Safe Imitation Learning via Fast Bayesian Reward Inference from Preferences

Daniel S. Brown [1] **Russell Coleman** [1 2] **Ravi Srinivasan** [2] **Scott Niekum** [1]

Abstract

Bayesian reward learning from demonstrations enables rigorous safety and uncertainty analysis when performing imitation learning. However, Bayesian reward learning methods are typically computationally intractable for complex control problems. We propose Bayesian Reward Extrapolation (Bayesian REX), a highly efficient Bayesian reward learning algorithm that scales to high-dimensional imitation learning problems by pre-training a low-dimensional feature encoding via self-supervised tasks and then leveraging preferences over demonstrations to perform fast Bayesian inference. Bayesian REX can learn to play Atari games from demonstrations, without access to the game score and can generate 100,000 samples from the posterior over reward functions in only 5 minutes on a personal laptop. Bayesian REX also results in imitation learning performance that is competitive with or better than state-of-the-art methods that only learn point estimates of the reward function. Finally, Bayesian REX enables efficient high-confidence policy evaluation without having access to samples of the reward function. These high-confidence performance bounds can be used to rank the performance and risk of a variety of evaluation policies and provide a way to detect reward hacking behaviors.

1. Introduction

It is important that robots and other autonomous agents can safely learn from and adapt to a variety of human preferences and goals. One common way to learn preferences and goals is via imitation learning, in which an autonomous agent learns how to perform a task by observing demonstrations of the task (Argall et al., 2009). When learning from demonstrations, it is important for an agent to be able to provide high-confidence bounds on its performance with respect to the demonstrator; however, while there exists much work on high-confidence off-policy evaluation in the reinforcement learning (RL) setting, there has been much less work on high-confidence policy evaluation in the imitation learning setting, where the reward samples are unavailable.

Prior work on high-confidence policy evaluation for imitation learning has used Bayesian inverse reinforcement learning (IRL) (Ramachandran & Amir, 2007) to allow an agent to reason about reward uncertainty and policy generalization error (Brown et al., 2018). However, Bayesian IRL is typically intractable for complex problems due to the need to repeatedly solve an MDP in the inner loop, resulting in high computational cost as well as high sample cost if a model is not available. This precludes robust safety and uncertainty analysis for imitation learning in high-dimensional problems or in problems in which a model of the MDP is unavailable. We seek to remedy this problem by proposing and evaluating a method for safe and efficient Bayesian reward learning via preferences over demonstrations. Preferences over trajectories are intuitive for humans to provide (Akrour et al., 2011; Wilson et al., 2012; Sadigh et al., 2017; Christiano et al., 2017; Palan et al., 2019) and enable better-than-demonstrator performance (Brown et al., 2019b;a). To the best of our knowledge, we are the first to show that preferences over demonstrations enable both fast Bayesian reward learning in high-dimensional, visual control tasks as well as efficient high-confidence performance bounds.

We first formalize the problem of high-confidence policy evaluation (Thomas et al., 2015) for imitation learning. We then propose a novel algorithm, Bayesian Reward Extrapolation (Bayesian REX), that uses a pairwise ranking likelihood to significantly increase the efficiency of generating samples from the posterior distribution over reward functions. We demonstrate that Bayesian REX can leverage neural network function approximation to learn useful reward features via self-supervised learning in order to efficiently perform deep Bayesian reward inference from visual demonstrations. Finally, we demonstrate that samples obtained from Bayesian REX can be used to solve the high-confidence policy evaluation problem for imitation learning. We evaluate our method on imitation learning for Atari games and demonstrate that we can efficiently compute high-confidence bounds on pol-

[1]Computer Science Department, The University of Texas at Austin. [2]Applied Research Laboratories, The University of Texas at Austin. Correspondence to: Daniel Brown <dsbrown@cs.utexas.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

icy performance, without access to samples of the reward function. We use these high-confidence performance bounds to rank different evaluation policies according to their risk and expected return under the posterior distribution over the unknown ground-truth reward function. Finally, we provide evidence that bounds on uncertainty and risk provide a useful tool for detecting reward hacking/gaming (Amodei et al., 2016), a common problem in reward inference from demonstrations (Ibarz et al., 2018) as well as reinforcement learning (Ng et al., 1999; Leike et al., 2017).

2. Related work

2.1. Imitation Learning

Imitation learning is the problem of learning a policy from demonstrations and can roughly be divided into techniques that use behavioral cloning and techniques that use inverse reinforcement learning. Behavioral cloning methods (Pomerleau, 1991; Torabi et al., 2018) seek to solve the imitation learning problem via supervised learning, in which the goal is to learn a mapping from states to actions that mimics the demonstrator. While computationally efficient, these methods suffer from compounding errors (Ross et al., 2011). Methods such as DAgger (Ross et al., 2011) and DART (Laskey et al., 2017) avoid this problem by repeatedly collecting additional state-action pairs from an expert.

Inverse reinforcement learning (IRL) methods seek to solve the imitation learning problem by estimating the reward function that the demonstrator is optimizing (Ng & Russell, 2000). Classical approaches repeatedly alternate between a reward estimation step and a full policy optimization step (Abbeel & Ng, 2004; Ziebart et al., 2008; Ramachandran & Amir, 2007). Bayesian IRL (Ramachandran & Amir, 2007) samples from the posterior distribution over reward functions, whereas other methods seek a single reward function that induces the demonstrator's feature expectations (Abbeel & Ng, 2004), often while also seeking to maximize the entropy of the resulting policy (Ziebart et al., 2008).

Most deep learning approaches for IRL use maximum entropy policy optimization and divergence minimization between marginal state-action distributions (Ho & Ermon, 2016; Fu et al., 2017; Ghasemipour et al., 2019) and are related to Generative Adversarial Networks (Finn et al., 2016). These methods scale to complex control problems by iterating between reward learning and policy learning steps. Alternatively, Brown et al. (2019b) use ranked demonstrations to learn a reward function via supervised learning without requiring an MDP solver or any inference time data collection. The learned reward function can then be used to optimize a potentially better-than-demonstrator policy. Brown et al. (2019a) automatically generate preferences over demonstrations via noise injection, allowing better-than-demonstrator performance even in the absence of explicit preference labels. However, despite their successes, deep learning approaches to IRL typically only return a point estimate of the reward function, precluding uncertainty and robustness analysis.

2.2. Safe Imitation Learning

While there has been much interest in imitation learning, less attention has been given to problems related to safety. SafeDAgger (Zhang & Cho, 2017) and EnsembleDAgger (Menda et al., 2019) are extensions of DAgger that give control to the demonstrator in states where the imitation learning policy is predicted to have a large action difference from the demonstrator. Other approaches to safe imitation learning seek to match the tail risk of the expert as well as find a policy that is indistinguishable from the demonstrations (Majumdar et al., 2017; Lacotte et al., 2019).

Brown & Niekum (2018) propose a Bayesian sampling approach to provide explicit high-confidence performance bounds in the imitation learning setting, but require an MDP solver in the inner-loop. Their method uses samples from the posterior distribution $P(R|D)$ to compute sample efficient probabilistic upper bounds on the policy loss of any evaluation policy. Other work considers robust policy optimization over a distribution of reward functions conditioned on demonstrations or a partially specified reward function, but these methods require an MDP solver in the inner loop, limiting their scalability (Hadfield-Menell et al., 2017; Brown et al., 2018; Huang et al., 2018). We extend and generalize the work of Brown & Niekum (2018) by demonstrating, for the first time, that high-confidence performance bounds can be efficiently obtained when performing imitation learning from high-dimensional visual demonstrations without requiring an MDP solver or model during reward inference.

2.3. Value Alignment and Active Preference Learning

Safe imitation learning is closely related to the problem of value alignment, which seeks to design methods that prevent AI systems from acting in ways that violate human values (Hadfield-Menell et al., 2016; Fisac et al., 2020). Research has shown that difficulties arise when an agent seeks to align its value with a human who is not perfectly rational (Milli et al., 2017) and there are fundamental impossibility results regarding value alignment unless the objective is represented as a set of partially ordered preferences (Eckersley, 2018).

Prior work has used active queries to perform Bayesian reward inference on low-dimensional, hand-crafted reward features (Sadigh et al., 2017; Brown et al., 2018; Bıyık et al., 2019). Christiano et al. (2017) and Ibarz et al. (2018) use deep networks to scale active preference learning to high-dimensional tasks, but require large numbers of active queries during policy optimization and do not perform

Bayesian reward inference. Our work complements and extends prior work by: (1) removing the requirement for active queries during reward inference or policy optimization, (2) showing that preferences over demonstrations enable efficient Bayesian reward inference in high-dimensional visual control tasks, and (3) providing an efficient method for computing high-confidence bounds on the performance of any evaluation policy in the imitation learning setting.

2.4. Safe Reinforcement Learning

Research on safe reinforcement learning (RL) usually focuses on safe exploration strategies or optimization objectives other than expected return (Garcıa & Fernández, 2015). Recently, objectives based on measures of risk such as value at risk (VaR) and conditional VaR have been shown to provide tractable and useful risk-sensitive measures of performance for MDPs (Tamar et al., 2015; Chow et al., 2015). Other work focuses on finding robust solutions to MDPs (Ghavamzadeh et al., 2016; Petrik & Russell, 2019), using model-based RL to safely improve upon suboptimal demonstrations (Thananjeyan et al., 2019), and obtaining high-confidence off-policy bounds on the performance of an evaluation policy (Thomas et al., 2015; Hanna et al., 2019). Our work provides an efficient solution to the problem of high-confidence policy evaluation in the imitation learning setting, in which samples of rewards are *not observed* and the demonstrator's policy is unknown.

2.5. Bayesian Neural Networks

Bayesian neural networks typically either perform Markov Chain Monte Carlo (MCMC) sampling (MacKay, 1992), variational inference (Sun et al., 2019; Khan et al., 2018), or use hybrid methods such as particle-based inference (Liu & Wang, 2016) to approximate the posterior distribution over neural network weights. Alternative approaches such as ensembles (Lakshminarayanan et al., 2017) or approximations such as Bayesian dropout (Gal & Ghahramani, 2016; Kendall & Gal, 2017) have also been used to obtain a distribution on the outputs of a neural network in order to provide uncertainty quantification (Maddox et al., 2019). We are not only interested in the uncertainty of the output of the reward function, but also in the uncertainty over the performance of a policy when evaluated under an uncertain reward function. Thus, we face the difficult problem of measuring the uncertainty in the evaluation of a policy, which depends on the stochasticity of the policy and the environment, as well as the uncertainty over the unobserved reward function.

3. Preliminaries

We model the environment as a Markov Decision Process (MDP) consisting of states $\mathcal{S}$, actions $\mathcal{A}$, transition dynamics $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$, reward function $R : \mathcal{S} \to \mathbb{R}$, initial state distribution S_0, and discount factor γ. Our approach extends naturally to rewards defined as $R(s, a)$ or $R(s, a, s')$; however, state-based rewards have some advantages. Fu et al. (2017) prove that a state-only reward function is a necessary and sufficient condition for a reward function that is disentangled from dynamics. Learning a state-based reward also allows the learned reward to be used as a potential function for reward shaping (Ng et al., 1999), if a sparse ground-truth reward function is available.

A policy π is a mapping from states to a probability distribution over actions. We denote the value of a policy π under reward function R as $V^\pi_R = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t R(s_t)|s_0 \sim S_0]$ and denote the value of executing policy π starting at state $s \in \mathcal{S}$ as $V^\pi_R(s) = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t R(s_t)|s_0 = s]$. Given a reward function R, the Q-value of a state-action pair (s, a) is $Q^\pi_R(s, a) = \mathbb{E}_\pi[\sum_{t=0}^\infty \gamma^t R(s_t)|s_0 = s, a_0 = a]$. We also denote $V^*_R = \max_\pi V^\pi_R$ and $Q^*_R(s, a) = \max_\pi Q^\pi_R(s, a)$.

Bayesian inverse reinforcement learning (IRL) (Ramachandran & Amir, 2007) models the environment as an MDP\R in which the reward function is unavailable. Bayesian IRL seeks to infer the latent reward function of a Boltzman-rational demonstrator that executes the following policy

$$\pi^\beta_R(a|s) = \frac{e^{\beta Q^*_R(s,a)}}{\sum_{b\in\mathcal{A}} e^{\beta Q^*_R(s,b)}}, \tag{1}$$

in which R is the true reward function of the demonstrator, and $\beta \in [0, \infty)$ represents the confidence that the demonstrator is acting optimally. Under the assumption of Boltzman rationality, the likelihood of a set of demonstrated state-action pairs, $D = \{(s, a) : (s, a) \sim \pi_D\}$, given a specific reward function hypothesis R, can be written as

$$P(D|R) = \prod_{(s,a)\in D} \pi^\beta_R(a|s) = \prod_{(s,a)\in D} \frac{e^{\beta Q^*_R(s,a)}}{\sum_{b\in\mathcal{A}} e^{\beta Q^*_R(s,b)}}. \tag{2}$$

Bayesian IRL generates samples from the posterior distribution $P(R|D) \sim P(D|R)P(R)$ via Markov Chain Monte Carlo (MCMC) sampling, but this requires solving for $Q^*_{R'}$ to compute the likelihood of each new proposal R'. Thus, Bayesian IRL methods are only used for low-dimensional problems with reward functions that are often linear combinations of a small number of hand-crafted features (Bobu et al., 2018; Bıyık et al., 2019). One of our contributions is an efficient Bayesian reward inference algorithm that leverages preferences over demonstrations in order to significantly improve the efficiency of Bayesian reward inference.

4. High Confidence Policy Evaluation for Imitation Learning

Before detailing our approach, we first formalize the problem of high-confidence policy evaluation for imitation learning. We assume access to an MDP\R, an evaluation policy π_{eval}, a set of demonstrations, $D = \{\tau_1, \ldots, \tau_m\}$, in which τ_i is either a complete or partial trajectory comprised of states or state-action pairs, a confidence level δ, and performance statistic $g : \Pi \times \mathcal{R} \to \mathbb{R}$, in which $\mathcal{R}$ denotes the space of reward functions and Π is the space of all policies.

The *High-Confidence Policy Evaluation problem for Imitation Learning* (HCPE-IL) is to find a high-confidence lower bound $\hat{g} : \Pi \times \mathcal{D} \to \mathbb{R}$ such that

$$\Pr(g(\pi_{\text{eval}}, R^*) \geq \hat{g}(\pi_{\text{eval}}, D)) \geq 1 - \delta, \tag{3}$$

in which R^* denotes the demonstrator's true reward function and $\mathcal{D}$ denotes the space of all possible demonstration sets. HCPE-IL takes as input an evaluation policy π_{eval}, a set of demonstrations D, and a performance statistic, g, which evaluates a policy under a reward function. The goal of HCPE-IL is to return a high-confidence lower bound $\hat{g}$ on the performance statistic $g(\pi_{\text{eval}}, R^*)$.

5. Deep Bayesian Reward Extrapolation

We now describe our main contribution: a method for scaling Bayesian reward inference to high-dimensional visual control tasks as a way to efficiently solve the HCPE-IL problem for complex imitation learning tasks. Our first insight is that the main bottleneck for standard Bayesian IRL (Ramachandran & Amir, 2007) is computing the likelihood function in Equation (2) which requires optimal Q-values. Thus, to make Bayesian reward inference scale to high-dimensional visual domains, it is necessary to either efficiently approximate optimal Q-values or to formulate a new likelihood. Value-based reinforcement learning focuses on efficiently learning optimal Q-values; however, for complex visual control tasks, RL algorithms can take several hours or even days to train (Mnih et al., 2015; Hessel et al., 2018). This makes MCMC, which requires evaluating large numbers of likelihood ratios, infeasible given the current state-of-the-art in value-based RL. Methods such as transfer learning have great potential to reduce the time needed to calculate Q^*_R for a new proposed reward function R; however, transfer learning is not guaranteed to speed up reinforcement learning (Taylor & Stone, 2009). Thus, we choose to focus on reformulating the likelihood function as a way to speed up Bayesian reward inference.

An ideal likelihood function requires little computation and minimal interaction with the environment. To accomplish this, we leverage recent work on learning control policies from preferences (Christiano et al., 2017; Palan et al., 2019; Bıyık et al., 2019). Given ranked demonstrations, Brown et al. (2019b) propose Trajectory-ranked Reward Extrapolation (T-REX): an efficient reward inference algorithm that transforms reward function learning into classification problem via a pairwise ranking loss. T-REX removes the need to repeatedly sample from or partially solve an MDP in the inner loop, allowing it to scale to visual imitation learning domains such as Atari and to extrapolate beyond the performance of the best demonstration. However, T-REX only solves for a point estimate of the reward function. We now discuss how a similar approach based on a pairwise preference likelihood allows for efficient sampling from the posterior distribution over reward functions.

We assume access to a sequence of m trajectories, $D = \{\tau_1, \ldots, \tau_m\}$, along with a set of pairwise preferences over trajectories $\mathcal{P} = \{(i, j) : \tau_i \prec \tau_j\}$. Note that we do not require a total-ordering over trajectories. These preferences may come from a human demonstrator or could be automatically generated by watching a learner improve at a task (Jacq et al., 2019; Brown et al., 2019b) or via noise injection (Brown et al., 2019a). Given trajectory preferences, we can formulate a pair-wise ranking likelihood to compute the likelihood of a set of preferences over demonstrations $\mathcal{P}$, given a parameterized reward function hypothesis R_θ. We use the standard Bradley-Terry model (Bradley & Terry, 1952) to obtain the following pairwise ranking likelihood function, commonly used in learning to rank applications such collaborative filtering (Volkovs & Zemel, 2014):

$$P(D, \mathcal{P} \mid R_\theta) = \prod_{(i,j) \in \mathcal{P}} \frac{e^{\beta R_\theta(\tau_j)}}{e^{\beta R_\theta(\tau_i)} + e^{\beta R_\theta(\tau_j)}}, \tag{4}$$

in which $R_\theta(\tau) = \sum_{s \in \tau} R_\theta(s)$ is the predicted return of trajectory τ under the reward function R_θ, and β is the inverse temperature parameter that models the confidence in the preference labels. We can then perform Bayesian inference via MCMC to obtain samples from $P(R_\theta \mid D, \mathcal{P}) \propto P(D, \mathcal{P} \mid R_\theta) P(R_\theta)$. We call this approach Bayesian Reward Extrapolation or Bayesian REX.

Note that using the likelihood function defined in Equation (4) does not require solving an MDP. In fact, it does not require any rollouts or access to the MDP. All that is required is that we first calculate the return of each trajectory under R_θ and compare the relative predicted returns to the preference labels to determine the likelihood of the demonstrations under the reward hypothesis R_θ. Thus, given preferences over demonstrations, Bayesian REX is significantly more efficient than standard Bayesian IRL. In the following section, we discuss further optimizations that improve the efficiency of Bayesian REX and make it more amenable to our end goal of high-confidence policy evaluation bounds.

5.1. Optimizations

In order to learn rich, complex reward functions, it is desirable to use a deep network to represent the reward function R_θ. While MCMC remains the gold-standard for Bayesian Neural Networks, it is often challenging to scale to deep networks. To make Bayesian REX more efficient and practical,

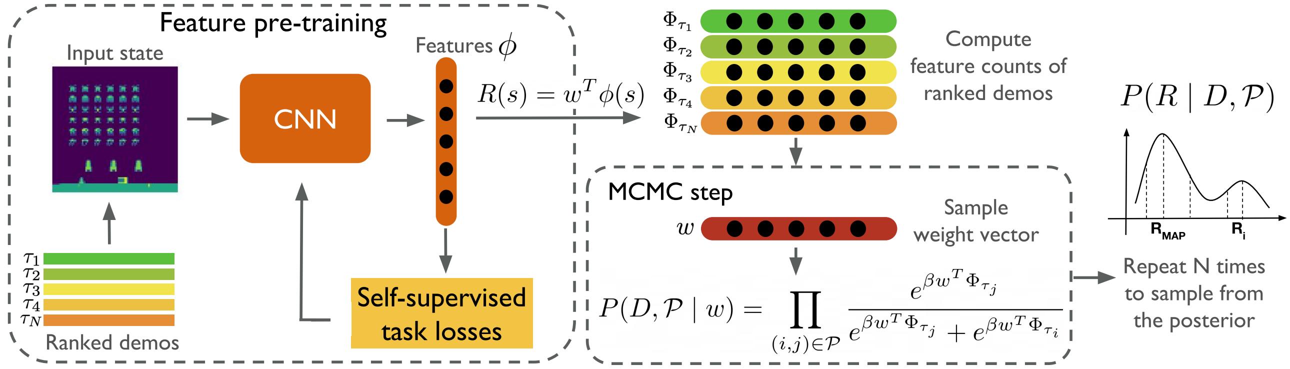

Figure 1. Bayesian Reward Extrapolation uses ranked demonstrations to pre-train a low-dimensional state feature embedding $\phi(s)$ via self-supervised losses. After pre-training, the latent embedding function $\phi(s)$ is frozen and the reward function is represented as a linear combination of the learned features: R(s) = $w^T\phi(s)$. MCMC proposal evaluations use a pairwise ranking likelihood that gives the likelihood of the preferences $\mathcal{P}$ over demonstrations D, given a proposal w. By pre-computing the embeddings of the ranked demonstrations, Φ_{τ_i}, MCMC sampling is highly efficient—it does not require access to an MDP solver or data collection during inference.

we propose to limit the proposal to only change the last layer of weights in R_θ when generating MCMC proposals—we will discuss pre-training the bottom layers of R_θ in the next section. After pre-training, we freeze all but the last layer of weights and use the activations of the penultimate layer as the latent reward features $\phi(s) \in \mathbb{R}^k$. This allows the reward at a state to be represented as a linear combination of k features: $R_\theta(s) = w^T\phi(s)$. Similar to work by Pradier et al. (2018), operating in a lower-dimensional latent space makes full Bayesian inference tractable.

A second advantage of using a learned linear reward function is that it allows us to efficiently compute likelihood ratios when performing MCMC. Consider the likelihood function in Equation (4). If we do not represent R_θ as a linear combination of pretrained features, and instead let any parameter in R_θ change during each proposal, then for m demonstrations of length T, computing $P(D, \mathcal{P} \mid R_\theta)$ for a new proposal R_θ requires $O(mT)$ forward passes through the entire network to compute $R_\theta(\tau_i)$. Thus, the complexity of generating N samples from the posterior results is $O(mTN|R_\theta|)$, where $|R_\theta|$ is the number of computations required for a full forward pass through the entire network R_θ. Given that we would like to use a deep network to parameterize R_θ and generate thousands of samples from the posterior distribution over R_θ, this many computations will significantly slow down MCMC proposal evaluation.

If we represent R_θ as a linear combination of pre-trained features, we can reduce this computational cost because

$$R_\theta(\tau) = \sum_{s\in\tau} w^T\phi(s) = w^T\sum_{s\in\tau}\phi(s) = w^T\Phi_\tau. \quad (5)$$

Thus, we can precompute and cache $\Phi_{\tau_i} = \sum_{s\in\tau_i}\phi(s)$ for $i = 1, \dots, m$ and rewrite the likelihood as

$$P(D, \mathcal{P} \mid R_\theta) = \prod_{(i,j)\in\mathcal{P}} \frac{e^{\beta w^T\Phi_{\tau_j}}}{e^{\beta w^T\Phi_{\tau_j}} + e^{\beta w^T\Phi_{\tau_i}}}. \quad (6)$$

Note that demonstrations only need to be passed through the reward network once to compute Φ_{τ_i} since the pre-trained embedding remains constant during MCMC proposal generation. This results in an initial $O(mT)$ passes through all but the last layer of R_θ to obtain Φ_{τ_i}, for $i = 1, \dots, m$, and then only $O(mk)$ multiplications per proposal evaluation thereafter—each proposal requires that we compute $w^T\Phi_{\tau_i}$ for $i = 1, \dots, m$ and $\Phi_{\tau_i} \in \mathbb{R}^k$. Thus, when using feature pre-training, the total complexity is only $O(mT|R_\theta| + mkN)$ to generate N samples via MCMC. This reduction in the complexity of MCMC from $O(mTN|R_\theta|)$ to $O(mT|R_\theta| + mkN)$ results in significant and practical computational savings because (1) we want to make N and R_θ large and (2) the number of demonstrations, m, and the size of the latent embedding, k, are typically several orders of magnitude smaller than N and $|R_\theta|$.

A third, and critical advantage of using a learned linear reward function is that it makes solving the HCPE-IL problem discussed in Section 4 tractable. Performing a single policy evaluation is a non-trivial task (Sutton et al., 2000) and even in tabular settings has complexity $O(|S|^3)$ in which $|S|$ is the size of the state-space (Littman et al., 1995). Because we are in an imitation learning setting, we would like to be able to efficiently evaluate any given policy across the posterior distribution over reward functions found via Bayesian REX. Given a posterior distribution over N reward function hypotheses we would need to solve N policy evaluations. However, note that given $R(s) = w^T\phi(s)$, the value function of a policy can be written as

$$V_R^\pi = \mathbb{E}_\pi[\sum_{t=0}^{T} R(s_t)] = w^T \mathbb{E}_\pi[\sum_{t=0}^{T} \phi(s_t)] = w^T \Phi_\pi, \quad (7)$$

in which we assume a finite horizon MDP with horizon T and in which Φ_π are the expected feature counts (Abbeel & Ng, 2004; Barreto et al., 2017) of π. Thus, given any evaluation policy π_{eval}, we only need to solve one policy evaluation problem to compute Φ_{eval}. We can then compute the expected value of π_{eval} over the entire posterior distribution of reward functions via a single matrix vector multiplication $W\Phi_{\pi_{\text{eval}}}$, where W is an N-by-k matrix with each row corresponding to a single reward function weight hypothesis w^T. This significantly reduces the complexity of policy evaluation over the reward function posterior distribution from $O(N|S|^3)$ to $O(|S|^3 + Nk)$.

When we refer to Bayesian REX we will refer to the optimized version described in this section (see the Appendix for full implementation details and pseudo-code)[1] . Running MCMC with 66 preference labels to generate 100,000 reward hypothesis for Atari imitation learning tasks takes approximately 5 minutes on a Dell Inspiron 5577 personal laptop with an Intel i7-7700 processor without using the GPU. In comparison, using standard Bayesian IRL to generate *one sample* from the posterior takes 10+ hours of training for a parallelized PPO reinforcement learning agent (Dhariwal et al., 2017) on an NVIDIA TITAN V GPU.

5.2. Pre-training the Reward Function Network

The previous section presupposed access to a pretrained latent embedding function $\phi : S \to \mathbb{R}^k$. We now discuss our pre-training process. Because we are interested in imitation learning problems, we need to be able to train $\phi(s)$ from the demonstrations without access to the ground-truth reward function. One potential method is to train R_θ using the pairwise ranking likelihood function in Equation (4) and then freeze all but the last layer of weights; however, the learned embedding may overfit to the limited number of preferences over demonstrations and fail to capture features relevant to the ground-truth reward function. Thus, we supplement the pairwise ranking objective with auxiliary objectives that can be optimized in a self-supervised fashion using data from the demonstrations.

We use the following self-supervised tasks to pre-train R_θ: (1) Learn an inverse dynamics model that uses embeddings $\phi(s_t)$ and $\phi(s_{t+1})$ to predict the corresponding action a_t (Torabi et al., 2018; Hanna & Stone, 2017), (2) Learn a forward dynamics model that predicts s_{t+1} from $\phi(s_t)$ and a_t (Oh et al., 2015; Thananjeyan et al., 2019), (3) Learn an embedding $\phi(s)$ that predicts the temporal distance between two randomly chosen states from the same demonstration (Aytar et al., 2018), and (4) Train a variational pixel-to-pixel autoencoder in which $\phi(s)$ is the learned latent encoding (Makhzani & Frey, 2017; Doersch, 2016). Table 1 summarizes the self-supervised tasks used to train $\phi(s)$.

Table 1. Self-supervised learning objectives used to pre-train $\phi(s)$.

Inverse Dynamics	$f_{\text{ID}}(\phi(s_t), \phi(s_{t+1})) \to a_t$
Forward Dynamics	$f_{\text{FD}}(\phi(s_t), a_t) \to s_{t+1}$
Temporal Distance	$f_{\text{TD}}(\phi(s_t), \phi(s_{t+x}) \to x$
Variational Autoencoder	$f_A(\phi(s_t)) \to s_t$

There are many possibilities for pre-training $\phi(s)$. We used the objectives described above to encourage the embedding to encode different features. For example, an accurate inverse dynamics model can be learned by only attending to the movement of the agent. Learning forward dynamics supplements this by requiring $\phi(s)$ to encode information about short-term changes to the environment. Learning to predict the temporal distance between states in a trajectory forces $\phi(s)$ to encode long-term progress. Finally, the autoencoder loss acts as a regularizer to the other losses as it seeks to embed all aspects of the state (see the Appendix for details). The Bayesian REX pipeline for sampling from the reward function posterior is shown in Figure 1.

5.3. HCPE-IL via Bayesian REX

We now discuss how to use Bayesian REX to find an efficient solution to the high-confidence policy evaluation for imitation learning (HCPE-IL) problem (see Section 4). Given samples from the distribution $P(w \mid D, \mathcal{P})$, where $R(s) = w^T\phi(s)$, we compute the posterior distribution over any performance statistic $g(\pi_{\text{eval}}, R^*)$ as follows. For each sampled weight vector w produced by Bayesian REX, we compute $g(\pi_{\text{eval}}, w)$. This results in a sample from the posterior distribution $P(g(\pi_{\text{eval}}, R) \mid D, \mathcal{P})$, i.e., the posterior distribution over performance statistic g. We then compute a $(1-\delta)$ confidence lower bound, $\hat{g}(\pi_{\text{eval}}, D)$, by finding the δ-quantile of $g(\pi_{\text{eval}}, w)$ for $w \sim P(w \mid D, \mathcal{P})$.

While there are many potential performance statistics g, we chose to focus on bounding the expected value of the evaluation policy, i.e., $g(\pi_{\text{eval}}, R^*) = V_{R^*}^{\pi_{\text{eval}}} = {w^*}^T\Phi_{\pi_{\text{eval}}}$. To compute a $1-\delta$ confidence bound on $V_{R^*}^{\pi_{\text{eval}}}$, we take advantage of the learned linear reward representation to efficiently calculate the posterior distribution over policy returns given preferences and demonstrations. This distribution over returns is calculated via a matrix vector product, $W\Phi_{\pi_{\text{eval}}}$, in which each row of W is a sample, w, from the MCMC chain and π_{eval} is the evaluation policy. We then sort the resulting vector and select the δ-quantile lowest value. This results in

[1] Project page, code, and demonstration data are available at https://sites.google.com/view/bayesianrex/

a $1-\delta$ confidence lower bound on $V_{R^*}^{\pi_{\rm eval}}$ and corresponds to the δ-Value at Risk (VaR) over $V_R^{\pi_{\rm eval}} \sim P(R \mid D, \mathcal{P})$ (Jorion, 1997; Brown & Niekum, 2018).

6. Experimental Results

6.1. Bayesian IRL vs. Bayesian REX

As noted previously, Bayesian IRL does not scale to high-dimensional tasks due to the requirement of repeatedly solving for an MDP in the inner loop. However, for low-dimensional problems it is still interesting to compare Bayesian IRL with Bayesian REX. We performed a large number of experiments on a variety of randomly generated gridworlds with low-dimensional reward features. We summarize our results here for three different ablations and give full results and implementation details in the appendix.

Ranked Suboptimal vs. Optimal Demos: Given a sufficient number of suboptimal ranked demonstrations (> 5), Bayesian REX performs on par and occasionally better than Bayesian IRL when given the same number of optimal demonstrations.

Only Ranked Suboptimal Demos Bayesian REX always significantly outperforms Bayesian IRL when both algorithms receive suboptimal ranked demonstrations. For fairer comparison, we used a Bayesian IRL algorithm designed to learn from both good and bad demonstrations (Cui & Niekum, 2018). We labeled the top $X\%$ ranked demonstrations as good and bottom $X\%$ ranked as bad. This improved results for Bayesian IRL, but Bayesian REX still performed significantly better across all X.

Only Optimal Demos: Given a sufficient number of optimal demonstrations (> 5), Bayesian IRL significantly outperforms Bayesian REX. To use Bayesian REX with only optimal demonstrations, we followed prior work (Brown et al., 2019a) and auto-generated pairwise preferences using uniform random rollouts that were labeled as less preferred than the demonstrations. In general, this performed much worse than Bayesian IRL, but for small numbers of demonstrations (≤ 5) Bayesian REX leverages self-supervised rankings to perform nearly as well as full Bayesian IRL.

These results demonstrate that if a very small number of unlabeled near-optimal demonstrations are available, then classical Bayesian IRL is the natural choice for performing reward inference. However, if any of these assumptions are not true, then Bayesian REX is a competitive and often superior alternative for performing Bayesian reward inference even in low-dimensional problems where an MDP solver is tractable. If a highly efficient MDP solver is not available, then Bayesian IRL is infeasible and Bayesian REX is the natural choice for Bayesian reward inference.

6.2. Visual Imitation Learning via Bayesian REX

We next tested the imitation learning performance of Bayesian REX for high-dimensional problems where classical Bayesian reward inferernce is infeasible. We pre-trained a 64 dimensional latent state embedding $\phi(s)$ using the self-supervised losses shown in Table 1 and the T-REX pairwise preference loss. We found via ablation studies that combining the T-REX loss with the self-supervised losses resulted in better performance than training only with the T-REX loss or only with the self-supervised losses (see Appendix for details). We then used Bayesian REX to generate 200,000 samples from the posterior $P(R \mid D, \mathcal{P})$. To optimize a control policy, we used Proximal Policy Optimization (PPO) (Schulman et al., 2017) with the MAP and mean reward functions from the posterior (see Appendix for details).

To test whether Bayesian REX scales to complex imitation learning tasks we selected five Atari games from the Arcade Learning Environment (Bellemare et al., 2013). We do not give the RL agent access to the ground-truth reward signal and mask the game scores and number of lives in the demonstrations. Table 2 shows the imitation learning performance of Bayesian REX. We also compare against the results reported by (Brown et al., 2019b) for T-REX, and GAIL (Ho & Ermon, 2016) and use the same 12 suboptimal demonstrations used by Brown et al. (2019b) to train Bayesian REX (see Appendix for details).

Table 2 shows that Bayesian REX is able to utilize preferences over demonstrations to infer an accurate reward function that enables better-than-demonstrator performance. The average ground-truth return for Bayesian REX surpasses the performance of the best demonstration across all 5 games. In comparison, GAIL seeks to match the demonstrator's state-action distributions which makes imitation learning difficult when demonstrations are suboptimal and noisy. In addition to providing uncertainty information, Bayesian REX remains competitive with T-REX (which only finds a maximum likelihood estimate of the reward function) and achieves better performance on 3 out of 5 games.

6.3. High-Confidence Policy Performance Bounds

Next, we ran an experiment to validate whether the posterior distribution generated by Bayesian REX can be used to solve the HCPE-IL problem described in Section 4. We evaluated four different evaluation policies, $A \prec B \prec C \prec D$, created by partially training a PPO agent on the ground-truth reward function and checkpointing the policy at various stages of learning. We ran Bayesian REX to generate 200,000 samples from $P(R \mid D, \mathcal{P})$. To address some of the ill-posedness of IRL, we normalize the weights w such that $\|w\|_2 = 1$. Given a fixed scale for the reward weights, we can compare the relative performance of the different evaluation policies when evaluated over the posterior.

Table 2. Ground-truth average scores when optimizing the mean and MAP rewards found using Bayesian REX. We also compare against the performnace of T-REX (Brown et al., 2019b) and GAIL (Ho & Ermon, 2016). Bayesian REX and T-REX are each given 12 demonstrations with ground-truth pairwise preferences. GAIL cannot learn from preferences so it is given 10 demonstrations comparable to the best demonstration given to the other algorithms. The average performance for each IRL algorithm is the average over 30 rollouts.

	Ranked Demonstrations		Bayesian REX Mean	Bayesian REX MAP	T-REX	GAIL
Game	Best	Avg	Avg (Std)	Avg (Std)	Avg	Avg
Beam Rider	1332	686.0	5,504.7 (2121.2)	**5,870.3** (1905.1)	3,335.7	355.5
Breakout	32	14.5	390.7 (48.8)	**393.1** (63.7)	221.3	0.28
Enduro	84	39.8	487.7 (89.4)	135.0 (24.8)	**586.8**	0.28
Seaquest	600	373.3	734.7 (41.9)	606.0 (37.6)	**747.3**	0.0
Space Invaders	600	332.9	**1,118.8** (483.1)	961.3 (392.3)	1,032.5	370.2

Table 3. Beam Rider policy evaluation bounds compared with ground-truth game scores. Policies A-D correspond to evaluation policies of varying quality obtained by checkpointing an RL agent during training. The No-Op policy seeks to hack the learned reward by always playing the no-op action, resulting in very long trajectories with high mean predicted performance but a very negative 95%-confidence (0.05-VaR) lower bound on expected return.

	Predicted		Ground Truth Avg.	
Policy	Mean	0.05-VaR	Score	Length
A	17.1	7.9	480.6	1372.6
B	22.7	11.9	703.4	1,412.8
C	45.5	24.9	1828.5	2,389.9
D	57.6	31.5	2586.7	2,965.0
No-Op	102.5	-1557.1	0.0	99,994.0

The results for Beam Rider are shown in Table 3. We show results for partially trained RL policies A–D. We found that the ground-truth returns for the checkpoints were highly correlated with the mean and 0.05-VaR (5th percentile policy return) returns under the posterior. However, we also noticed that the trajectory length was also highly correlated with the ground-truth reward. If the reward function learned via IRL gives a small positive reward at every time step, then long polices that do the wrong thing may look good under the posterior. To test this hypothesis we used a No-Op policy that seeks to exploit the learned reward function by not taking any actions. This allows the agent to live until the Atari emulator times out after 99,994 steps.

Table 3 shows that while the No-Op policy has a high expected return over the chain, looking at the 0.05-VaR shows that the No-Op policy has high risk under the distribution, much lower than evaluation policy A. Our results demonstrate that reasoning about probabilistic worst-case performance may be one potential way to detect policies that exhibit so-called reward hacking (Amodei et al., 2016) or that have overfit to certain features in the demonstrations that are correlated with the intent of the demonstrations, but do not lead to desired behavior, a common problem in imitation learning (Ibarz et al., 2018; de Haan et al., 2019).

Table 4. Breakout policy evaluation bounds compared with ground-truth game scores. Top Half: No-Op never releases the ball, resulting in high mean predicted performance but a low 95%-confidence bound (0.05-VaR). The MAP policy has even higher risk but also high expected return. Bottom Half: After rerunning MCMC with a ranked trajectory from both the MAP and No-Op policies, the posterior distribution matches the true preferences.

Risk profiles given initial preferences				
	Predicted		Ground Truth Avg.	
Policy	Mean	0.05-VaR	Score	Length
A	1.5	0.5	1.9	202.7
B	6.3	3.7	15.8	608.4
C	10.6	5.8	27.7	849.3
D	13.9	6.2	41.2	1020.8
MAP	98.2	-370.2	401.0	8780.0
No-Op	41.2	1.0	0.0	7000.0
Risk profiles after rankings w.r.t. MAP and No-Op				
A	0.7	0.3	1.9	202.7
B	8.7	5.5	15.8	608.4
C	18.3	12.1	27.7	849.3
D	26.3	17.1	41.2	1020.8
MAP	606.8	289.1	401.0	8780.0
No-Op	-5.0	-13.5	0.0	7000.0

Table 4 contains policy evaluation results for the game Breakout. The top half of the table shows the mean return and 95%-confidence lower bound on the expected return under the reward function posterior for four evaluation policies as well as the MAP policy found via Bayesian IRL and a No-Op policy that never chooses to release the ball. Both the MAP and No-Op policies have high expected returns under the reward function posterior, but also have high risk (low 0.05-VaR). The MAP policy has much higher risk than the

Table 5. Beam Rider human demonstrations.

	Predicted		Ground Truth	
Policy	Mean	0.05-VaR	Avg.	Length
good	12.4	5.8	1092	1000.0
bad	10.7	4.5	396	1000.0
pessimal	6.6	0.8	0	1000.0
adversarial	8.4	2.4	176	1000.0

Table 6. Enduro evaluation of a variety of human demonstrations.

	Predicted		Ground Truth	
Policy	Mean	0.05-VaR	Avg.	Length
good	246.7	-113.2	177	3325.0
periodic	230.0	-130.4	44	3325.0
neutral	190.8	-160.6	0	3325.0
ram	148.4	-214.3	0	3325.0

No-Op policy, despite good true performance. One likely reason is that, as shown in Table 2, the best demonstrations given to Bayesian REX only achieved a game score of 32. Thus, the MAP policy represents an out of distribution sample and thus has potentially high risk, since Bayesian REX was not trained on policies that hit any of the top layers of bricks. The ranked demonstrations do not give enough evidence to eliminate the possibility that only lower layers of bricks should be hit.

To test whether active learning can help, we incorporated two active queries: a single rollout from the MAP policy and a single rollout from the No-Op policy and ranked them as better and worse, respectively, than the original set of 12 suboptimal demonstrations. As the bottom of Table 4 shows, adding two more ranked demonstrations and re-running Bayesian inference, results in a significant change in the risk profiles of the MAP and No-Op policy—the No-Op policy is now correctly predicted to have high risk and low expected returns and the MAP policy now has a much higher 95%-confidence lower bound on performance.

6.4. Human Demonstrations

To investigate whether Bayesian REX is able to correctly rank human demonstrations, we used Bayesian REX to calculate high-confidence performance bounds for a variety of human demonstrations (see the Appendix for full details and additional results).

We generated four human demonstrations for Beam Rider: (1) *good*, a good demonstration that plays the game well, (2) *bad*, a bad demonstration that seeks to play the game but does a poor job, (3) *pessimal*, a demonstration that does not shoot enemies and seeks enemy bullets, and (4) *adversarial* a demonstration that pretends to play the game by moving and shooting but tries to avoid actually shooting enemies. The resulting high-confidence policy evaluations are shown in Table 5. The high-confidence bounds and average performance over the posterior correctly rank the behaviors. This provides evidence that the learned linear reward correctly rewards actually destroying aliens and avoiding getting shot, rather than just flying around and shooting.

Next we demonstrated four different behaviors when playing Enduro: (1) *good* a demonstration that seeks to play the game well, (2) *periodic* a demonstration that alternates between speeding up and passing cars and then slowing down and being passed, (3) *neutral* a demonstration that stays right next to the last car in the race and doesn't try to pass or get passed, and (4) *ram* a demonstration that tries to ram into as many cars while going fast. Table 6 shows that Bayesian REX is able to accurately predict the performance and risk of each of these demonstrations and gives the highest (lowest 0.05-VaR) risk to the *ram* demonstration and the least risk to the *good* demonstration.

7. Conclusion

Bayesian reasoning is a powerful tool when dealing with uncertainty and risk; however, existing Bayesian reward learning algorithms often require solving an MDP in the inner loop, rendering them intractable for complex problems in which solving an MDP may take several hours or even days. In this paper we propose a novel deep learning algorithm, Bayesian Reward Extrapolation (Bayesian REX), that leverages preference labels over demonstrations to make Bayesian reward inference tractable for high-dimensional visual imitation learning tasks. Bayesian REX can sample tens of thousands of reward functions from the posterior in a matter of minutes using a consumer laptop. We tested our approach on five Atari imitation learning tasks and showed that Bayesian REX achieves state-of-the-art performance in 3 out of 5 games. Furthermore, Bayesian REX enables efficient high-confidence performance bounds for arbitrary evaluation policies. We demonstrated that these high-confidence bounds allow an agent to accurately rank different evaluation policies and provide a potential way to detect reward hacking and value misalignment.

We note that our proposed safety bounds are only safe with respect to the assumptions that we make: good feature pre-training, rapid MCMC mixing, and accurate preferences over demonstrations. Future work includes using exploratory trajectories for better pre-training of the latent feature embeddings, developing methods to determine when a relevant feature is missing from the learned latent space, and using high-confidence performance bounds to perform safe policy optimization in the imitation learning setting.

Acknowledgements

This work has taken place in the Personal Autonomous Robotics Lab (PeARL) at The University of Texas at Austin. PeARL research is supported in part by the NSF (IIS-1724157, IIS-1638107,IIS-1617639, IIS-1749204) and ONR(N00014-18-2243, N00014-17-1-2143). This research was also sponsored by the Army Research Office and was accomplished under Cooperative Agreement Number W911NF-19-2-0333. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Abbeel, P. and Ng, A. Y. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the 21st International Conference on Machine Learning*, 2004.

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

Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., and Mané, D. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Argall, B. D., Chernova, S., Veloso, M., and Browning, B. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

Aytar, Y., Pfaff, T., Budden, D., Paine, T. L., Wang, Z., and de Freitas, N. Playing hard exploration games by watching youtube. *arXiv preprint arXiv:1805.11592*, 2018.

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., van Hasselt, H. P., and Silver, D. Successor features for transfer in reinforcement learning. In *Advances in neural information processing systems*, pp. 4055–4065, 2017.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Bıyık, E., Palan, M., Landolfi, N. C., Losey, D. P., and Sadigh, D. Asking easy questions: A user-friendly approach to active reward learning. In *Conference on Robot Learning (CoRL)*, 2019.

Bobu, A., Bajcsy, A., Fisac, J. F., and Dragan, A. D. Learning under misspecified objective spaces. *arXiv preprint arXiv:1810.05157*, 2018.

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

Brown, D. S. and Niekum, S. Efficient Probabilistic Performance Bounds for Inverse Reinforcement Learning. In *AAAI Conference on Artificial Intelligence*, 2018.

Brown, D. S., Cui, Y., and Niekum, S. Risk-aware active inverse reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2018.

Brown, D. S., Goo, W., and Niekum, S. Better-than-demonstrator imitation learning via automaticaly-ranked demonstrations. In *Conference on Robot Learning (CoRL)*, 2019a.

Brown, D. S., Goo, W., Prabhat, N., and Niekum, S. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*, 2019b.

Chow, Y., Tamar, A., Mannor, S., and Pavone, M. Risk-sensitive and robust decision-making: a cvar optimization approach. In *Advances in Neural Information Processing Systems*, 2015.

Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., and Amodei, D. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, pp. 4299–4307, 2017.

Cui, Y. and Niekum, S. Active reward learning from critiques. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

de Haan, P., Jayaraman, D., and Levine, S. Causal confusion in imitation learning. In *Advances in Neural Information Processing Systems*, pp. 11693–11704, 2019.

Dhariwal, P., Hesse, C., Klimov, O., Nichol, A., Plappert, M., Radford, A., Schulman, J., Sidor, S., Wu, Y., and Zhokhov, P. Openai baselines. https://github.com/openai/baselines, 2017.

Doersch, C. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

Eckersley, P. Impossibility and uncertainty theorems in ai value alignment (or why your agi should not have a utility function). *arXiv preprint arXiv:1901.00064*, 2018.

Finn, C., Christiano, P., Abbeel, P., and Levine, S. A connection between generative adversarial networks, inverse reinforcement learning, and energy-based models. *arXiv preprint arXiv:1611.03852*, 2016.

Fisac, J. F., Gates, M. A., Hamrick, J. B., Liu, C., Hadfield-Menell, D., Palaniappan, M., Malik, D., Sastry, S. S., Griffiths, T. L., and Dragan, A. D. Pragmatic-pedagogic value alignment. In *Robotics Research*, pp. 49–57. Springer, 2020.

Fu, J., Luo, K., and Levine, S. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059, 2016.

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

Ghasemipour, S. K. S., Zemel, R., and Gu, S. A divergence minimization perspective on imitation learning methods. *arXiv preprint arXiv:1911.02256*, 2019.

Ghavamzadeh, M., Petrik, M., and Chow, Y. Safe policy improvement by minimizing robust baseline regret. In *Advances in Neural Information Processing Systems*, pp. 2298–2306, 2016.

Hadfield-Menell, D., Russell, S. J., Abbeel, P., and Dragan, A. Cooperative inverse reinforcement learning. In *Advances in neural information processing systems*, pp. 3909–3917, 2016.

Hadfield-Menell, D., Milli, S., Abbeel, P., Russell, S. J., and Dragan, A. Inverse reward design. In *Advances in neural information processing systems*, pp. 6765–6774, 2017.

Hanna, J., Niekum, S., and Stone, P. Importance sampling policy evaluation with an estimated behavior policy. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, June 2019.

Hanna, J. P. and Stone, P. Grounded action transformation for robot learning in simulation. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Hessel, M., Modayil, J., Van Hasselt, H., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

Ho, J. and Ermon, S. Generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*, pp. 4565–4573, 2016.

Huang, J., Wu, F., Precup, D., and Cai, Y. Learning safe policies with expert guidance. In *Advances in Neural Information Processing Systems*, pp. 9105–9114, 2018.

Ibarz, B., Leike, J., Pohlen, T., Irving, G., Legg, S., and Amodei, D. Reward learning from human preferences and demonstrations in atari. In *Advances in Neural Information Processing Systems*, 2018.

Jacq, A., Geist, M., Paiva, A., and Pietquin, O. Learning from a learner. In *International Conference on Machine Learning*, pp. 2990–2999, 2019.

Jorion, P. *Value at risk*. McGraw-Hill, New York, 1997.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pp. 5574–5584, 2017.

Khan, M. E., Nielsen, D., Tangkaratt, V., Lin, W., Gal, Y., and Srivastava, A. Fast and scalable bayesian deep learning by weight-perturbation in adam. *arXiv preprint arXiv:1806.04854*, 2018.

Lacotte, J., Ghavamzadeh, M., Chow, Y., and Pavone, M. Risk-sensitive generative adversarial imitation learning. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 2154–2163, 2019.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in neural information processing systems*, pp. 6402–6413, 2017.

Laskey, M., Lee, J., Fox, R., Dragan, A., and Goldberg, K. Dart: Noise injection for robust imitation learning. *Conference on Robot Learning (CoRL)*, 2017.

Leike, J., Martic, M., Krakovna, V., Ortega, P. A., Everitt, T., Lefrancq, A., Orseau, L., and Legg, S. Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*, 2017.

Littman, M. L., Dean, T. L., and Kaelbling, L. P. On the complexity of solving markov decision problems. *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*, 1995.

Liu, Q. and Wang, D. Stein variational gradient descent: A general purpose bayesian inference algorithm. In *Advances in neural information processing systems*, pp. 2378–2386, 2016.

MacKay, D. J. A practical bayesian framework for backpropagation networks. *Neural computation*, 4(3):448–472, 1992.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pp. 13132–13143, 2019.

Majumdar, A., Singh, S., Mandlekar, A., and Pavone, M. Risk-sensitive inverse reinforcement learning via coherent risk models. In *Robotics: Science and Systems*, 2017.

Makhzani, A. and Frey, B. J. Pixelgan autoencoders. In *Advances in Neural Information Processing Systems*, pp. 1975–1985, 2017.

Menda, K., Driggs-Campbell, K., and Kochenderfer, M. J. Ensembledagger: A bayesian approach to safe imitation learning. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5041–5048. IEEE, 2019.

Milli, S., Hadfield-Menell, D., Dragan, A., and Russell, S. Should robots be obedient? In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pp. 4754–4760, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529, 2015.

Ng, A. Y. and Russell, S. J. Algorithms for inverse reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, pp. 663–670, 2000.

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

Oh, J., Guo, X., Lee, H., Lewis, R. L., and Singh, S. Action-conditional video prediction using deep networks in atari games. In *Advances in neural information processing systems*, pp. 2863–2871, 2015.

Palan, M., Landolfi, N. C., Shevchuk, G., and Sadigh, D. Learning reward functions by integrating human demonstrations and preferences. In *Proceedings of Robotics: Science and Systems (RSS)*, June 2019.

Petrik, M. and Russell, R. H. Beyond confidence regions: Tight bayesian ambiguity sets for robust mdps. *arXiv preprint arXiv:1902.07605*, 2019.

Pomerleau, D. A. Efficient training of artificial neural networks for autonomous navigation. *Neural Computation*, 3(1):88–97, 1991.

Pradier, M. F., Pan, W., Yao, J., Ghosh, S., and Doshi-Velez, F. Projected bnns: Avoiding weight-space pathologies by learning latent representations of neural network weights. *arXiv preprint arXiv:1811.07006*, 2018.

Ramachandran, D. and Amir, E. Bayesian inverse reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artifical intelligence*, pp. 2586–2591, 2007.

Ross, S., Gordon, G., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635, 2011.

Sadigh, D., Dragan, A. D., Sastry, S. S., and Seshia, S. A. Active preference-based learning of reward functions. In *Proceedings of Robotics: Science and Systems (RSS)*, 2017.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Sun, S., Zhang, G., Shi, J., and Grosse, R. Functional variational bayesian neural networks. *arXiv preprint arXiv:1903.05779*, 2019.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Tamar, A., Glassner, Y., and Mannor, S. Optimizing the cvar via sampling. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pp. 2993–2999, 2015.

Taylor, M. E. and Stone, P. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(Jul):1633–1685, 2009.

Thananjeyan, B., Balakrishna, A., Rosolia, U., Li, F., McAllister, R., Gonzalez, J. E., Levine, S., Borrelli, F., and Goldberg, K. Safety augmented value estimation from demonstrations (saved): Safe deep model-based rl for sparse cost robotic tasks. *arXiv preprint arXiv:1905.13402*, 2019.

Thomas, P. S., Theocharous, G., and Ghavamzadeh, M. High-confidence off-policy evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 3000–3006, 2015.

Torabi, F., Warnell, G., and Stone, P. Behavioral cloning from observation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI)*, July 2018.

Volkovs, M. N. and Zemel, R. S. New learning methods for supervised and unsupervised preference aggregation. *The Journal of Machine Learning Research*, 15(1):1135–1176, 2014.

Wilson, A., Fern, A., and Tadepalli, P. A bayesian approach for policy learning from trajectory preference queries. In *Advances in neural information processing systems*, pp. 1133–1141, 2012.

Zhang, J. and Cho, K. Query-efficient imitation learning for end-to-end simulated driving. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. Maximum entropy inverse reinforcement learning. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pp. 1433–1438, 2008.