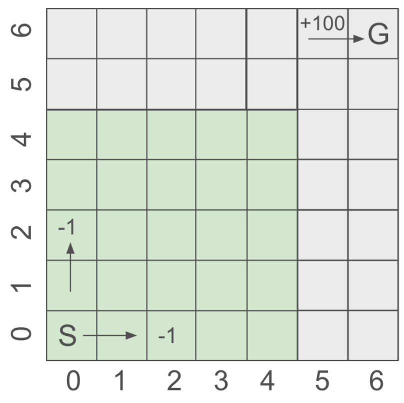
Letting the initial weight $w_0(s)$ be the initial $P(H_0)$, we calculate the likelihood of the data using the transition function calculated from the prior data, $\hat{T}_{prior}$, and the transition function learned online, $\hat{T}$. Then after every episode e the weight $w_e(s)$ is updated accdoring to Eq. 1. Let $\mathcal{D}_e(s)$ be the set of all transition data $\{(s,a,s')\}$ collected in episode e starting from state s.

$$w_{e+1}(s) = \frac{\underbrace{[\Pi_{(s,a,s')\in\mathcal{D}_e(s)} T_{prior}(s,a,s')]}_{\text{likelihood of data under } H_0} \underbrace{w_e(s)}_{P(H_0)}}{\underbrace{[\Pi_{(s,a,s')\in\mathcal{D}_e(s)} T_{prior}(s,a,s')]}_{\text{likelihood of data under } H_0} \underbrace{w_e(s)}_{P(H_0)} + \underbrace{[\Pi_{(s,a,s')\in\mathcal{D}_e(s)} T(s,a,s')]}_{\text{likelihood of data under } H_1} \underbrace{(1-w_e(s))}_{P(H_1)}} \tag{1}$$

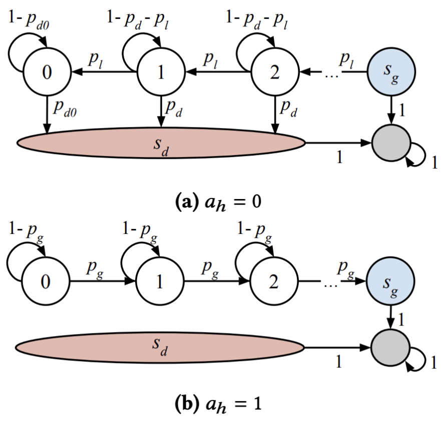
If we further assume that there are regions of the state space in which the current environment matches the prior study environment and others in which it does not, we can improve upon this method by applying smoothing to values of $w_e(s)$ across steps in each episode so that states that are close together have similar weights. In the examples that follow, we apply an exponential moving average over weights at each timestep of an episode.

B Appendix: Environment Details

B.1 Environment Visualizations



(a) Gridworld MDP described in Sec. 4.1



(b) Human MDP from the Chainworld environment. Image from Nofshin et al. (2024).

Figure 4: Two environments used in Sec. 4

B.2 Chainworld Additional Details

The human is modeled by MDP $M_h =< S_h, A_h, T_h, R_h, \gamma_h >$, representing progress towards the task goal. Each state reflects the amount of progress the human has made towards that goal. If the human takes action 0 (Fig. 4b(a)), they may stay in the same state, lose progress towards the goal (move back one state), or disengage (enter state s_d). If instead they take action 1 (Fig. 4b(b)), they will either remain in the same state or progress one state closer to the goal. The human takes the optimal action according to their MDP. If the human disengages, they cannot re-engage.

The AI is represented by MDP $M_{AI} =< S_{AI}, A_{AI}, T_{AI}, R_{AI}, \gamma_{AI} >$. [2] The AI assists the human, who is a myopic decision-maker (low γ_h), at reaching the goal state. To achieve this goal, the AI has two possible actions. It can send an intervention that temporarily increases γ_h (action 1) or do nothing (action 0). Its two-dimensional state space is represented by $S_{AI} = [s_h, a_h]$, the human's state and previous action.

In the experiments below, we consider the AI agent's MDP. Parameters for the human and AI MDPs were chosen such that the agent can reach the goal (i.e. the right combination of agent actions can lead the human to the goal), and such that T_{AI} differs for the AI's two actions (e.g. if the human is not sufficiently myopic, he will take action 1 regardless of AI interaction and BHTPR will have no impact). The environment has 10 human states. Please see Nofshin et al. (2024) for more details on the MDP construction.

One natural way to construct a reward function in the presence of an undesirable absorbing state is with a large negative reward in this state. In this case, the agent receives the large negative reward upon encountering the absorbing state and correctly learns to avoid taking that action again in the given state. Consequently, simple epsilon-greedy methods can learn in this setting. Since immediate feedback is available in the form of a large negative reward every time the agent encounters the disengagement state, this implementation of the chainworld does not represent a sparse, distal reward unless the action probabilities are set such that the disengagement state is not frequently encountered. To reflect a sparse rewards environment that we are primarily concerned with, results below do not

[2]Our algorithm considers an episodic setting where $\gamma_{AI} = 1$.

use a reward function with a large negative reward in the absorbing state (instead $R(s, a) = -1$ in the absorbing state, as in all other states), however results for this variation are available in Appx. E.

C Appendix: Experiment Details

In the experiments, we vary the following elements: stochasticity of the transitions, correctness of the prior study data, and the maximum number of steps per episode. By default in the gridworld, unless otherwise stated, the previous study data are correct for states in which the x and y coordinates are less than 5 (illustrated in green in Fig. 4a), and the maximum episode length is 150 steps. In the chainworld examples, previous study data are correct for states numbered less than or equal to 13 and the maximum episode length is 50. In both, the weights $w_0(s)$ in the algorithm are initialized to 0.5 for all states and transitions are deterministic by default.

D Appendix: Additional Results

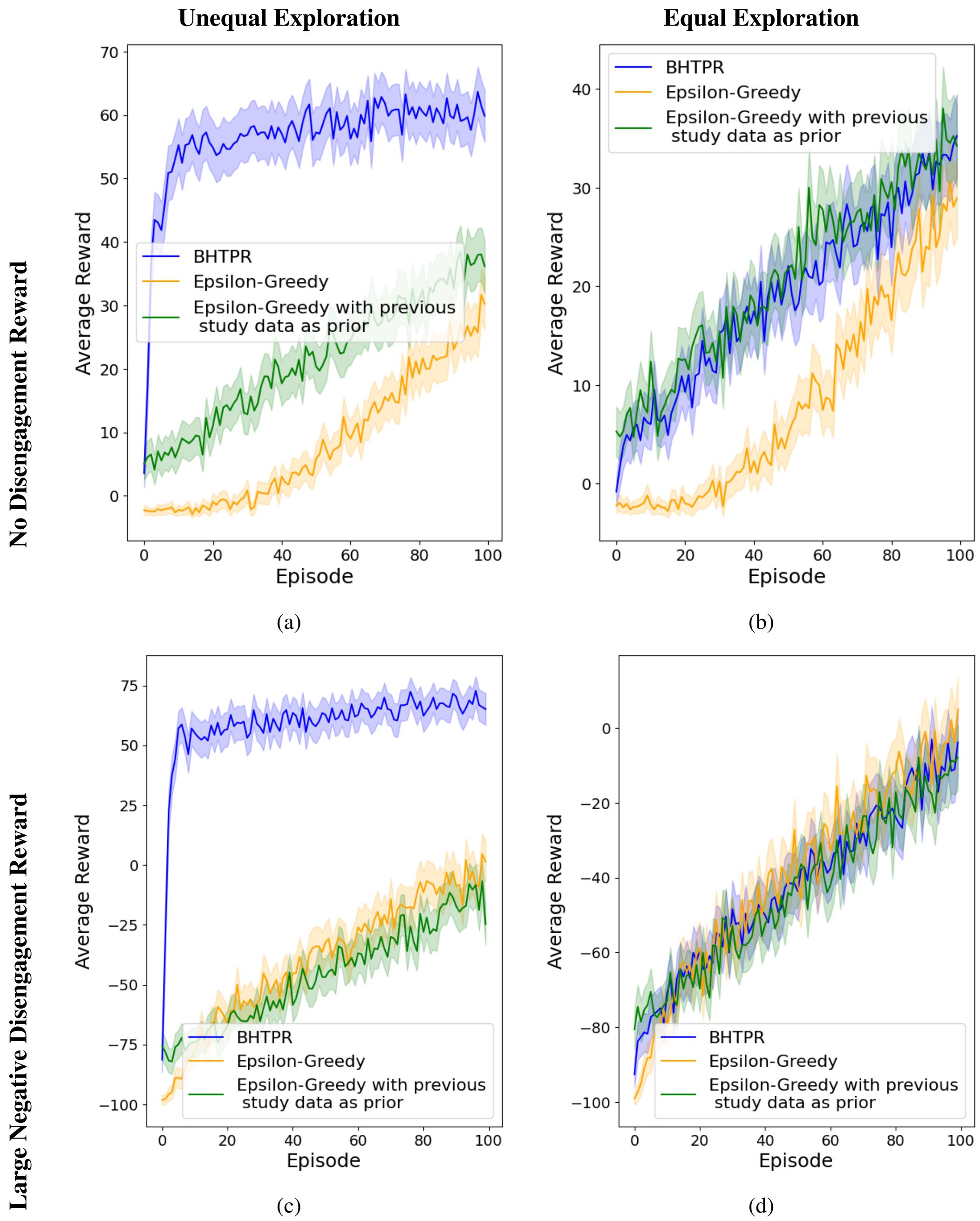


Figure 5: Effect of varying disengagement reward and exploration in the chainworld environment.

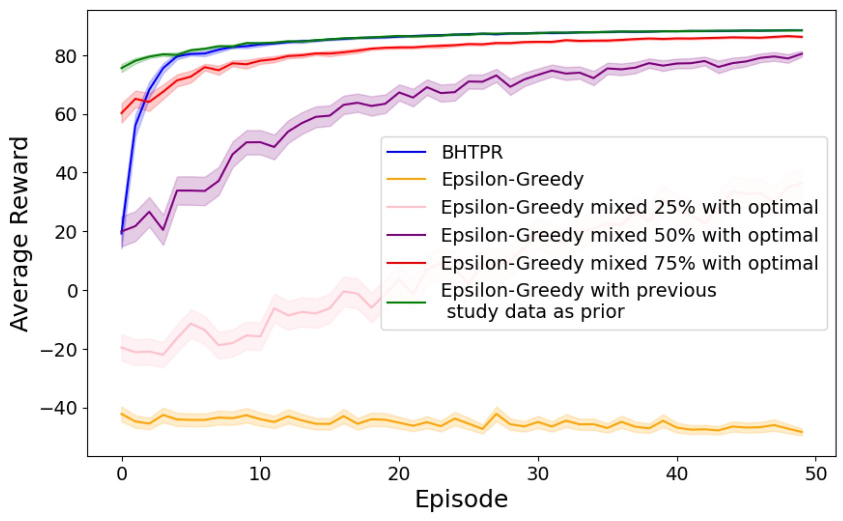


Figure 6: As the percentage of optimal actions mixed with an epsilon-greedy increases, performance approaches BHTPR. Plot from Grid MDP environment, previous study data matches current environment in all states.

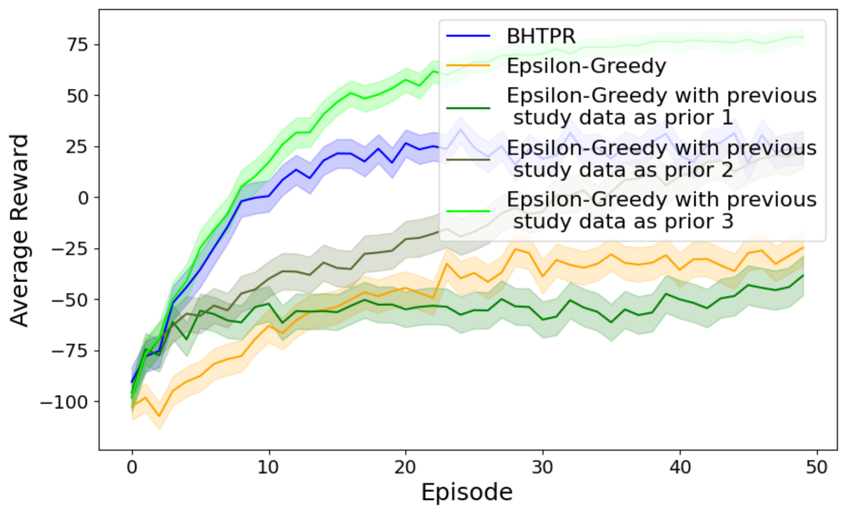


Figure 7: Using the previous study data as a prior on T can outperform BHTPR for the correct magnitude of prior, however BHTRP avoids tuning the magnitude of the prior. Grid MDP environment.

BHTPR is robust to a wide range of initial values for $w_0(s)$ (Fig. 8), though performance can improve slightly with expert-informed initializations.

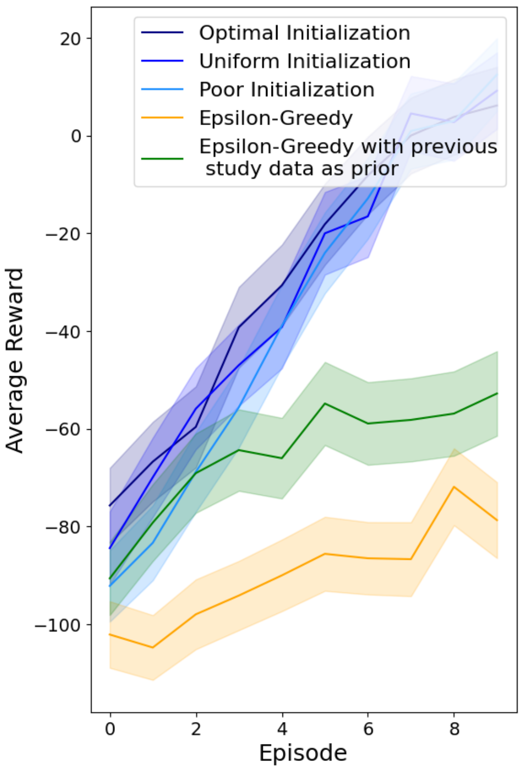
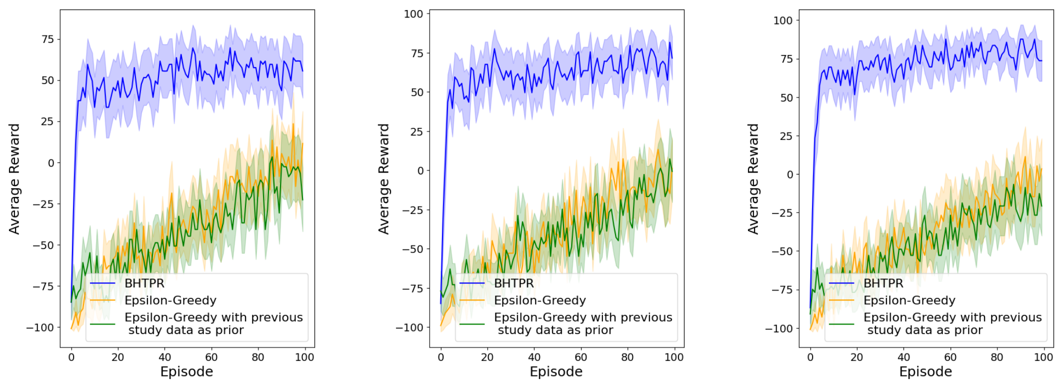


Figure 8: Comparison of initializations for algorithm parameter $w(s)$ in the Grid MDP.

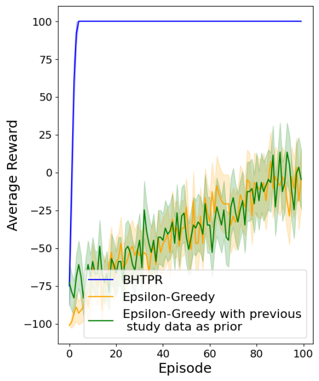
E Appendix: Chainworld results with alternative reward function.

The plots below reflect the same set up as in the body of the paper, but with a large, negative disengagement reward. As discussed in Sec. 4.1, a large negative reward in the disengagement state speeds up learning, especially for the epsilon-greedy baseline.

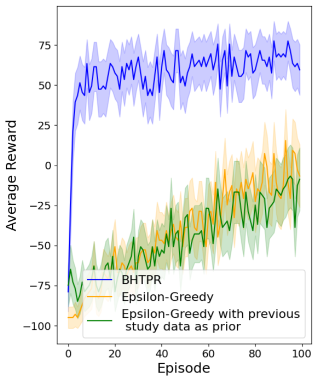


(a) Chainworld, 50-step episodes (b) Chainworld, 30-step episodes (c) Chainworld, 20-step episodes

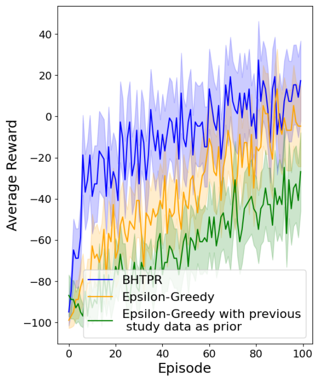
Figure 9: **Episode length.** In the chainworld, maximum episode length does not have a large effect on performance.



(a) Chainworld, perfect prior study data

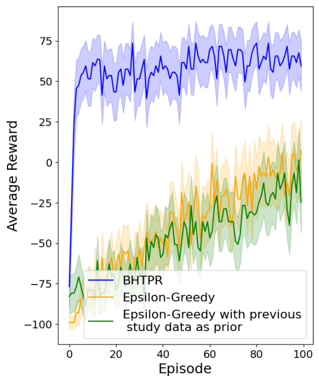


(b) Chainworld, prior study data incorrect for states ≥ 14

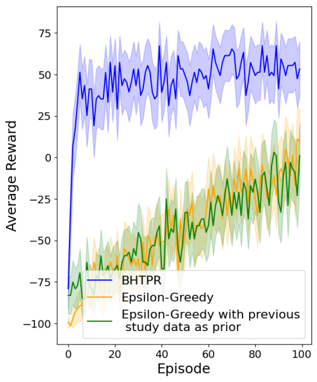


(c) Chainworld, prior study data incorrect for states ≥ 8

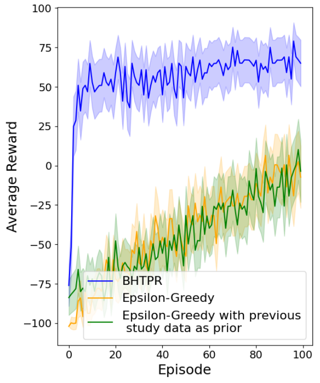
Figure 10: **Correctness of prior study data.** Consistent with the reward function described in the body of the paper, BHTPR performs best when the prior study data matches the current environment (a), but also outperforms or matches baselines when the data does not match the environment of interest in all states.



(a) Second row plot 1

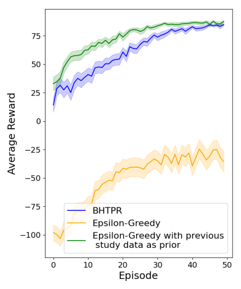


(b) Second row plot 2

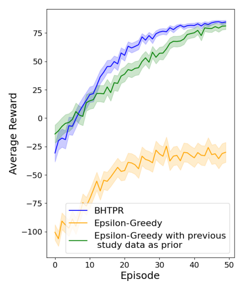


(c) Second row plot 3

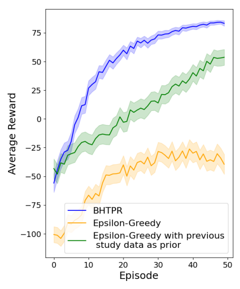
Figure 11: **Stochasticity** Results are similar across levels of stochasticity, with BHTPR performing well in all cases.



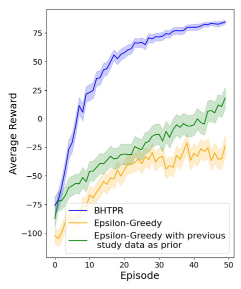
(a) 75% correct 25% incorrect.



(b) 50% correct 50% incorrect.



(c) 25% correct 75% incorrect.



(d) 0% correct 100% incorrect.

Figure 12: **Distribution shift** When the states with corrupted data have a transition distribution that is close to correct, using the data as a prior on $T(s, a, \cdot)$ performs best, however when the data are entirely right or wrong, our method outperforms. Note that in these examples, not all states have corrupted data. It is the transition distribution of the data in the states with corrupted data that varies. Plots from Grid MDP environment.

Acknowledgments

Supported by NIH award 5R01MH123804-02, NIH/NIBIB and OD P41EB028242, and NIH/NIDA P50DA054039. S. A. Murphy holds concurrent appointments at Harvard University and as an Amazon Scholar. This paper describes work performed at Harvard University and is not associated with Amazon.

References

Michael P Fay and Erica H Brittain. *Statistical Hypothesis Testing in Context: Volume 52: Reproducibility, Inference, and Science*, volume 52. Cambridge University Press, 2022.

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

Dylan Hadfield-Menell, Smitha Milli, Pieter Abbeel, Stuart J Russell, and Anca Dragan. Inverse reward design. *Advances in neural information processing systems*, 30, 2017.

Predrag Klasnja, Shawna Smith, Nicholas J Seewald, Andy Lee, Kelly Hall, Brook Luers, Eric B Hekler, and Susan A Murphy. Efficacy of contextually tailored suggestions for physical activity: a micro-randomized optimization trial of heartsteps. *Annals of Behavioral Medicine*, 53(6):573–582, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Alessandro Lazaric. Transfer in reinforcement learning: a framework and a survey. In *Reinforcement Learning: State-of-the-Art*, pp. 143–173. Springer, 2012.

Yiren Lu, Justin Fu, George Tucker, Xinlei Pan, Eli Bronstein, Rebecca Roelofs, Benjamin Sapp, Brandyn White, Aleksandra Faust, Shimon Whiteson, et al. Imitation is not enough: Robustifying imitation with reinforcement learning for challenging driving scenarios. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7553–7560. IEEE, 2023.

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

Eura Nofshin, Siddharth Swaroop, Weiwei Pan, Susan Murphy, and Finale Doshi-Velez. Reinforcement learning interventions on boundedly rational human agents in frictionful tasks. In *Proceedings of the... International Joint Conference on Autonomous Agents and Multiagent Systems: AAMAS. International Joint Conference on Autonomous Agents and Multiagent Systems*, volume 2024, pp. 1482, 2024.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

Satinder Singh, Richard L Lewis, and Andrew G Barto. Where do rewards come from. In *Proceedings of the annual conference of the cognitive science society*, pp. 2601–2606. Cognitive Science Society, 2009.

Henry Sowerby, Zhiyuan Zhou, and Michael L Littman. Designing rewards for fast learning. *arXiv preprint arXiv:2205.15400*, 2022.

Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

Anna L Trella, Kelly W Zhang, Inbal Nahum-Shani, Vivek Shetty, Finale Doshi-Velez, and Susan A Murphy. Reward design for an online reinforcement learning algorithm supporting oral self-care. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 15724–15730, 2023.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Kelly W Zhang, Omer Gottesman, and Finale Doshi-Velez. A bayesian approach to learning bandit structure in markov decision processes. *arXiv preprint arXiv:2208.00250*, 2022.