Autonomous Assessment of Demonstration Sufficiency

Alina Trinh

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2024-89
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2024/EECS-2024-89.html

May 10, 2024

Acknowledgement

Thank you to my advisor Professor Stuart Russell for your insightful discussions and mentorship throughout the years. Thank you to Professor Daniel S. Brown for your never-ending guidance and encouragement that have helped me grow immensely as a researcher. Thank you to all my labmates, colleagues, and co-authors at BAIR, CHAI, and the University of Utah for showing me the ropes and inspiring me every day. And thank you to my family, for their unwavering love and support throughout my academic journey are the foundations of my success today.

AUTONOMOUS ASSESSMENT OF DEMONSTRATION SUFFICIENCY

by

Tu Trinh

A thesis submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Stuart J. Russell, Advisor
Professor Pieter Abbeel, Second Reader

Spring 2024

The thesis of Tu Trinh, titled AUTONOMOUS ASSESSMENT OF DEMONSTRATION SUFFICIENCY, is approved:

Advisor Stuart Russell Date 5/1/24

Second Reader PAbbeel Date 5/10/24

University of California, Berkeley

AUTONOMOUS ASSESSMENT OF DEMONSTRATION SUFFICIENCY

Abstract

AUTONOMOUS ASSESSMENT OF DEMONSTRATION SUFFICIENCY

by

Tu Trinh

Master of Science in Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Stuart J. Russell, Advisor

Professor Pieter Abbeel, Second Reader

In this thesis we examine the problem of **demonstration sufficiency**: how can an agent self-assess whether or not it has received enough demonstrations from an expert to ensure a desired level of performance? To address this problem, we propose a novel self-assessment approach based on Bayesian inverse reinforcement learning and value-at-risk, enabling learning-from-demonstration ("LfD") agents to compute high-confidence bounds on their performance and use these bounds to determine when they have received a sufficient number of demonstrations. We propose and evaluate two definitions of sufficiency: (1) normalized expected value difference, which measures regret with respect to the human's unobserved reward function, and (2) percent improvement over a baseline policy. We demonstrate how to formulate high-confidence bounds on both of these metrics. We evaluate our approach in simulation in both discrete and continuous state-space domains and illustrate the feasibility of developing a robotic system that can accurately evaluate demonstration sufficiency. We also show how the agent can utilize active learning in asking for demonstrations from specific states which results in fewer demos needed for the agent to still maintain high confidence in its policy. Finally, via a user study, we show that our approach successfully enables agents to accomplish tasks at users' desired performance levels, without needing too many or perfectly optimal demonstrations. This thesis is an extended version of [49].

To my family

without whom I would not be submitting a thesis today. Thank you for all of the lessons, laughs, and love you have given me throughout all these years. Thank you for continuously inspiring me to strive for the best version of myself and to stay curious about the world around me. I love you!

Contents

Contents ii

List of Figures iii

List of Tables iv

Introduction 1

Related Work 3

Preliminaries 3

Method 6

Empirical Results 9

Methodology Extensions 14

User Study 16

Discussion 20

Bibliography 22

List of Figures

1 *Demonstration sufficiency:* Pictured is an illustrative living room in which the demonstrator is seeking to teach the agent to reach the coffee table while avoiding the expensive rugs. That is, the table has large positive reward and the rugs have large negative reward. (Left) The agent receives two demonstrations (the black arrows) from the human but, through our self-assessment method discussed below, deems them insufficient for it to be highly confident that its learned policy has low regret–it does not yet have strong evidence about the relative rewards of the different features in the room. (Right) The agent receives an additional two demonstrations and, after applying our method, deems them sufficient to guarantee with high confidence that its learned policy will have low regret if evaluated under the unobserved, true reward function. 2
2 The environments in which we examine demonstration sufficiency assessment. . 9
3 *nEVD method compared to baseline methods for determining demonstration sufficiency:* The x-axis across all subfigures denote the nEVD bound threshold the agent was using to assess demonstration sufficiency with our method. "Conv." denotes the convergence baseline with a patience hyperparameter of p. "V.S." is the validation set baseline with an interval hyperparameter of i. Subfigures (a) and (b) show the sample efficiency and identification accuracy measures, respectively, for the discrete domain (gridworld and driving). Subfigures (c) and (d) show the sample efficiency and identification accuracy measures, respectively, for the continuous domain (lander and lavaworld). Bands around the lines denote standard error. 11
4 *User study.* The waiting screen as the agent is calculating on a gridworld environment. 17
5 *User study.* The agent requests another demonstration in the driving environment. 18
6 *User study.* The agent's final learned policy is displayed in white and the user is asked for a ranking. 19

List of Tables

1 Number of demonstrations required under a 95% confidence Hoeffding-based bound to reach each nEVD threshold. Our method (showcased in Figure 3) requires orders of magnitude fewer demonstrations to reach the same bounds. . 13

2 *PIOB method:* F1 scores for the two domains, for (a subset of) each percent improvement bound threshold used. 15

3 *Passive vs. active demonstration selection:* The mean and standard deviation of reduction in proportion of states needed between active and passive demonstration selection. 16

4 *User study results:* The three methods' performances in our user study. We still use proportion of states to assess sample efficiency. However, instead of using F1 score for identification accuracy, we use user evaluation: now that we have real humans, this gives a better sense of whether or not the agent's policy is actually aligned with the human's intent. 19

Acknowledgments

Thank you to my advisor Professor Stuart Russell for your guidance and always insightful discussions. From being a student in your introduction to AI class to being a graduate student in your lab, I am very grateful and consider myself extremely lucky to have had your mentorship throughout these years.

Thank you to Professor Daniel S. Brown (University of Utah) for your patient support and undying encouragement when I first started conducting research in AI systems. I have grown immensely as a researcher under your wing. Thank you to Haoyu Chen for sharing your expertise so generously. I am proud to present some of our collaborative work here in this thesis.

Thank you to my co-authors, labmates, and colleagues for inspiring and exciting me every day: Khanh Nguyen, Ben Plaut, Sam Toyer, Cam Allen, Justin Svegliato, Erik Jenner, Shreyas Kapur, Jonathan Stray, Olivia Watkins, Alex Souly, Yun Lu, Dillon Bowen, Scott Emmons, Edmund Mills, Ian Baker, Cassidy Laidlaw, Niklas Lauffer, Bhaskar Mishra, Sana Pandey, Elvis Hsieh, and Jiahai Feng. Research is never a solo journey and you have been the best collaborators. I will dearly miss our daily lunches.

1 Introduction

Imagine an agent that must learn how to navigate across uneven terrains, help scientists conduct chemistry lab experiments, or assemble materials in a construction site. These are all examples of safety-critical settings where it is crucial that the agent's likelihood of failure is near zero, but also where the environment and task may be too complex for the human to decompose into a comprehensive reward function for use in standard (deep) reinforcement learning techniques to steer the agent's behavior. In such cases we turn to learning from demonstration (LfD) [3, 43] where we can teach the agent how best to accomplish these tasks by providing examples in action. The key question here becomes, how many demonstrations must we provide before we can be confident that the agent has learned the task successfully? We can try to generate as many demonstrations as possible, feed them to the agent, and cross our fingers hoping that they will be enough. But what will happen if they are uninformative, ambiguous, or missing important states, conditions, or trajectories? Should we resign to closely monitoring the agent every time it attempts to perform the task, negating the benefits of an otherwise semi-autonomous assistant?

Such considerations focus on how the *human* can be responsible for the agent's learning success, which can require much time, effort, and resources. In this thesis, we focus on how some of this burden can be lifted from the human and placed on the AI agent instead. Particularly, we examine the problem of **demonstration sufficiency**, in which the agent can self-assess whether or not it has received enough training examples in order to achieve a desired level of performance on a given task—even if it does not know the demonstrator's true, intended reward function.

Our main insight is the following: maintaining a belief distribution over the demonstrator's true, but unobserved, reward function, enables an agent to reason about its performance under this distribution and determine, with high-confidence, when it has received enough demonstrations to satisfy a desired performance threshold.

To maintain a belief distribution over reward functions, we propose a novel application of Bayesian IRL (BIRL) [42] that uses samples from the posterior distribution over reward functions, given demonstrations, to enable the agent to evaluate its current learned policy and determine how confident it is that this policy has sufficiently good performance. We propose two definitions of demonstration sufficiency: (1) whether, with high confidence, the learned policy has low regret compared to the optimal policy under the unobserved reward function of the demonstrator and (2) whether the learned policy will, with high confidence, outperform a given baseline policy (e.g., a policy that is known to be safe but is suboptimal) by a desired margin. Our approach allows an agent to self-assess when it has received enough demonstrations to enable it to meet one of the above performance criteria.

By proposing a Bayesian approach to demonstration sufficiency self-assessment, we encourage agents to properly reason about and under uncertainty. For example, if the human demonstrator happens to provide redundant or ambiguous demonstrations, the agent will have a high level of uncertainty regarding the humans' true intention, leading it to continue to ask for additional demonstrations. Each time the agent receives a new demonstration, it can

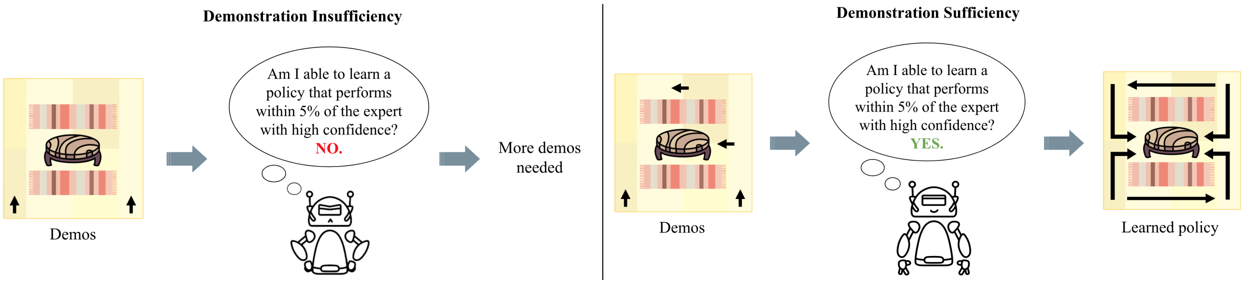

Demonstration Insufficiency
Demos
Am I able to learn a policy that performs within 5% of the expert with high confidence?
NO.
More demos needed
Demonstration Sufficiency
Demos
Am I able to learn a policy that performs within 5% of the expert with high confidence?
YES.
Learned policy

Figure 1: *Demonstration sufficiency:* Pictured is an illustrative living room in which the demonstrator is seeking to teach the agent to reach the coffee table while avoiding the expensive rugs. That is, the table has large positive reward and the rugs have large negative reward. (Left) The agent receives two demonstrations (the black arrows) from the human but, through our self-assessment method discussed below, deems them insufficient for it to be highly confident that its learned policy has low regret–it does not yet have strong evidence about the relative rewards of the different features in the room. (Right) The agent receives an additional two demonstrations and, after applying our method, deems them sufficient to guarantee with high confidence that its learned policy will have low regret if evaluated under the unobserved, true reward function.

then reassess its uncertainty and evaluate its performance under the posterior distribution of likely rewards. Eventually, one of two outcomes can occur:

- The agent determines that the current demonstrations have been sufficient in teaching it the task and signals to the demonstrator that it does not require any further training examples. It is now ready to be deployed (or transferred to subsequent training stages or testing stages).

- The agent determines that all of the demonstrations provided are still insufficient. It remains unconfident it can perform the task as desired and abstains from doing anything hasty or unsafe.

Figure 1 shows an illustrative example of this process.

An important benefit of this self-assessment approach is that it removes the need for the human to predict when the agent has had enough training data. Indeed, it is often difficult for humans to inspect an agent's policy or learned reward function to determine whether it is aligned with their intent. We argue that agents should instead be able to self-assess their performance, relative to their uncertainty over the human's intent.

In the following pages, we formalize the problem of demonstration sufficiency assessment, derive how an agent can self-assess its performance using the two stopping conditions proposed above, and evaluate our approach across several domains using both simulated demonstrations and human-provided demonstrations in a user study.

2 Related Work

This work falls under the area of autonomous self-assessment of agents and other AI systems. Previous work examines how an agent can assess its performance and communicate its shortcomings to a human expert [40, 17]; however, most existing performance metrics and studies do not involve learning from demonstration, and those that do, focus on communication and knowledge-sharing [32, 27] rather than addressing how an AI agent can directly self-assess whether a learned policy or reward function is above a desired safety threshold.

Another prior work [7] studies optimal stopping for agent teaching but uses information gain from pairwise preferences instead of policy performance estimated from demonstrations. Other works have looked at knowing when to stop collecting demonstrations for behavioral cloning when performing sim2real transfer [45] or knowing when an agent can guarantee high confidence itself in learning to grasp an unknown object [22], but they do not consider the inverse reinforcement learning setting considered in this thesis.

Prior work does consider high-confidence performance bounds for inverse reinforcement learning [1, 47]. However, the bounds obtained by these methods are generally loose and correlate to a high number of training examples needed to show the agent. We build off more recent work [13, 15, 14] that demonstrate tighter bounds on performance but do not consider how these bounds can be used for autonomous assessment of demonstration sufficiency.

Finally, our work is also related to pedagogic teaching by demonstrations which studies how to craft demonstrations that will be maximally informative [18, 16, 31, 51, 33]; however, such prior work only considers this problem from the teacher's perspective and assumes that the teacher has privileged information about the student's learning algorithm, as well as complete knowledge of the reward function they seek to teach. By contrast, we focus on developing algorithms from the student's perspective, i.e., algorithms that allow the agent to know when it has received sufficient demonstrations, without any assumptions about the demonstrations being highly informative.

3 Preliminaries

Markov Decision Processes

We model the environments that our agent (AI system, robot, etc.) interacts with as a **Markov decision process** (MDP). An MDP is a five-tuple $\mathcal{M} = \langle S, A, T, R, \gamma \rangle$ consisting of

- S: the set of states in the environment, following an initial state distribution S_0
- A: the set of actions the agent can take from each state
- T: a transition function $S \times A \times S \to [0, 1]$ that denotes the probabilities of landing in a state s' after the agent takes action a from state s

- R: a reward function, either $S \rightarrow r$, the reward an agent obtains by arriving at a state s (the definition this work uses), or $S \times A \rightarrow r$, the reward it gets by taking action a from state s, or $S \times A \times S \rightarrow r$, the reward it gets by taking action a from state s and landing in state s'
- γ: a discount factor $\in (0, 1)$ that controls how future rewards are discounted from their original value

Following prior work [1, 53, 13, 29, 4], we assume that the reward function R can be defined in terms of a linear combination of features: for an MDP with features $\phi(s) \in \mathbb{R}^k$, $R(s) = w^T\phi(s)$ where $w \in \mathbb{R}^k$ is a vector of feature weights.

A **policy** π is a mapping from states to a probability distribution over actions. Each state has a value, defined as

$$V_R^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma_t R(s_t) | s_0 = s \right]$$

or the expected discounted reward to be gained by starting in that state. The total discounted reward expected to be gained by following a particular policy π is denoted as

$$V_R^\pi = \mathbb{E}_{s \sim S_0} V_R^\pi(s)$$

Values are calculated not only for states but also for state-action pairs. The Q-value function for a state-action pair denotes expected discounted reward to be gained by starting in a state s and taking action a. It is defined as

$$Q_R^\pi(s, a) = R(s) + \gamma \sum_{s' \in S} T(s, a, s') V_R^\pi(s')$$

Given this, we can obtain the optimal policy for an MDP—the state to action distribution mapping that will yield the highest expected reward—as

$$\pi^*(s) = \operatorname*{argmax}_a Q(s, a)$$

for each state.

Bayesian Inverse Reinforcement Learning

In **inverse reinforcement learning** (IRL) [53, 4], we seek the unknown, underlying reward function of an MDP given demonstrations [39]. We denote a set of demonstrations by D, which we define to be a set of state-action pairs: $D = \{(s_1, a_1), \ldots, (s_n, a_n)\}$. **Bayesian inverse reinforcement learning** (BIRL) [42] is a form of IRL that estimates the posterior distribution over reward functions given demonstrations, $P(R|D) \propto P(D|R)P(R)$, where the demonstrator is assumed to follow a softmax policy, leading to the following likelihood function:

$$P(D|R) = \prod_{(s,a)\in D} P((s,a)|R) = \prod_{(s,a)\in D} \frac{e^{\beta Q^*_R(s,a)}}{\sum_{b\in A} e^{\beta Q^*_R(s,b)}} \tag{1}$$

where $\beta \in [0, \infty)$ represents the confidence in the demonstrator's optimality (a higher β means the demonstrator is more likely to give optimal demonstrations) and $Q^*_R(s,a) = \max_\pi Q^\pi_R(s,a)$ is the optimal Q-value for a state and action under the reward function R. Equation (1) assigns higher likelihoods to demonstrated actions that result in higher Q-values under R compared to alternative actions. Eq. (1) is an example of **Boltzmann rationality**, a model that has found widespread utility in economics [11, 36], psychology [5, 24, 25], and AI [54, 20, 8, 21, 28, 34] as a useful model of human decision-making and can be seen as the maximum entropy distribution over choices for a satisficing agent [28].

Markov Chain Monte Carlo Sampling

Markov chain Monte Carlo (MCMC) sampling [38] is a statistical technique used to generate samples from a probability distribution P by constructing a Markov chain that has P as its equilibrium distribution. The Metropolis-Hastings algorithm [26], a widely used MCMC method, enables sampling when the form of the target distribution P is known only up to a normalizing constant. At each iteration of the algorithm, a new sample is drawn from a proposal distribution $\tilde{P}$ centered at the current sample and is accepted with some probability dependent on the ratio of the two samples' likelihoods under P. Most applications of BIRL, including ours, use Metropolis-Hastings in order to sample from the posterior $P(R|D)$ [42], assuming a uniform prior distribution, though this distribution can take on any form depending on the domain and can be a way for human demonstrators to inject domain knowledge into the agent.

Value-at-Risk

Value-at-risk is a probabilistic measure of worst-case performance [30, 48]. The α-Value-at-Risk, or α-VaR, is the α-worst-case value of a random variable Z, where $\alpha \in (0, 1)$ is the quantile level. This is defined as

$$\nu_\alpha(Z) = F_Z^{-1}(\alpha) = \inf\{z : F_Z(z) \geq \alpha\} \tag{2}$$

where $F_Z(z) = P(Z \leq z)$, the cumulative distribution function of Z. The higher the value of α, the more risk-sensitive we are.

4 Method

Problem Definition

Our aim is to determine whether or not an agent has received sufficient demonstrations in order to complete a task in a way that aligns with the expert's intended policy derived from their unobserved reward function R^*. We first study how agents can quantify their assessment of the goodness of their policy compared to the expert's by using high-confidence bounds on *regret*. The agent should request more demonstrations if it is not yet highly confident that its learned policy will have low regret compared to the expert's, and it should declare demonstration sufficiency if it is highly confident.

Formally, given an MDP with an unobserved reward function R^*, a set of demonstrations D, a confidence parameter α, and a performance threshold ϵ, we want the agent to be able to determine when it is α-confident that its policy regret, if evaluated under the demonstrator's true reward function R^*, is no worse than ϵ. Thus, demonstration sufficiency is achieved when

$$P\left(\text{regret}(\pi_{\text{agent}}, R^*) \leq \epsilon \,|\, D\right) \geq \alpha \tag{3}$$

In this work, we use $\alpha = 0.95$ and provide results for varying values of ϵ. Note that in practice, π_{agent} can be any agent policy. In our work, we set π_{agent} to be π_{MAP}, the optimal policy corresponding to the maximum a posteriori reward estimate R_{MAP} learned by the agent during Bayesian IRL. This policy is retrieved using value iteration and policy extraction.

Determining Demonstration Sufficiency

We must now select a measure of regret. Prior work on IRL [42, 35, 50, 13, 15] has typically used a measure of policy regret (also known as "policy loss") called the **expected value difference** (EVD), defined as

$$EVD(\pi_{\text{agent}}, R^*) = V^*_{R^*} - V^{\pi_{\text{agent}}}_{R^*} \tag{4}$$

While this is a common metric for comparing different IRL algorithms, reward functions are equivalent under positive scaling and affine shifts [39, 2], making a threshold defined in raw reward units likely uninterpretable to human demonstrators and other stakeholders. Thus, we propose the use of demonstration sufficiency thresholds defined in terms of **normalized expected value difference** (nEVD):

$$\text{regret}(\pi_{\text{agent}}, R^*) := nEVD(\pi_{\text{agent}}, R^*) = \frac{V^*_{R^*} - V^{\pi_{\text{agent}}}_{R^*}}{V^*_{R^*} - V^{\pi_{\text{rand}}}_{R^*}} \tag{5}$$

where π_{rand} is a uniform random policy. Normalizing with respect to a random uniform policy enables the demonstrator to specify a regret threshold in terms of a more interpretable percentage rather than in raw reward units. It also offers an additional dimension for comparison as it conveys how much the agent's learned policy deviates from the optimal policy compared to a completely random one (which should be relatively very suboptimal).

Next, because the ground-truth reward function R^* is unknown to the agent, it is impossible to calculate its true regret, $nEVD(\pi_{\text{agent}}, R^*)$. Instead, to perform demonstration sufficiency self-assessment, we propose a Bayesian approach that leverages Bayesian IRL to sample from $P(R|D)$, the posterior distribution of reward functions given demonstrations, then uses these reward samples to calculate an α-Value-at-Risk (α-VaR) upper bound on regret [13, 15]. Then, the agent should declare demonstration sufficiency when

$$\nu_\alpha\left(nEVD(\pi_{\text{agent}}, R)\right) \leq \epsilon, \text{ for } R \sim P(R|D) \tag{6}$$

Dealing with Finite Sampling Errors

As of now, our goal is to find an α-quantile worst-case bound on $nEVD(\pi_{\text{agent}}, R^*)$ by computing the α-VaR over $P(R|D)$. In practice, since we do not know $P(R|D)$ explicitly, we must obtain samples from the posterior, $\mathcal{R} = \{R \sim P(R|D)\}$, via Markov chain Monte Carlo (MCMC) methods [42]. Thus, we need to be careful about the error induced by samples and make sure that we do not underestimate the policy regret due to merely sampling from the posterior.

Recall that rewards are represented by feature weights w. These weights $\hat{w}_i$ are sampled according to a normal proposal distribution with mean $\hat{w}_{i-1}$ and standard deviation σ. We normalize them such that $\|\hat{w}\|_2 = 1$ to guarantee unique proposals, as there can be infinitely many reward functions representing the same environment beliefs if one applies a scaling factor. We implemented an adaptive version of MCMC where σ is automatically tuned during the sampling process. If the current accept rate r is higher than a target accept rate r^*, the step size will decrease by $\Delta\sigma$; if it is lower, the step size will increase by $\Delta\sigma$, where $\Delta\sigma = \frac{\sigma}{\sqrt{i+1}}(r - r^*)$ and i is the index of the current MCMC sample or iteration.

Bayesian IRL has rapid finite-time mixing guarantees and converges to the true posterior, making it a viable method to estimate $P(R|D)$ [42], but we still need to deal with error and uncertainty when estimating the value-at-risk. We do this as follows.

For each sample $R_i \sim P(R|D) \in \mathcal{R}$ we first compute

$$X_i = nEVD(\pi_{\text{agent}}, R_i) \tag{7}$$

giving us samples from the posterior distribution of normalized expected value differences conditioned on the human-provided demonstrations. Given n samples of X, we can obtain a point estimate of the α-VaR by sorting $X_1, \ldots, X_n$ in ascending order to get order statistics Z, then take the α-quantile. This gives us Z_k as an estimate of the α-VaR, where $k = \lceil \alpha n \rceil$.

However, simply setting $k = \lceil \alpha n \rceil$ does not incorporate our confidence in this α-VaR point estimate. So, we follow Brown et al. [13] to do so, using a high-confidence threshold of $\delta = 0.95$. By definition, we have that $P(X_i < \nu_\alpha(X)) = \alpha$ for any sample X_i, $i \in 0, \ldots, n$. Having sorted these samples to obtain order statistics Z_j, $j \in 0, \ldots, n$, we can calculate

Algorithm 1: Demonstration Sufficiency (nEVD)

```
Calculate the $\alpha$-VaR bound index $k = \lceil n\alpha + F_{\mathcal{N}}^{-1}\delta\sqrt{n\alpha(1-\alpha)} - \frac{1}{2}\rceil$
for $j = 0, 1, 2, \ldots$ do
    Collect a new demo and add it to $D$
    Using MCMC, compute $R_{\text{MAP}}$ and obtain randomly sampled rewards
      $R_1, R_2, \ldots, R_n$
    Run value iteration on $R_{\text{MAP}}$ and extract policy $\pi_{\text{MAP}} =: \pi_{\text{agent}}$
    for $i = 1, \ldots, n$ do
        Perform policy evaluation of $\pi_{\text{agent}}$ on reward sample $R_i$ to obtain $V_{R_i}^{\text{agent}}$
        Perform policy evaluation of $\pi_{\text{rand}}$ on reward sample $R_i$ to obtain $V_{R_i}^{\text{rand}}$
        Run value iteration on $R_i$ to obtain $V_{R_i}^*$
        Calculate $nEVD_i = \frac{V_{R_i}^* - V_{R_i}^{\text{agent}}}{V_{R_i}^* - V_{R_i}^{\text{rand}}}$
    end
    Sort $\{nEVD_i\}_{i=1}^n$ and find the $\alpha$-VaR bound $nEVD_{(k)}$
    if $nEVD_{(k)} > \epsilon$ then
        Repeat
    else
        Stop
    end
end
```

the probability for any Z_j that the α-VaR is less than Z_j using the binomial cumulative distribution function (CDF):

$$\begin{aligned} P(\nu_\alpha(X) < Z_j) &= F(j-1; n, \alpha) && (8) \\ &= \sum_{i=0}^{j-1} \binom{n}{i} \alpha^i (1-\alpha)^{n-i} && (9) \end{aligned}$$

Note that $\nu_\alpha(X)$ is the 100α percentile value of X. Thus, for the order statistic Z_j to be larger than $\nu_\alpha(X)$, we must have that $\nu_\alpha(X)$ is greater than at most $j-1$ samples. This probability is given by the binomial CDF, $F(j-1; n, \alpha)$, which gives the probability of getting $j-1$ or fewer successes in n trials (hence Eq. (8)). In this formulation, a success is when a sample X_i is less than $\nu_\alpha(X)$, making the probability of success, $P(X_i < \nu_\alpha(X))$, equal to α by definition of α-VaR; it follows that the probability of failure, $P(X_i \geq \nu_\alpha(X))$, is $1-\alpha$, hence Eq. (9). Finally, to get a $\delta = 95\%$ confidence bound on $\nu_\alpha(X)$, we can use the inverse binomial CDF, F^{-1}. Thus, the order statistic Z_k, where $k = F^{-1}(0.95; n, \alpha)$, forms a 0.95-confidence bound on $\nu_\alpha(X)$. We use the above derivation to compute 95%-confidence bounds on the α-VaR throughout this work.

Algorithm 1 above shows a succinct psuedocode of our high-confidence nEVD bounding method. $F_{\mathcal{N}}^{-1}$ is the inverse Gaussian distribution.

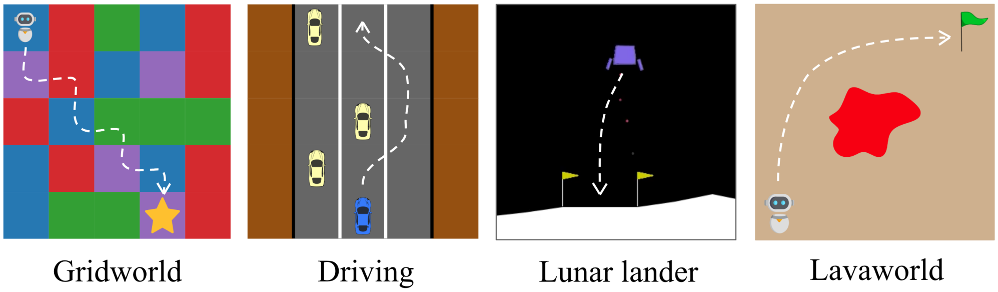


Figure 2: The environments in which we examine demonstration sufficiency assessment.

5 Empirical Results

Experimental Design

Figure 2 shows the environments we use to test our methodology. Two have discrete state spaces (Gridworld and Driving) and two have continuous state spaces (Lunar Lander and Lavaworld). We generated multiple randomized MDP instances of each environment and tested three methods on the same set of MDPs: our approach and two baselines, discussed below.

In each environment we simulate human demonstrations by sampling states uniformly at random and providing an optimal action for that state (discrete environments) or optimal trajectory starting from that state (continuous environments). One demonstration is given at each iteration the agent has not yet declared demonstration sufficiency. Our test environments are:

- **Gridworld:** A discrete state space $M \times N$ environment where each state has one of four features, each associated with a different reward weight. One of these states is the goal state. The agent can take one of four actions: up, down, left, and right.

- **Driving:** A discrete, infinite horizon environment with different road conditions and traffic. There are two off-road patches on either side of the main roads. The end of the road segment is connected to the beginning, simulating a continuous road. Actions are drive straight, move to left lane, and move to right lane.

- **Lunar lander [12]:** A continuous state space environment from OpenAI Gym where a craft attempts to land on a specific landing pad on the moon.

- **Lavaworld [29]:** A continuous state space environment where the agent must navigate towards a goal while both avoiding a pit of lava that appears randomly in the environment and maintaining a smooth, non-jerky trajectory.

Baselines

To the best of our knowledge, we are the first to study demonstration sufficiency for LfD agents. Thus, we adapt two stopping criteria from supervised learning [41] into heuristic baselines to compare with our approach.

- **Convergence (Conv.)**: Given a "patience" hyperparameter p, the agent signals demonstration sufficiency when its policy π_{MAP} does not change over p consecutive demonstrations.
- **Validation set (V.S.)**: Every i^{th} demonstration is added to a held-out set. If for each (s, a) in the held-out set, $\pi_{\text{MAP}}(s) = a$, the agent declares demonstration sufficiency.

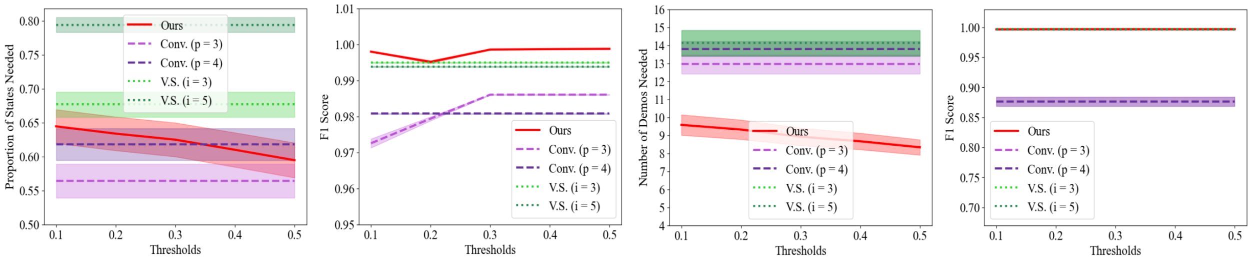
Dependent Measures

We argue that for an agent to successfully self-assess its performance and signal to the demonstrator when it is finished learning, it must be able to (1) correctly identify when the current demonstrations are truly sufficient vs. when they are truly insufficient and (2) do so in an efficient manner so as to minimize human burden, effort, and supervision during the training phase while still maintaining utmost safety and alignment. As such, we focus on the following two dependent measures when evaluating the three stopping condition approaches above:

- **Identification accuracy:** We use *F1 score* to represent identification accuracy, defined as

  $$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{10}$$

  True positive (TP) means that $\text{regret}(\pi_{\text{agent}}, R^*) \leq \epsilon$ when the agent declares demonstration sufficiency; that is, the agent's current learned policy at the time of demonstration sufficiency truly has low regret with respect to the expert's reward function. A false positive (FP) is when the agent declares demonstration sufficiency but $\text{regret}(\pi_{\text{agent}}, R^*) > \epsilon$. A false negative (FN) is when the agent does not declare demonstration sufficiency but $\text{regret}(\pi_{\text{agent}}, R^*)$ is actually already less than ϵ.
- **Sample efficiency:** While having good accuracy is important, practicality in terms of human interaction burden is also crucial. More specifically, we do not want the human to have to give too many demonstrations to the agent before it can learn a high-performing policy. As such, we measure the *proportion of demonstrations needed* before the agent determines it can stop receiving demonstrations. For discrete environments this is the number of unique states where a demonstrator action is provided. For continuous environments we use the number of demonstrated trajectories.



(a) (b) (c) (d)

Figure 3: *nEVD method compared to baseline methods for determining demonstration sufficiency:* The x-axis across all subfigures denote the nEVD bound threshold the agent was using to assess demonstration sufficiency with our method. "Conv." denotes the convergence baseline with a patience hyperparameter of p. "V.S." is the validation set baseline with an interval hyperparameter of i. Subfigures (a) and (b) show the sample efficiency and identification accuracy measures, respectively, for the discrete domain (gridworld and driving). Subfigures (c) and (d) show the sample efficiency and identification accuracy measures, respectively, for the continuous domain (lander and lavaworld). Bands around the lines denote standard error.

Analysis

For our nEVD stopping condition, we tested five different thresholds, or ϵ values: 0.1, 0.2, 0.3, 0.4, 0.5. We stop at 0.5 as, by definition of nEVD, this denotes a regret that is exactly half that of a random policy; any larger regret we deemed unreasonable. For the convergence baseline, we tested five different patience hyperparameters, $p = 1, 2, 3, 4, 5$, denoting the number of iterations for which the agent's policy must remain unchanged. For the validation set baseline, we also tested five different interval hyperparameters, $i = 3, 4, 5, 6, 7$, denoting how often a demonstration is saved into a validation set instead of being used for BIRL. All of these values were determined based on common early stopping parameters and training set split sizes from supervised learning. After running experiments on hyperparameters for both baselines, we selected the best hyperparameter (one that gave the best sample efficiency vs. identification accuracy tradeoff) for each baseline in order to compare with our regret confidence bounding method[1].

We tested the following hypotheses:

H1. Our method achieves higher F1 scores than baseline methods.

[1] Note that both the convergence and validation set baselines do not rely on the nEVD thresholds to assess demonstration sufficiency. We overlay the best baseline performances simply to show how our method compares to them across all nEVD thresholds; the F1 score plots additionally show that we are better at achieving true low regret regardless of nEVD threshold.

H2. Our method requires fewer demonstrations to be given to the agent before it declares demonstration sufficiency, compared to baseline methods.

The results in Figure 3 show that the nEVD bounding method generally outperforms both baseline stopping conditions. In the discrete domains ((a) and (b)), our method achieves a higher F1 score, near 1.00 for all thresholds, than the validation set baseline (V.S.), while requiring at least 25% fewer demonstrations. This can be attributed to the fact that V.S. needs to set aside usable demonstrations for its held-out set, and on top of that requires an exact match between π_{MAP} states and held-out states. The convergence baseline (Conv.) has high sample efficiency, but this comes at the cost of a much lower F1 score, a consistent trend across both the discrete and continuous domains. This can be attributed to the fact that Conv. depends on the stability of π_{MAP}, not its actual performance.

In the continuous domains (Figure 3(c) and (d)), our sample efficiency over the baseline methods becomes much clearer. We believe the difference is so stark because in environments with a continuous state space, there is more ambiguity regarding the demonstrator's true reward R^*. This ambiguity causes R_{MAP} to vary widely, which means that the baselines end up requesting many demonstrations. This results in V.S. achieving a high F1-score due to exact matching with the optimal policy, but it comes with the aforementioned sacrifice in sample efficiency.

Our method maintains high F1 scores *and* high sample efficiency in both domains because it takes into account how well the agent's current policy π_{agent} will perform under the ground-truth reward function compared to an expert using our high-confidence bounds. It does not require that π_{agent} converges to or matches any singular policy so long as the agent is confident that π_{agent} achieves low regret; after all, we argue, after determining confidence in low regret, there is minimal benefit in waiting for more demonstrations to try and get the policy to be an exact function, especially since there can be multiple different policies that all achieve a similar low level of regret under the true reward function. Moreover, a converged policy does not necessarily mean it will generalize well to the expert's true intended reward function. It may just mean that consecutive demonstrations convey very similar information.

One final, major benefit of determining demonstration sufficiency based on high-confidence bounds on nEVD is that it allows human demonstrators to control the agent's performance directly according to desired confidence levels (α, δ) and performance thresholds (ϵ) without relying on proxies such as iterations or exact demonstrations as in the baselines.

To more rigorously compare the methods' identification accuracy, we conducted a hypothesis test for **H1**. Since the distribution of F1 scores across all methods and their corresponding thresholds did not meet normality or variance homogeneity assumptions, we used a Kruskal-Wallis test, which yielded statistically significant results for both the discrete ($H = 71, p \approx 0$) and continuous ($H = 83, p \approx 0$) domains. Subsequent Dunn post-hoc tests with the Bonferroni correction and median comparisons revealed that there was a statistically significant difference between our method's F1 scores and Conv.'s ($p \approx 0$), but no significant difference between our method's scores and V.S.'s. ($p \approx 1$). Thus, our results partially support **H1** (though results for **H2** below show that our method is statistically significantly more sample-efficient than V.S., suggesting it is still better overall).

We ran the same set of statistical tests to compare each method's sample efficiency across thresholds. Kruskal-Wallis yielded statistically significant results for both the discrete domain ($H = 358, p \approx 0$) and continuous domain ($H = 601, p \approx 0$). Dunn and median comparisons revealed that our method required fewer demonstrations for the continuous domain ($p \approx 0$) compared to the convergence baseline and fewer demonstrations for both domains ($p \approx 0$ for both) compared to the validation set baseline. While Conv. required fewer demonstrations for the discrete domain than our method, the median difference was only 12%. Our results provide decisive evidence for **H2**, especially since most environments agents encounter in the real world will have continuous state spaces.

Comparison to Prior Theoretical Bounds

In this section we examine the efficiency at which our method obtains confidence bounds compared to prior work [1, 47] in IRL that uses Hoeffding bounds. While these works were also focused on determining the optimal number of demonstrations to achieve a policy regret bound, their bounds depend on loose concentration inequalities regarding the demonstrator's state occupancy frequencies. As a result, these bounds are highly impractical for determining real-world demonstration sufficiency. To showcase this, we averaged results across a common set of gridworld MDPs with $\alpha = 0.95$ for our method and a 95% confidence level for the other two methods. Table 1 shows how many demonstrations the latter require to reach each of our policy loss thresholds. When compared with Figure 3, the results in Table 1 show that our approach provides a dramatic improvement in practicality over prior high-confidence bounds for agents that learn from demonstrations. The implications of this is that if demonstrations happen to be redundant or ambiguous, agents using our confidence bounding approach can adapt and be able to identify that there still is a high level of uncertainty about the demonstrator's intent and thus request additional demonstrations. Meanwhile, Hoeffding-

Threshold	Abbeel and Ng [1]	Syed and Schapire [47]
0.1	1,624,056	3,654,126
0.2	406,014	913,532
0.3	180,451	406,014
0.4	101,504	228,383
0.5	64,963	146,166

Table 1: Number of demonstrations required under a 95% confidence Hoeffding-based bound to reach each nEVD threshold. Our method (showcased in Figure 3) requires orders of magnitude fewer demonstrations to reach the same bounds.

based methods would incorrectly declare demonstration sufficiency, because they are focused on demonstration quantity rather than quality.

Noisy Demonstrations Ablation

Finally, we studied how our method performs given noisy, or suboptimal, demonstrations. We ran a small experiment that varied the percentage of noisy demonstrations and assessed how identification accuracy and sample efficiency changed as noise increased. On average, we found that identification accuracy decreases slowly with noise, remaining above 95% until more than about 30% of demonstrations are suboptimal. The same trend could be found for the true positive rate, indicating that even with noisy demonstrations, an agent using our nEVD bounds is still able to correctly pinpoint at which point it can safely stop receiving training data. On the other hand, the false positive rate increases faster but still remains below 5% until around 20% of demonstrations are suboptimal. This trend is expected since the agent will be misled towards an incorrect reward and policy given very noisy demonstrations. Meanwhile, we found that there was no clear trend in relation to noise when it came to sample efficiency; across all noise levels, sample efficiency datapoints remained roughly within ±6% of each other. Overall, this experiment provides some evidence that our methods are decently robust to noise—20% to 30% of demonstrations can be suboptimal, which is promising for real-world applications.

6 Methodology Extensions

Percent Improvement over a Baseline Policy

Our framework of using high-confidence bounds to help agents reason under uncertainty regarding their policy performance can be applied to another flavor of demonstration sufficiency, one based on performance gain rather than loss.

There can be many situations where a baseline policy already exists, e.g., a robot comes pre-deployed with a default policy, or the demonstrator has previously trained a safe policy for one task and now wants to teach the agent a related task. In such scenarios, a stopping condition based on bounds on improvement over the baseline policy would allow the agent to learn a policy that performs better under the true reward function, with high confidence. We define this as **Percent Improvement Over a Baseline** (PIOB):

$$PIOB(\pi_{\text{agent}}, \pi_{\text{base}}, R) = \frac{V_R^{\pi_{\text{agent}}} - V_R^{\pi_{\text{base}}}}{V_R^{\pi_{\text{base}}}} \tag{11}$$

Using the same approach as before, we sample reward functions from the Bayesian posterior given demonstrations and use these samples to create a bound on performance gain at a given confidence level. The agent signals demonstration sufficiency when its estimated lower bound on PIOB surpasses the user-provided improvement threshold. Since the agent is

PIOB Bound Threshold	Discrete F1 Score	Continuous F1 Score
20%	1.00 ± 0.00	0.99 ± 0.00
40%	0.97 ± 0.00	0.99 ± 0.00
60%	0.95 ± 0.00	0.99 ± 0.00

Table 2: *PIOB method:* F1 scores for the two domains, for (a subset of) each percent improvement bound threshold used.

trying to obtain a lower bound on policy improvement rather than an upper bound on policy loss, the agent uses a $(1-\alpha)$-worst-case value:

$$\nu_{1-\alpha}(Z) = F_Z^{-1}(1-\alpha) = \sup\{z : F_Z(z) \leq (1-\alpha)\} \tag{12}$$

Given a set of demonstrations D, a baseline policy π_{base}, and an improvement threshold ϵ, demonstration sufficiency is now determined by whether the agent policy sufficiently improves over the baseline with high confidence.

Using the same four environments, we found that sample efficiency was similar to what was achieved with the nEVD bound method (67% ± 2% of states for discrete, 6.12 ± 0.22 states for continuous) and, as expected, decreased with increasing threshold values, especially if the original baseline policy was already high-performing. Table 2 highlights the resulting F1 scores using the PIOB bound method: while they start out high, similar to those achieved with nEVD bounds, they decrease as the threshold increases because the agent accrues more false negatives—due to the conservative nature of our high-confidence performance bounds. We do not see this as a large concern because these bounds are designed to be lower bounds on policy gain. That is, the agent may underestimate the quality of its learned policy, which in reality can turn out to be better than expected.

Active Learning

Our previous empirical experiments used demonstrations that were given in a "passive" manner, where demonstrations were randomly chosen by the (simulated) human. In this section, we investigate the benefits of applying prior work on risk-aware active queries [15] to allow the agent to actively query for demonstrations, which can achieve better sample efficiency while still maintaining high identification accuracy.

For the two discrete environments where there are a finite set of states to select from, we conducted experiments using the same MDPs and dependent measures for both nEVD and PIOB bounding methods, with a key difference being that the agent is able to actively query a specific state where it wants an additional demonstration. At each iteration, the agent calculates which state has the highest α-VaR bound on expected value difference (EVD), then requests a demonstration from this state to be added to the demo set D for the next

	Reduction	
Stopping Condition	Mean	StdDev
nEVD	12.95%	1.62%
Percent improvement	13.24%	0.65%

Table 3: *Passive vs. active demonstration selection:* The mean and standard deviation of reduction in proportion of states needed between active and passive demonstration selection.

iteration. Note that we use unnormalized state EVD here (similar to Eq. (4) but for a single state instead of the whole policy). This is because the normalization factor helps quantify the performance of a policy but is unnecessary computation if we are merely comparing the EVDs of different states with each other. Formally, the agent will select a state, s^*, for requesting a new demonstration according to

$$s^* = \operatorname*{argmax}_{s \in S} \nu_\alpha(EVD(s, R)) = \operatorname*{argmax}_{s \in S} \nu_\alpha(V_R^*(s) - V_R^{\pi_{\text{agent}}}(s)) \tag{13}$$

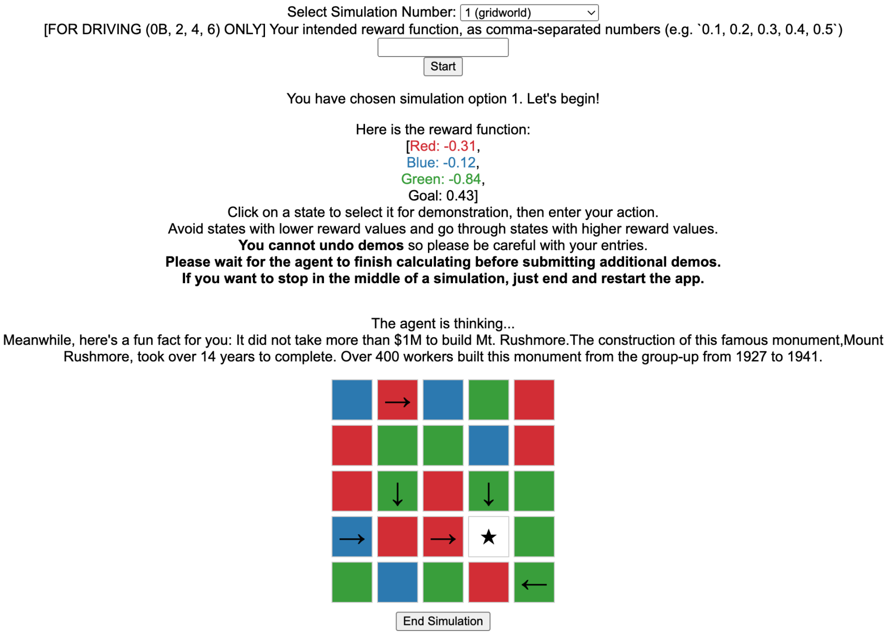
where R are the reward functions sampled from the Bayesian posterior and $V_R^*(s)$ and $V_R^{\pi_{\text{agent}}}(s)$ are the values of state s under R for the respective policies.

Note that both the nEVD bound and PIOB bound stopping conditions use EVD in selecting a state to actively query. While in practice the demonstrator can set the selection metric to be any measure he or she prefers (e.g. percent improvement in any one state's value), we believe that it is best for the agent to select states based on which one currently results in the most policy regret compared to an expert, to ensure as high-performing a policy as possible.

We find that active demonstration selection results in significantly fewer state-action pairs required for the agent to signal demonstration sufficiency, compared to passive demonstration selection, with no compromise in identification accuracy. When actively querying, the agent is able to pinpoint exactly what information it needs before it can be confident in learning a high-performing policy, instead of the demonstrator having to guess what information would be most useful. This reduction in the percentage of states needed is showcased in Table 3.

7 User Study

We designed a user study in order to evaluate our approach with demonstrations provided by real humans, focusing on our first proposed stopping condition of high-confidence bounds on nEVD. We recruited 11 participants from the university campus, aged 18-55, 64% male, 36% female.



Select Simulation Number: 1 (gridworld)

[FOR DRIVING (0B, 2, 4, 6) ONLY] Your intended reward function, as comma-separated numbers (e.g. `0.1, 0.2, 0.3, 0.4, 0.5`)

You have chosen simulation option 1. Let's begin!

Here is the reward function:
[Red: -0.31,
Blue: -0.12,
Green: -0.84,
Goal: 0.43]

Click on a state to select it for demonstration, then enter your action.
Avoid states with lower reward values and go through states with higher reward values.
You cannot undo demos so please be careful with your entries.
Please wait for the agent to finish calculating before submitting additional demos.
If you want to stop in the middle of a simulation, just end and restart the app.

The agent is thinking...
Meanwhile, here's a fun fact for you: It did not take more than $1M to build Mt. Rushmore.The construction of this famous monument,Mount Rushmore, took over 14 years to complete. Over 400 workers built this monument from the group-up from 1927 to 1941.

🟦	🟥 →	🟦	🟩	🟥
🟥	🟩	🟩	🟦	🟥
🟥	🟩 ↓	🟥	🟩 ↓	🟩
🟦 →	🟥	🟥 →	★	🟩
🟩	🟦	🟩	🟥	🟩 ←

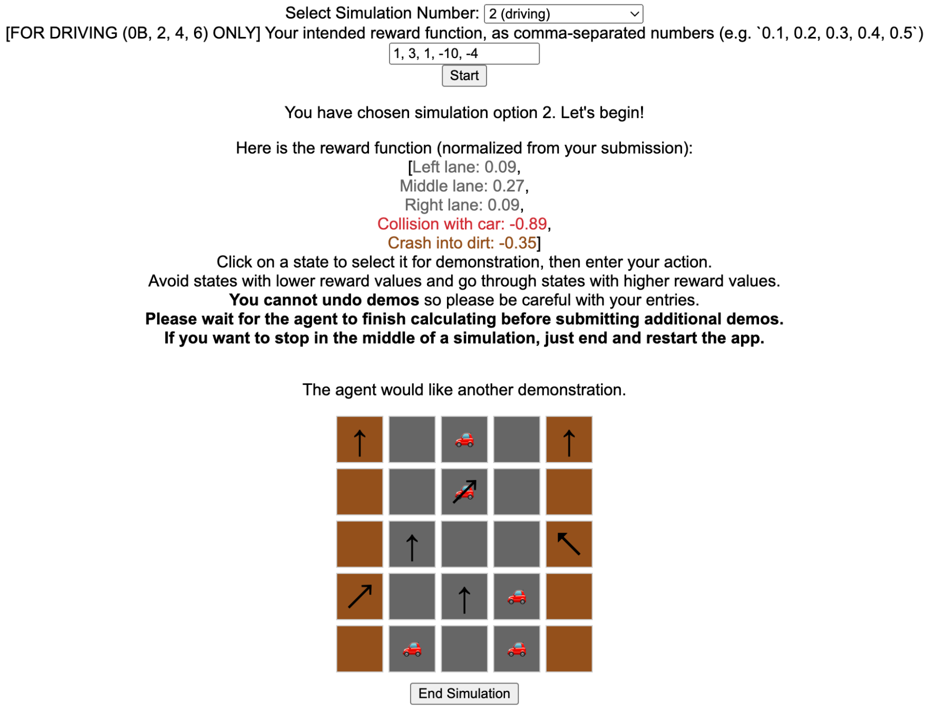
Figure 4: *User study.* The waiting screen as the agent is calculating on a gridworld environment.

Experimental Design

To keep the user study session within a reasonable amount of time, we designed six rounds of experiments for our participants to execute via an online interface: two environments × three demonstration sufficiency methods. The study was blind in that users did not know which round used which method. We designed instances of the gridworld and driving environments which would be both easy enough for users to handle during the study, and, like in the real world, difficult enough where users could provide good, but not always optimal, demonstrations [37]. The three methods used were nEVD bounds (ours), convergence, and validation set. We used $\alpha = 0.95$ and a threshold of 0.3 for our method, $p = 3$ for convergence, and $i = 5$ for validation set (the median hyperparameter values for each method).

For each round, users were presented with either a gridworld or driving environment to teach the agent in. They were instructed to sequentially provide demonstrations, which were (state, action) pairs, via the online interface, until the robot declared demonstration sufficiency.

For gridworld, users were shown a reward function as a weight vector, where each reward value was color-coded to match a feature. They were told to guide the robot towards the goal as fast as possible while avoiding low-reward features (see example in Figure 4). For driving, we described the relevant features (three lanes, collision, and dirt patch) and requested users create their own reward function as a weighted combination of those features. We provided



Select Simulation Number: 2 (driving)

[FOR DRIVING (0B, 2, 4, 6) ONLY] Your intended reward function, as comma-separated numbers (e.g. `0.1, 0.2, 0.3, 0.4, 0.5`)

1, 3, 1, -10, -4

You have chosen simulation option 2. Let's begin!

Here is the reward function (normalized from your submission):
[Left lane: 0.09,
Middle lane: 0.27,
Right lane: 0.09,
Collision with car: -0.89,
Crash into dirt: -0.35]
Click on a state to select it for demonstration, then enter your action.
Avoid states with lower reward values and go through states with higher reward values.
You cannot undo demos so please be careful with your entries.
Please wait for the agent to finish calculating before submitting additional demos.
If you want to stop in the middle of a simulation, just end and restart the app.

The agent would like another demonstration.

Figure 5: *User study.* The agent requests another demonstration in the driving environment.

examples to help, such as, "If you want to drive towards the right as much as possible and avoid accidents, your reward function could be 1, 2, 3, -10, -5." Their reward function was then normalized to have an L2 norm of 1 to be consistent with our methodology. Users were then told to give demonstrations according to this custom reward function (see example in Figure 5).

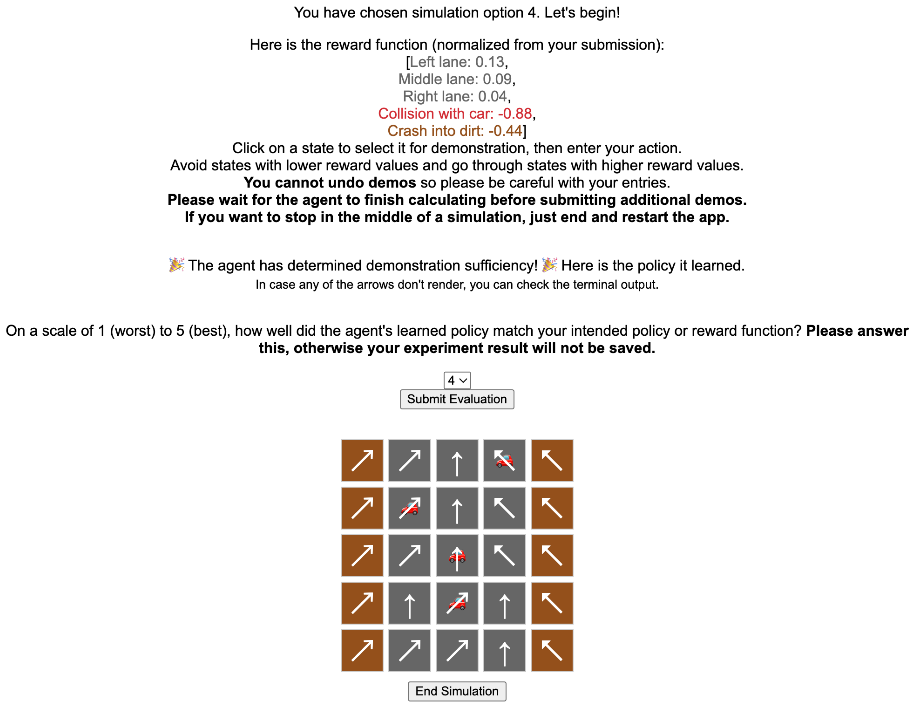
At the end of each of the six rounds, users were shown a visual display of the robot's learned policy and asked, "On a scale of 1 (worst) to 5 (best), how well did the agent's learned policy match your intended policy or reward function?" See Figure 6.

Analysis

We tested the following hypotheses:

H3. Users liked the policies that our method learned more than those that the baseline methods learned.

H4. The proportion of demonstrated states our method required in the user study is less than what the baseline methods required.



You have chosen simulation option 4. Let's begin!

Here is the reward function (normalized from your submission):
[Left lane: 0.13,
Middle lane: 0.09,
Right lane: 0.04,
Collision with car: -0.88,
Crash into dirt: -0.44]
Click on a state to select it for demonstration, then enter your action.
Avoid states with lower reward values and go through states with higher reward values.
You cannot undo demos so please be careful with your entries.
Please wait for the agent to finish calculating before submitting additional demos.
If you want to stop in the middle of a simulation, just end and restart the app.

🎉 The agent has determined demonstration sufficiency! 🎉 Here is the policy it learned.
In case any of the arrows don't render, you can check the terminal output.

On a scale of 1 (worst) to 5 (best), how well did the agent's learned policy match your intended policy or reward function? **Please answer this, otherwise your experiment result will not be saved.**

Figure 6: *User study.* The agent's final learned policy is displayed in white and the user is asked for a ranking.

		Gridworld			Driving	
Metric	Ours	Conv.	V.S.	Ours	Conv.	V.S.
Prop. states	0.22 ± 0.05	0.45 ± 0.03	0.64 ± 0.01	0.25 ± 0.06	0.69 ± 0.07	0.60 ± 0.03
User eval.	4.31 ± 0.40	3.47 ± 0.32	4.00 ± 0.53	4.15 ± 0.28	3.00 ± 0.71	4.00 ± 0.39

Table 4: *User study results:* The three methods' performances in our user study. We still use proportion of states to assess sample efficiency. However, instead of using F1 score for identification accuracy, we use user evaluation: now that we have real humans, this gives a better sense of whether or not the agent's policy is actually aligned with the human's intent.

Results from our user study are shown in Table 4. For each hypothesis we ran Kruskal-Wallis tests and Dunn post-hoc tests with median comparisons for each environment. For **H3**, Kruskal-Wallis did not yield statistically significant results for either gridworld or driving ($0.05 < p < 0.1$). Table 4 does show that our method achieves a higher mean user evaluation than the baselines; thus the lack of statistical significance could be due to a non-standardized evaluation scale (i.e. users may have different internal assessments of what a, say, 3 vs. 4

means). For **H4**, Kruskal-Wallis yielded statistically significant results for both environments ($H = 19, p < 0.0001$ for gridworld; $H = 14, p < 0.001$ for driving). Dunn further revealed that our method required a lower proportion of demonstrated states for gridworld ($p < 0.05$ for both baselines) and for driving ($p < 0.05$ for both baselines).

Our user study revealed two more interesting outcomes. First, we found that our method was much more sample efficient in the user study than in our empirical experiments. Comparing with the empirical results for an nEVD bound threshold of 0.3, our method required over 60% fewer demonstrations to be shown in the user study. We hypothesize that this is because the actual human demonstrators were more likely to choose highly informative demonstrations instead of random ones, enabling faster learning[2]. Second, unlike in our empirical experiments, the user-provided demonstrations were indeed suboptimal at times; on average, 14% of user demos were suboptimal for gridworld, 8% for driving. The noise for driving shows that users aren't perfect at following even their own specified reward function, an interesting area future work can explore. Nevertheless, our approach still was able to efficiently and accurately determine demonstration sufficiency, indicating its robustness to noisy, real-world data.

8 Discussion

Pathway to Deployment

Deploying our demonstration sufficiency methods onto a physical agent or other AI system is a matter of integrating the algorithms into or extending the agent or system's existing software and then having a human available to provide demonstrations. Demonstrations for physical agents are often provided through teleoperation, kinesthetic teaching, or even videos. Our methods assume that the agent shares the same capabilities as the demonstrator, can correctly map demonstrated states and actions into its own state and action spaces, and can perform policy optimization (either model-based or model-free). While these are strong assumptions, they are common in HRI and are not unrealistic given recent advances in feature alignment [9, 10], cross-embodiment IRL [52], and offline reward and policy learning [46]. The nEVD stopping condition can be used when no baseline policy exists or is able to be provided, or when the demonstrator wants to ensure confidence in minimizing policy loss itself. Meanwhile, the percent improvement stopping condition can be used in situations in which a baseline policy can be provided and the demonstrator is focused on improving this existing policy. Selecting the thresholds for the stopping conditions and other parameters will depend on the risk-sensitivity of the environment and user discretion, though $\alpha = 0.95$ is most commonly used. Finetuning these values will enable the demonstrator to adjust the agent's performance and conservativeness to their liking.

[2] Future work should investigate how close these human demonstrations are to optimally pedagogic demonstrations [18, 16].

Limitations and Future Work

One of the limitations in our experiments is the repeated running of MCMC in the BIRL algorithm, which is time- and resource-intensive, especially as the number of samples increases. Implementing successor features could optimize transfer learning between different R_{MAP} reward functions [6], improving MCMC efficiency. Alternatively, merging our Bayesian approach with [19] to estimate the reward function without requiring the inner-loop MDP solver can also be an interesting area of future work. In addition, future work should explore the benefits of active queries in continuous-state domains.

Furthermore, while our empirical experiments and user study provide some evidence that our methodologies are compatible with suboptimal demonstrations, future work could make this application more robust by running a calibration stage before demonstration collection to estimate the suboptimality of the demonstrator [44, 23] and tune β in the Bayesian inference algorithm. Finally, it will be interesting to study whether mutual information or posterior entropy could be used for estimating demonstration sufficiency.

Conclusion

In this work, we formalized the problem of demonstration sufficiency and proposed several methods which an agent can use to determine whether it has received enough demonstration data. Our empirical and user study results provide promising evidence that our methods allow agents to self-assess their performance in cases where the reward function is unobserved by first estimating this reward from human demonstrations then bounding their performance under it. Rather than simply giving agents as many demonstrations as possible and hoping that they will eventually learn the correct policy, our work takes the onus off the demonstrator by enabling intelligent AI systems to detect themselves when they are highly confident that they can use the existing demonstrations to learn a high-performing policy. It is our hope that researchers and practitioners using our methods will be able to partake in safer, more efficient, and more personalized training and deployment of LfD systems.

Bibliography

[1] Pieter Abbeel and Andrew Y Ng. “Apprenticeship learning via inverse reinforcement learning”. In: *Proceedings of the twenty-first international conference on Machine learning*. 2004, p. 1.

[2] Kareem Amin and Satinder Singh. “Towards resolving unidentifiability in inverse reinforcement learning”. In: *arXiv preprint arXiv:1601.06569* (2016).

[3] Brenna D Argall et al. “A survey of robot learning from demonstration”. In: *Robotics and autonomous systems* 57.5 (2009), pp. 469–483.

[4] Saurabh Arora and Prashant Doshi. “A survey of inverse reinforcement learning: Challenges, methods and progress”. In: *Artificial Intelligence* 297 (2021), p. 103500.

[5] Chris L Baker, Rebecca Saxe, and Joshua B Tenenbaum. “Action understanding as inverse planning”. In: *Cognition* 113.3 (2009), pp. 329–349.

[6] André Barreto et al. “Successor Features for Transfer in Reinforcement Learning”. In: *arXiv preprint arXiv:1606.05312v2* (2018).

[7] Erdem Biyik et al. “Learning Reward Functions from Diverse Sources of Human Feedback: Optimally Integrating Demonstrations and Preferences”. In: *CoRR* abs/2006.14091 (2020). arXiv: `2006.14091`. URL: `https://arxiv.org/abs/2006.14091`.

[8] Michael Bloem and Nicholas Bambos. “Infinite time horizon maximum causal entropy inverse reinforcement learning”. In: *53rd IEEE conference on decision and control*. IEEE. 2014, pp. 4911–4916.

[9] Andreea Bobu et al. “Inducing structure in reward learning by learning features”. In: *The International Journal of Robotics Research* 41.5 (2022), pp. 497–518.

[10] Andreea Bobu et al. “SIRL: Similarity-based Implicit Representation Learning”. In: *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction*. 2023, pp. 565–574.

[11] Ralph Allan Bradley and Milton E Terry. “Rank analysis of incomplete block designs: I. The method of paired comparisons”. In: *Biometrika* 39.3/4 (1952), pp. 324–345.

[12] Greg Brockman et al. *OpenAI Gym*. 2016. arXiv: `1606.01540 [cs.LG]`.

[13] Daniel Brown and Scott Niekum. “Efficient probabilistic performance bounds for inverse reinforcement learning”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 32. 1. 2018.

[14] Daniel Brown et al. “Safe imitation learning via fast bayesian reward inference from preferences”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 1165–1177.

[15] Daniel S Brown, Yuchen Cui, and Scott Niekum. “Risk-aware active inverse reinforcement learning”. In: *Conference on Robot Learning*. PMLR. 2018, pp. 362–372.

[16] Daniel S Brown and Scott Niekum. “Machine teaching for inverse reinforcement learning: Algorithms and applications”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01. 2019, pp. 7749–7758.

[17] Neerincx Burghouts Huizing. “Robotic self-assessment of competence”. In: (2022).

[18] Maya Cakmak and Manuel Lopes. “Algorithmic and human teaching of sequential decision tasks”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 26. 1. 2012, pp. 1536–1542.

[19] Alex J. Chan and Mihaela van der Schaar. “Scalable Bayesian Inverse Reinforcement Learning”. In: *CoRR* abs/2102.06483 (2021). arXiv: `2102.06483`. URL: `https://arxiv.org/abs/2102.06483`.

[20] Anca D Dragan, Kenton CT Lee, and Siddhartha S Srinivasa. “Legibility and predictability of robot motion”. In: *2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. IEEE. 2013, pp. 301–308.

[21] Chelsea Finn, Sergey Levine, and Pieter Abbeel. “Guided cost learning: Deep inverse optimal control via policy optimization”. In: *International conference on machine learning*. PMLR. 2016, pp. 49–58.

[22] Letian Fu et al. “Legs: Learning efficient grasp sets for exploratory grasping”. In: *2022 International Conference on Robotics and Automation (ICRA)*. IEEE. 2022, pp. 8259–8265.

[23] Gaurav R Ghosal et al. “The Effect of Modeling Human Rationality Level on Learning Rewards from Multiple Feedback Types”. In: *arXiv preprint arXiv:2208.10687* (2022).

[24] Noah D Goodman, Chris L Baker, and Joshua B Tenenbaum. “Cause and intent: Social reasoning in causal learning”. In: *Proceedings of the 31st annual conference of the cognitive science society*. Cognitive Science Society Amsterdam. 2009, pp. 2759–2764.

[25] Noah D Goodman and Andreas Stuhlmüller. “Knowledge and implicature: Modeling language understanding as social cognition”. In: *Topics in cognitive science* 5.1 (2013), pp. 173–184.

[26] W. K. Hastings. “Monte Carlo Sampling Methods Using Markov Chains and Their Applications”. In: *Biometrika* 57.1 (1970), pp. 97–109. ISSN: 00063444. URL: `http://www.jstor.org/stable/2334940` (visited on 04/29/2024).

[27] Cory J Hayes, Maryam Moosaei, and Laurel D Riek. “Exploring implicit human responses to robot mistakes in a learning from demonstration task”. In: *2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*. IEEE. 2016, pp. 246–252.

[28] Hong Jun Jeon, Smitha Milli, and Anca Dragan. “Reward-rational (implicit) choice: A unifying formalism for reward learning”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 4415–4426.

[29] Ananth Jonnavittula and Dylan P Losey. “I know what you meant: Learning human objectives by (under) estimating their choice set”. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 2747–2753.

[30] Philippe Jorion. “Value at risk”. In: (2000).

[31] Parameswaran Kamalaruban et al. “Interactive teaching algorithms for inverse reinforcement learning”. In: *arXiv preprint arXiv:1905.11867* (2019).

[32] Nathan Koenig, Leila Takayama, and Maja Matarić. “Communication and knowledge sharing in human–robot interaction and learning from demonstration”. In: *Neural Networks* 23.8-9 (2010), pp. 1104–1112.

[33] Pallavi Koppol, Henny Admoni, and Reid Simmons. “Interaction Considerations in Learning from Humans”. In: *IJCAI*. 2021.

[34] Cassidy Laidlaw and Anca Dragan. “The Boltzmann Policy Distribution: Accounting for Systematic Suboptimality in Human Models”. In: *International Conference on Learning Representations*. 2021.

[35] Sergey Levine, Zoran Popovic, and Vladlen Koltun. “Nonlinear inverse reinforcement learning with gaussian processes”. In: *Advances in neural information processing systems* 24 (2011).

[36] R Duncan Luce. *Individual choice behavior: A theoretical analysis*. Courier Corporation, 2012.

[37] Ajay Mandlekar et al. “What Matters in Learning from Offline Human Demonstrations for Robot Manipulation”. In: *Conference on Robot Learning*. PMLR. 2022, pp. 1678–1690.

[38] Nicholas Metropolis et al. “Equation of State Calculations by Fast Computing Machines”. In: *The Journal of Chemical Physics* 21.6 (June 1953), pp. 1087–1092. ISSN: 0021-9606. DOI: `10.1063/1.1699114`. eprint: `https://pubs.aip.org/aip/jcp/article-pdf/21/6/1087/18802390/1087\_1\_online.pdf`. URL: `https://doi.org/10.1063/1.1699114`.

[39] Andrew Y Ng, Stuart Russell, et al. “Algorithms for inverse reinforcement learning.” In: *ICML*. Vol. 1. 2000, p. 2.

[40] Adam Norton et al. “Metrics for Robot Proficiency Self-assessment and Communication of Proficiency in Human-robot Teams”. In: *ACM Transactions on Human-Robot Interaction (THRI)* 11.3 (2022), pp. 1–38.

[41] Zac Pullar-Strecker et al. “Hitting the target: stopping active learning at the cost-based optimum”. In: *Springer*. 2022.

[42] Deepak Ramachandran and Eyal Amir. “Bayesian Inverse Reinforcement Learning.” In: *IJCAI*. Vol. 7. 2007, pp. 2586–2591.

[43] Harish Ravichandar et al. “Recent advances in robot learning from demonstration”. In: *Annual review of control, robotics, and autonomous systems* 3 (2020), pp. 297–330.

[44] Mariah L Schrum et al. “MIND MELD: Personalized Meta-Learning for Robot-Centric Imitation Learning.” In: *HRI*. 2022, pp. 157–165.

[45] Satvik Sharma et al. “Learning Switching Criteria for Sim2Real Transfer of Robotic Fabric Manipulation Policies”. In: *IEEE International Conference on Automation Science and Engineering (CASE)*. 2022.

[46] Daniel Shin, Anca Dragan, and Daniel S Brown. “Benchmarks and Algorithms for Offline Preference-Based Reward Learning”. In: *Transactions on Machine Learning Research* (2022).

[47] Umar Syed and Robert E Schapire. “A game-theoretic approach to apprenticeship learning”. In: *Advances in Neural Information Processing Systems* 20 (2007).

[48] Aviv Tamar, Yonatan Glassner, and Shie Mannor. “Optimizing the CVaR via sampling”. In: *Twenty-Ninth AAAI Conference on Artificial Intelligence*. 2015.

[49] Tu Trinh, Haoyu Chen, and Daniel S. Brown. “Autonomous Assessment of Demonstration Sufficiency via Bayesian Inverse Reinforcement Learning”. In: *Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction*. HRI ’24. Association for Computing Machinery, 2024, pp. 725–733. ISBN: 9798400703225. DOI: 10.1145/3610977.3634984. URL: https://doi.org/10.1145/3610977.3634984.

[50] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. “Maximum entropy deep inverse reinforcement learning”. In: *arXiv preprint arXiv:1507.04888* (2015).

[51] Gaurav Yengera et al. “Curriculum Design for Teaching via Demonstrations: Theory and Applications”. In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 10496–10509.

[52] Kevin Zakka et al. “Xirl: Cross-embodiment inverse reinforcement learning”. In: *Conference on Robot Learning*. PMLR. 2022, pp. 537–546.

[53] Shao Zhifei and Er Meng Joo. “A survey of inverse reinforcement learning techniques”. In: *International Journal of Intelligent Computing and Cybernetics* (2012).

[54] Brian D Ziebart et al. “Maximum entropy inverse reinforcement learning.” In: *Aaai*. Vol. 8. Chicago, IL, USA. 2008, pp. 1433–1438.