KERNEL DENSITY BAYESIAN INVERSE REINFORCEMENT LEARNING

A PREPRINT

Aishwarya Mandyam
Gladstone Institutes
Department of Computer Science
Princeton University
aishwarya@princeton.edu

Didong Li
Department of Computer Science
Princeton University
didongli@princeton.edu

Diana Cai
Department of Computer Science
Princeton University
dcai@cs.princeton.edu

Andrew Jones
Department of Computer Science
Princeton University
aj13@princeton.edu

Barbara E. Engelhardt
Stanford University
Gladstone Institutes
barbarae@stanford.edu

September 5, 2022

ABSTRACT

Inverse reinforcement learning (IRL) is a powerful framework for learning the reward function of an agent by observing its behavior, but IRL algorithms that infer point estimates of the reward function can be misleading. A Bayesian approach to IRL models a distribution over possible reward functions that explain the set of observations, alleviating the shortcomings of learning a single point estimate. Existing Bayesian approaches for IRL use a Q-value function, estimated using Q-learning, in place of the likelihood function. However, the resulting posterior is computationally intensive to calculate, has few theoretical guarantees, and the likelihood calculation used is often a poor approximation for the true likelihood function. We introduce kernel density Bayesian IRL (KD-BIRL), a method that uses conditional kernel density estimation to directly approximate the likelihood function in a Bayesian posterior. We prove that the resulting posterior distribution contracts to the correct reward function as the sample size increases, leading to a flexible and efficient framework that is applicable to environments with complex state spaces. We demonstrate KD-BIRL's computational benefits and ability to represent uncertainty in the recovered reward function through a series of experiments in a Gridworld environment and on a healthcare task.

1 Introduction

Reinforcement learning (RL) methods find policies that maximize an agent's long-term expected reward within a Markov decision process (MDP). In many observational data settings, we observe a sequence of states and actions for an agent that is carrying out a policy driven by an unknown reward function. In these cases, it is of interest to infer this reward function in order to uncover the factors driving the agent's behavior. For example, in a hospital setting, we observe the treatment schedule for a patient along with measurements of the patient's physiological state. In order to understand the underlying factors influencing treatment decisions, we may be interested in identifying the doctor's reward function (i.e., objectives)—which is typically complex and mostly unobserved—and how this function drives treatment decisions given a patient's state. It is particularly difficult to infer the reward function in clinical settings because the vector of observed covariates for a patient at a given time is often noisy and partially missing, and there may be several candidate reward functions that can explain the doctor's behavior.

Inverse reinforcement learning (IRL) methods provide a framework to recover an agent's reward function given observations of the agent's behavior. Early IRL algorithms sought a point estimate (i.e., a single reward function) that best explained the observed behavior [2, 30]. These IRL frameworks led to applications in path planning [29], urban navigation [50], and robotics settings [23, 34]. A point estimate of the reward function can also aid in imitation learning, where the inferred reward function is used to fit RL policies that better replicate desired behavior.

Despite the success of early IRL approaches, there are limitations to inferring a point estimate of the reward function. First, the IRL problem is often non-identifiable [2, 35, 50, 51]. There may be multiple (and possibly infinite) reward functions that explain a set of behaviors equally well. For this reason, point estimates are incapable of describing the full set of viable reward functions.

Second, for finite training data, point estimates fail to capture the uncertainty and noise in the data generating process. Thus, it is advantageous to take a Bayesian approach, which treats the reward function as inherently random and communicates a degree of uncertainty that relies on the dataset distribution. At a high level, a Bayesian approach to IRL computes a posterior distribution over possible reward functions that places mass on reward functions proportional to how well those reward functions explain the observed behavior [5, 12, 14, 26, 27, 33].

However, existing Bayesian IRL methods are computationally demanding. In Bayesian modeling, the specification of the likelihood has a large impact on the resulting posterior distribution. The formulation of the likelihood function in the IRL setting (i.e., the function describing the probability of observing a given state-action pair given a reward function) is unknown. Existing approaches replace it with an optimal Q-value function, denoted by $Q^\star$, that best approximates the long-term expected reward for a given state-action pair ([33]). The Q-value function must be learned using Q-learning [42], a "forward RL" algorithm, or one that solves an environment's MDP. The original algorithm pioneered by Ramachandran and Amir [33] and the majority of its successors use Markov chain Monte Carlo (MCMC) sampling to compute a posterior over the reward function, and every iteration of MCMC requires forward RL for each sampled reward function. This step is computationally intensive, especially with infinite or high-dimensional state spaces. Additionally, a posterior that uses $Q^\star$ as a likelihood is equivalent to a Gibbs posterior [6, 48] and lacks several desirable theoretical properties [6].

We address these challenges by introducing kernel density Bayesian inverse reinforcement learning (KD-BIRL), a method that (1) estimates the likelihood function directly, leading to theoretical guarantees for the consistency of the resulting posterior distribution, and (2) disassociates the number of times forward RL is required with the number of iterations of MCMC sampling, which greatly reduces the computational complexity. The contributions of our work are the following.

1. We propose KD-BIRL, a Bayesian IRL method that uses a conditional kernel density estimator to approximate the likelihood without relying on computationally expensive RL procedures (Section 3).
2. We justify our method theoretically by proving posterior consistency, showing that the posterior contracts to the correct reward function as the sample size increases (Section 4).
3. Using a Gridworld environment and a real-world healthcare scenario, we demonstrate that KD-BIRL's posterior estimates more closely align with ground-truth reward functions than those of previously proposed methods (Section 5).

2 Preliminaries

2.1 Background: Inverse reinforcement learning (IRL)

The goal of IRL methods is to infer the reward function of an agent given its behavior. A reinforcement learning agent interacts with and responds to an environment that can be defined using a Markov Decision Process (MDP). An MDP is represented by a 4-tuple $(\mathcal{S}, \mathcal{A}, P, R)$, where $\mathcal{S}$ is the state space; $\mathcal{A}$ is the set of actions; $P(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ defines state-transition probabilities from timestep t to $t+1$; and $R : \mathcal{S} \to \mathbb{R}$ is a reward function, where $R \in \mathcal{R}$ and $\mathcal{R}$ denotes the space of reward functions. The input to an IRL algorithm is a set of expert demonstrations, $\{(s_t, a_t)\}_{t=1}^n$, where each demonstration is a 2-tuple (s_t, a_t) representing an agent's state and chosen action at timestep t. These demonstrations are assumed to arise from an agent acting according to policy, $\pi^\star : \mathcal{S} \to \mathcal{A}$, that is optimal for a fixed but unknown reward function R. Given these demonstrations, IRL algorithms seek a reward function $R^\star$ such that $\pi^\star$ is optimal with respect to $R^\star$.

We now formalize the IRL problem. Let $s \in \mathcal{S} \subseteq \mathbb{R}^p$ denote a state vector of length p. The value function for a given policy π is $V^\pi(s) = \mathbb{E}_{p_0}[R|s, \pi]$ where R is the reward associated with starting in state s and following the policy π, and p_0 is the probability distribution across starting states. Without loss of generality, assume that the optimal policy is given by $\pi^\star(s) = a^\star$, where $a^\star$ is the optimal action to be taken at state s for a given reward function. To identify a reward function R for which π is optimal, the long-term expected value of the optimal policy under R must be at least the long-term expected value of selecting an action that deviates from the optimal policy. That is, when R is correct, it must hold that for all states $s \in \mathcal{S}$ and actions $a \in \mathcal{A} \setminus a^\star$,

$$\mathop{\mathbb{E}}_{s' \sim P_{sa^\star}} [V^\pi(s')] \geq \mathop{\mathbb{E}}_{s' \sim P_{sa}} [V^\pi(s')],$$

where $P_{sa} = p(s'|s,a)$ is the probability of transitioning from state s to state s' after taking action a.

Bayesian approaches to IRL treat the reward function R as inherently random. By specifying a prior distribution for R and a likelihood function for the observed data, these methods can then infer a posterior distribution over the reward function R given n expert demonstrations of an agent acting according to an optimal policy $\{(s_i^e, a_i^e)\}_{i=1}^n$. Here, the posterior density is proportional to the product of the prior distribution on the reward function, $p(R)$, and the likelihood of the expert demonstrations given the reward function:

$$p\left(R \mid \{(s_i^e, a_i^e)\}_{i=1}^n\right) \propto p(R) \prod_{i=1}^n p(s_i^e, a_i^e | R). \tag{1}$$

In the initial formulation of Bayesian IRL [33], the authors propose using a Q-value function in the likelihood calculation in Equation (1). The Q-value function for a given policy π at timestep t is $Q^\pi(s_t, a_t) = r_t + \gamma \mathbb{E}_{s' \sim P}[V^\pi(s')]$, where s_t, a_t, and r_t are the state, action, and reward at time t, and $\gamma \in [0, 1]$ is a discount factor. This approach uses an optimal Q-value function, $Q^\star$, in place of the likelihood. Thus, the "likelihood" takes the form

$$p(s, a \mid R) \propto e^{\alpha Q^\star(s,a,R)}, \tag{2}$$

where $Q^\star(s, a, R)$ is the optimal Q-value function for reward function R, and $\alpha > 0$ is an inverse temperature parameter that represents confidence in the agent's ability to select optimal actions.

There are several potential challenges with learning the aforementioned BIRL posterior. First, the optimal $Q^\star$ is found using Q-learning, a forward RL algorithm, and typically, $Q^\star$ is expensive to estimate for a new R. Using approximate inference algorithms, such as MCMC or variational Bayes, requires evaluating $Q^\star$ on every iteration of sampling or optimization. In addition, because Equation 2 is a loss-based function (rather than a true likelihood), the resulting function is not a classical Bayesian posterior [40] and doesn't have theoretical guarantees regarding posterior contraction. While this can be sufficient for imitation learning, it does not allow for posterior predictive sampling, and updates to the Gibbs posterior are not rational unless the Q-value function satisfies certain conditions [6] (see Appendix A.3). Here, rational updates imply that new evidence is appropriately incorporated to modify the posterior from the prior. The Q-value function can be approximated using any real-value prediction method from neural networks to linear regression, and the rationality of the resulting posterior updates is not discussed in prior work. More specifically, there is no way to confirm that $Q^\star$ satisfies Assumption 3 of the guidelines for updating belief distributions in Bissiri et al. [6]. Furthermore, a Q-value function is a poor approximation of density, or likelihood. In previous formulations, a higher Q-value associated with a state would imply larger density in that state, meaning that an agent prefers traveling to that state. This may not be true depending on the optimality of such an agent. Additionally, Q-value estimates can be incorrect for states that are very rarely visited, as often happens in infinite state spaces.

2.2 Previous Work

Several extensions to the original BIRL algorithm [33] have been proposed. The first set of methods identify nonlinear and nonparametric reward functions [13, 14, 24, 31, 45]. These algorithms use a variety of strategies to model the reward such as Gaussian processes [24, 31], Indian Buffet Process (IBP) priors [13], and Dirichlet process mixture models [14] to learn reward functions for MDPs with large state spaces that may include sub-goals. Other methods reduce computational complexity by using either more informative priors [38], different sampling procedures (e.g., Metropolis-Hastings [14] or expectation-maximization [49]), variational inference to approximate the posterior [12], or by learning several reward functions that each describe a subset of the state space [26, 27]. However, all of these approaches use a Q-value function in place of the likelihood, and hence still suffer from consistency and computational issues; this construction is both inefficient and limits desirable Bayesian behaviors with regards to posterior sampling and uncertainty quantification.

To address these computational and consistency issues, it is necessary to either directly estimate the likelihood, reduce the number of times forward RL is performed, or modify the reward function parameterization. Recent work proposes a variational Bayes framework, Approximate Variational Reward Imitation Learning (AVRIL) [12], to approximate the full posterior. This method improves upon existing work by avoiding re-estimating $Q^\star$ for every sampled reward function, allowing it to bypass some of the computational inefficiencies of Ramachandran and Amir [33]'s initial formulation. However, AVRIL still requires the use of a Q-value function, resulting in a misspecified optimization objective and a posterior estimate that can still be computationally expensive to calculate. The only existing method to avoid using a

Q-value function entirely for the likelihood [28] instead approximates it either using real-time dynamic programming or action comparison. Other work proposes a method that enables imitation learning in complex control problems [9]; this approach builds on the feature-based reward function technique [1, 17] and identifies a low-dimensional feature encoding that can be used to parameterize the reward function. All of these techniques are best suited for environments with a closed-loop controller that provides instant feedback, as in an online environment.

3 Methods

3.1 Conditional kernel density estimation

As discussed above, directly estimating the likelihood function can lead to theoretical guarantees of consistency on the resulting posterior distribution. To estimate the likelihood function $p(s, a|R)$, we first observe that it can be viewed as the conditional density of the state-action pair given the reward function. Thus, any appropriate conditional density estimator could be applied; examples include the conditional kernel density estimator (CKDE, [44]) and Gaussian processes (GPs). In this paper, we adopt the CKDE because it is nonparametric, has a closed form, and is straightforward to implement [19, 20]. Motivated by the equation $p(y|x) = \frac{p(x,y)}{p(x)}$ (where x and y are two generic random variables), the CKDE estimates the conditional density $p(y|x)$ by approximating the joint distribution $p(x, y)$ and marginal distribution $p(x)$ separately via kernel density estimation (KDE). Given pairs of observations $\{(x_j, y_j)\}_{j=1}^m$, the KDE approximations for the joint and marginal distributions are

$$\widehat{p}(x, y) = \frac{1}{m}\sum_{j=1}^{m} K\left(\frac{x - x_j}{h}\right) K'\left(\frac{y - y_j}{h'}\right), \qquad \widehat{p}(x) = \frac{1}{m}\sum_{j=1}^{m} K\left(\frac{x - x_j}{h}\right), \tag{3}$$

where K and K' are kernel functions with bandwidths $h, h' > 0$, respectively. To approximate the conditional density, the CKDE simply takes the ratio of these two KDE approximations:

$$\widehat{p}(y|x) = \frac{\widehat{p}(x, y)}{\widehat{p}(x)} = \sum_{j=1}^{m} \frac{K\left(\frac{x-x_j}{h}\right) K'\left(\frac{y-y_j}{h'}\right)}{\sum_{\ell=1}^{m} K\left(\frac{x-x_\ell}{h}\right)}. \tag{4}$$

3.2 Kernel Density Bayesian IRL

We now describe the proposed conditional kernel density Bayesian inverse reinforcement learning (KD-BIRL) method.

KD-BIRL uses a CKDE approximation $\widehat{p}_m(s, a \mid R)$ to estimate the likelihood $p(s, a \mid R)$. While the standard form of the CKDE (Equation (4)) uses the difference between two samples (e.g., $x - x_j$) as input to the kernel functions, this difference can be replaced by any suitable distance metric [**?**]. To estimate the joint and marginal distributions, $p(s, a, R)$ and $p(R)$, we must specify two distance functions: one for comparing state-action tuples and one for comparing reward functions. We denote these as $d_s : (\mathcal{S} \times \mathcal{A}) \times (\mathcal{S} \times \mathcal{A}) \to \mathbb{R}_+$ and $d_r : \mathcal{R} \times \mathcal{R} \to \mathbb{R}_+$, respectively, and we discuss specific choices for them later.

The CKDE approximation is then given by

$$\widehat{p}_m(s, a \mid R) = \frac{\widehat{p}_m(s, a, R)}{\widehat{p}_m(R)} = \sum_{j=1}^{m} \frac{K\left(\frac{d_s((s,a),(s_j,a_j))}{h}\right) K\left(\frac{d_r(R,R_j)}{h'}\right)}{\sum_{\ell=1}^{m} K\left(\frac{d_r(R,R_l)}{h'}\right)}, \tag{5}$$

where $h, h' > 0$ are the bandwidth hyperparameters. Similar to earlier work, we assume that each unique state has an independent scalar reward, allowing us to parameterize the reward function as a vector [28, 33]. This vector is the same length as the cardinality of the state space $|S|$, and each element of the vector corresponds to the reward received in that state.

Note that fitting a CKDE for the likelihood $p(s, a|R)$, requires estimating the density across a range of reward functions and state-action pairs. To enable this, we construct an additional set of demonstrations – which we call the *training dataset* $\{(s_j, a_j, R_j)\}_{j=1}^m$ – to augment the observed expert demonstrations $\{(s_i^e, a_i^e)\}_{i=1}^n$ that arise from an agent acting according to the reward $R^\star$. The training dataset contains demonstrations from agents whose policies optimize for reward functions that may be distinct from those of the expert. Each sample in the training dataset is a state-action pair associated with a reward function. There will be many state-action pairs that correspond to the same reward function; therefore, R_j is not unique. Here, we choose k training set reward functions and generate m/k demonstrations from each of the agents' optimal policies. We select the reward functions $R_1, \ldots, R_k \sim u$, where u is a distribution on

Algorithm 1 Kernel Density Bayesian IRL

```
1: Input: MDP $M$, $k$, $m$, $n$, true reward $R^\star$, # MCMC iterations $c$
2: Generate $k$ reward functions $R_1 \ldots, R_k \sim u$.
3: for $q = 1, \ldots, k$ do
4:     $\pi_q := \mathrm{PolicyIteration}(M, R_q)$
5:     Generate $m/k$ demonstrations using agent $\pi_q$
6: end for
7: $\pi_{opt} := \mathrm{PolicyIteration}(M, R^\star)$
8: Generate $n$ demonstrations using agent $\pi_{opt}$
9: for $l = 1, \ldots, c$ do
10:     Sample a reward function $\tilde{R}$ from the posterior distribution
11:     Update the density function of the posterior $\widehat{p_m^n}$
12: end for
13: Output: all sampled reward functions
```

$\mathcal{R}$. Intuitively, the more representative the reward functions $R_1, \ldots, R_k$ are of the reward vector space, the better the resulting density estimate. In our experiments, we use a uniform distribution for u. The goal of the training dataset is to augment the observed trajectories beyond the expert demonstrations, allowing the CKDE to better estimate the posterior distribution across the space of reward functions. However, the posterior distribution is determined largely by the expert demonstrations, not the rewards in the training dataset.

Using the CKDE in Equation (5), we can now estimate the posterior density function of R given n expert demonstrations, m training demonstrations, and prior $p(R)$:

$$\widehat{p}_m^n(R|\{s_i^e, a_i^e\}_{i=1}^n) \propto p(R) \prod_{i=1}^n \widehat{p}_m(s_i^e, a_i^e \mid R) = p(R) \prod_{i=1}^n \sum_{j=1}^m \frac{K\left(\frac{d_s((s_i^e, a_i^e), (s_j, a_j))}{h}\right) K\left(\frac{d_r(R, R_j)}{h'}\right)}{\sum_{\ell=1}^m K\left(\frac{d_r(R, R_l)}{h'}\right)}. \tag{6}$$

Here, we use a uniform prior $p(R)$ and Euclidean distance for d_s, d_r, but these can be altered depending on information known about the reward function or state space in advance [3]. For example, if the reward function is assumed to be linear, the cosine distance is more appropriate for d_r. Several non-uniform priors may be appropriate for $p(R)$ given characteristics of the MDP, including Gaussian [31], Beta [33], and Chinese restaurant process (CRP) [26].

In KD-BIRL, the kernel is chosen to be Gaussian, that is, $K(x) = \exp(-\|x\|^2)$. We choose a Gaussian kernel because it can approximate bounded and continuous functions well. The bandwidth hyperparameters can be chosen using rule-of-thumb procedures [41].

To infer the posterior estimate in Equation (6), we sample rewards from the posterior using MCMC. We note a key computational gain of our approach over BIRL, which also is a sampling-based algorithm: here we only use forward RL to generate the training dataset, rather than in each iteration of MCMC, as the posterior estimate in Equation (6) does not depend on $Q^\star$. This drastically reduces the computational complexity compared to existing Bayesian IRL algorithms that rely on forward RL for every iteration of MCMC sampling, and opens the door for performing Bayesian IRL in complex environments with large state spaces. The full KD-BIRL algorithm is summarized in Algorithm 1.

4 Theoretical guarantees of KD-BIRL

Because KD-BIRL calculates the density function of a true Bayesian posterior distribution (Equation (6)), we can reason about its asymptotic behavior. In particular, we want to ascertain that this posterior estimate contracts as it receives more samples. Because the IRL problem is in general non-identifiable, the "correct" reward function as defined by existing methods [2, 35, 50, 51], is not unique. In this work, we assume that any two reward functions that can produce the same set of observations can be considered equivalent: $R_1 \simeq R_2$ if $\|p(\cdot|R_1) - p(\cdot|R_2)\|_{L_1}$. We then define the *equivalence class* $[R^\star]$ for the correct reward function $R^\star$ as $[R^\star] = \{R \in \mathcal{R} : R \simeq R^\star\}$. As such, the correct posterior distribution is one that places higher mass to the equivalence class of the correct reward function $R^\star$, denoted by $[R^\star]$. Existing formulations of Bayesian IRL have assumed that the reward function $R \in \mathbb{R}^d$ where d is the cardinality of the state space $\mathcal{S}$. In this work, we assume that the reward is a vector, but do not restrict the length to d. We recommend that if $|\mathcal{S}|$ is small and finite, then $d = |\mathcal{S}|$.

We first focus on the likelihood estimation step and show that, when the size of the training dataset m approaches ∞, the likelihood estimated using a CKDE (Equation (5)) converges to the true likelihood $p(s, a \mid R)$.

Lemma 4.1. *Let $h_m, h'_m > 0$ be the bandwidths chosen for the CKDE. Assume that both $p(s,a|R)$ and $p(R)$ are square integrable and twice differentiable with a square integrable and continuous second order derivative, and that $mh_m^{p/2} \to \infty$ and $m{h'_m}^{p/2} \to \infty$ as $m \to \infty$. Then,*

$$\widehat{p}_m(s,a|R) \xrightarrow[m\to\infty]{P} p(s,a|R),\ \forall (s,a,R) \in \mathcal{S} \times \mathcal{A} \times \mathcal{R}.$$

Proof. We now use the continuity of the likelihood, the finite sample analysis of multivariate kernel density estimators in Wand and Jones [46][Section 4.4, Equation 4.16](Appendix A.9) which defines the Mean Integrated Square Error (MISE) of the density function, and Theorem 1(Appendix A.8) of Chacón and Duong [11][Section 2.6-2.9], which asserts that as the sample size increases, the mean of the density estimator converges and variance prevents the mean from exploding. We can use Theorem 1 because we assume that the density function is square integrable and twice differentiable, and that the bandwidth approaches 0 as the dataset size increases. Then, up to a constant, for a given state-action pair (s,a),

$$\frac{1}{m}\sum_{j=1}^{m} e^{-d_s((s,a),(s_j,a_j))^2/(2h)} e^{-d_r(R,R_j)^2/(2h')} \xrightarrow[m\to\infty]{P} p(s,a,R).$$

The same holds true for d_r, $\frac{1}{m}\sum_{\ell=1}^{m} e^{-d_r(R,R_\ell)^2/(2h')} \xrightarrow[m\to\infty]{P} p(R)$. By the Continuous Mapping Theorem [25], we conclude that

$$\widehat{p}_m(s,a|R) \xrightarrow[m\to\infty]{P} \frac{p(s,a,R)}{p(R)} = p(s,a|R).$$

□

Lemma 4.1 verifies that we can estimate the unknown likelihood using a CKDE, which opens the door to Bayesian inference. Next, we show that as n, the size of expert demonstrations, and m, the size of the training dataset, grow, the posterior distribution generated using KD-BIRL contracts to the equivalence class of the correct reward $[R^\star]$, where $R^\star$ was used to generate the expert demonstrations $\{s_i^e, a_i^e\}_{i=1}^n$.

Theorem 4.2. *Assume the prior for R, denoted by Π, satisfies $\Pi(\{R : \mathrm{KL}(R, R^\star) < \epsilon\}) > 0$ for any $\epsilon > 0$, where KL is the Kullback–Leibler divergence. Assume $\mathcal{R} \subseteq \mathbb{R}^d$ is a compact set. Then, the posterior measure corresponding to the posterior density function $\widehat{p}_m^n$ defined in Equation (6), denoted by Π_m^n, is consistent w.r.t. the L_1 distance; that is,*

$$\Pi_m^n(\{R : \|p(\cdot|R) - p(\cdot|R^\star)\|_{L_1} < \epsilon\}) \xrightarrow[n\to\infty]{m\to\infty} 1.$$

Proof. By Lemma 4.1, as $m \to \infty$, $\widehat{p}_m$ converges to the true likelihood, so we can adopt existing tools from Bayesian asymptotic theory.

We first define an equivalence relation on $\mathcal{R}$, denoted by $\simeq$:

$$R_1 \simeq R_2 \text{ iff } p(\cdot|R_1) = p(\cdot|R_2),\ a.e.$$

Note that $\simeq$ satisfies reflexivity, symmetry, and transitivity, and is therefore an equivalence relation. We denote the equivalence class by $[\cdot]$, that is, $[R] = \{R' : R' \simeq R\}$, and the quotient space is defined as $\widetilde{\mathcal{R}} := \mathcal{R}/\!\simeq\, = \{[R] : R \in \mathcal{R}\}$. The corresponding canonical projection is denoted by $\pi : \mathcal{R} \to \widetilde{\mathcal{R}},\ R \mapsto [R]$. Then, the projection π induces a prior distribution on $\widetilde{\mathcal{R}}$ denoted by $\widetilde{\Pi}$: $\widetilde{\Pi}(A) := \Pi(\pi^{-1}(A))$. Moreover, $\widetilde{\mathcal{R}}$ admits a metric $\widetilde{d}$:

$$\widetilde{d}([R_1],[R_2]) := \|p(\cdot|R_1) - p(\cdot|R_2)\|_{L^1}.$$

Because this metric uses the L^1 norm, it satisfies symmetry and triangular inequality. Additionally, it is true that

$$\widetilde{d}([R_1],[R_2]) = 0 \iff p(\cdot|R_1) = p(\cdot|R_2),\ a.e. \iff R_1 \simeq R_2 \iff [R_1] = [R_2],$$

so $\widetilde{d}$ fulfills the identity of indiscernibles principle. As a result, $\widetilde{d}$ is a valid distance metric on $\widetilde{\mathcal{R}}$.

Then consider the following Bayesian model:

$$(s,a)|[R] \simeq p(s,a|[R]),\ [R] \in \widetilde{\mathcal{R}},\ [R] \simeq \widetilde{\Pi}.$$

This model is well-defined since $p(s,a|[R])$ is independent of the representative of $[R]$ by the definition of the equivalence class. Observe that $\mathrm{KL}(R, R^*) = \mathrm{KL}([R],[R^*])$ by the definition of the equivalence class. Then, let

$A = \{[R] : \mathrm{KL}([R], [R^*]) < \epsilon\} \subset \widetilde{\mathcal{R}}$. We can define $\pi^{-1}(A) = \{R \in \mathcal{R} : \mathrm{KL}(R, R^*) < \epsilon\} \subset \mathcal{R}$. As a result, $\widetilde{\Pi}(\{[R] : \mathrm{KL}([R], [R^*]) < \epsilon\}) = \Pi(\{R : \mathrm{KL}(R, R^*) < \epsilon\}) > 0$ for any $\epsilon > 0$, that is, the KL support condition is satisfied . Moreover, the mapping $[R] \to p(\cdot|R)$ is one-to-one. Because the Bayesian model is parameterized by $[R]$ and we assume that $\mathcal{R}$ is a compact set, by van der Vaart [44][Lemma 10.6](Appendix A.11) there exist consistent tests as required in Schwartz's Theorem . Then, by Schwartz [39](Appendix A.10), the posterior $\widetilde{\Pi}_n$ on $\widetilde{\mathcal{R}}$ is consistent. That is, for any $\epsilon > 0$, $\widetilde{\Pi}_m^n(\{[R] : \widetilde{d}([R], [R^*]) < \epsilon\}) \xrightarrow[n\to\infty]{m\to\infty} 1$. Put in terms of the original parameter space,

$$\Pi_m^n(\{R : \|p(\cdot|R) - p(\cdot|R^\star)\|_{L_1} < \epsilon)\}) = \widetilde{\Pi}_m^n(\{[R] : \widetilde{d}([R], [R^*]) < \epsilon\}) \xrightarrow[n\to\infty]{m\to\infty}, \forall \epsilon > 0.$$

□

Theorem 4.2 implies that the posterior Π_m^n assigns higher mass to equivalence class of $R^\star$, denoted by $[R^\star]$, where $R^\star$ is a representative member of this equivalence class. With theoretical support in hand, we next show that KD-BIRL empirically outperforms previous approaches.

5 Experiments

Here, we evaluate the accuracy and computational efficiency of KD-BIRL. We compare the performance of KD-BIRL to a recent method, AVRIL [12], and the original Bayesian IRL algorithm (BIRL [33]). We demonstrate results using two environments: a 2×2 Gridworld environment [7] and a Sepsis management clinical environment [4]. We choose the Euclidean distance for the distance metrics d_r and d_s in both environments and to infer the posterior distribution, we use STAN [43], which implements a Hamiltonian Monte Carlo algorithm.

Previous work uses Expected Value Difference (EVD) [8, 13, 14, 24] to quantitatively evaluate these methods. EVD is defined as $|V^*(r^A) - V^{\pi^*(r^L)}(r^A)|$ where $V^\pi = \sum_s p_0(s)V^\pi$ is the value of policy π with initial state distribution p_0, r^A is the true reward and r^L is the learned reward. The value of the policy π is the value of the executing the policy, defined as $V^\pi = \mathbb{E}[\sum_{t=0}^\infty \gamma^t R(s_t, a_t)|\alpha, \pi]$. We use EVD because it allows us to compare KD-BIRL to related methods. In general, it is difficult to define a practical equivalence class for reward functions without assuming anything about the way that they are parameterized. In Section 4 we define the equivalence class of the correct reward function $[R^\star]$ as the set of reward functions that produce the same observations as the correct reward function $R^\star$. In large state spaces, it is difficult to quantify similar behaviors. As a result, we use EVD, which quantifies similarity based on the value of the behavior trajectory. EVD allows us to identify whether we can recover the same total reward as the agent used to generate the expert observations; the lower the EVD, the better our learned reward recapitulates the expert reward (see Appendix A.4 for more details).

5.1 Gridworld environment

We begin our experiments in a 2×2 Gridworld environment. The MDP here is defined by the grid's 2×2 discrete state space $\mathcal{S}$; the action space $\mathcal{A} = \{\texttt{NO ACTION}, \texttt{UP}, \texttt{RIGHT}, \texttt{LEFT}, \texttt{DOWN}\}$; and the true reward function $R = R^\star$, which is used to generate synthetic data but is unobserved by the IRL algorithms, is a vector of length 4. We structure the reward function such that each state has an independent scalar reward parameter. We specify the domain of each of these parameters to be the unit interval; thus, each feasible reward function can be represented by a vector $\mathbf{R} \in [0, 1]^4$.

To fit the BIRL and AVRIL posteriors, we first generate 200 expert demonstration trajectories. BIRL uses MCMC sampling to generate reward samples; we use 4000 burn-in iterations, and generate reward samples during the subsequent 1000 iterations. BIRL uses one hyperparameter, α, which indicates confidence in the agent's ability to choose optimal actions; we set this value to 1. AVRIL uses the expert demonstrations to train an agent to predict optimal actions. To generate reward samples, we use the AVRIL agent to predict the variational mean and standard deviation of reward in each of the four states, and use these statistics to generate reward samples, assuming that the samples arise from a multivariate normal distribution. By default, AVRIL sets α to 1. For KD-BIRL's posterior distribution, we specify a uniform prior on each of the reward function parameters, $r_s \sim \mathrm{Unif}(0, 1)$ for $s = 1, \ldots, 4$.

5.1.1 KD-BIRL posterior contraction

As discussed above, our theoretical results suggest that posterior obtained by KD-BIRL contracts to the true reward function. We now empirically demonstrate this contraction as the size of the training dataset m and the size of the expert behavior dataset n increase, using EVD. If the posterior distribution places higher mass on reward functions that are equivalent to the true reward function, the reward samples from the distribution result in lower EVDs. Here,

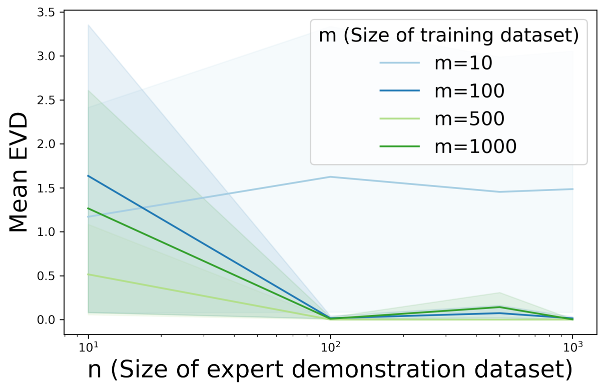

Figure 1: We demonstrate the contraction of KD-BIRL's posterior by increasing the size of our datasets and reporting the mean EVD 95% confidence intervals (using shading) for the resulting distributions. m is the size of the training dataset and n is the size of the expert demonstrations dataset. As m and n increase, the EVD converges to 0, indicating that KD-BIRL posterior contracts to the true reward function.

we fit KD-BIRL for the true reward function $\mathbf{R} = [0.1, 0.2, 0.3, 0.4]$ across a range of values for m and n. For each combination of m and n, we fit KD-BIRL ten times and compute the mean expected value difference (EVD) for 100 samples from each posterior distribution. We find that, as the sample sizes increase, the EVD converges to 0, except for trivially small sample sizes (i.e., $m = 10$) (Figure 1). This result empirically validates our theoretical claims and indicates that KD-BIRL's posterior approximation is able to recover a valid distribution over reward functions for a 2×2 Gridworld environment.

5.1.2 KD-BIRL posterior estimates require fewer instances of Q-learning

We now quantify the computational complexity associated with learning posterior distributions given trajectories generated using the true reward $\mathbf{R} = [0.1, 0.2, 0.3, 0.4]$ using KD-BIRL, BIRL, and AVRIL. As discussed earlier, for previous methods, much of the computational cost associated with learning these posterior distributions arises from repeated instances of forward RL learn $Q^\star$. KD-BIRL only uses forward RL during dataset generation. Alternatively, BIRL requires forward RL for every iteration of MCMC sampling; several thousand iterations are required for the sampler to converge. AVRIL only uses one instance of forward RL, and uses a variational inference approach that uses a neural network to parameterize the variational family. Here, we vary the number of iterations of MCMC sampling for BIRL (each iteration requires one instance of forward RL), and the number of reward functions in the training dataset for KD-BIRL (each of which requires forward RL), and plot the EVDs for reward samples from the resulting posterior distributions. Our results indicate that, with fewer instances of forward RL, KD-BIRL reward samples better replicate the behavior of the expert demonstrations than those of BIRL; consequently, even though AVRIL requires fewer instances of forward RL, it is at the expense of accuracy in the posterior distribution, as highlighted by the stagnant EVD (Figure 2).

5.1.3 KD-BIRL's posterior distribution outperforms related approaches

Next, we investigate and visualize KD-BIRL's estimated posterior distribution, and compare it to the posteriors recovered by AVRIL and BIRL. We first show density plots to visualize posterior samples for the three methods. All methods attempt to recover the reward function $\mathbf{R} = [0, 0, 0, 1]$. We find that the reward function samples from KD-BIRL's posterior are numerically closer to the true reward function than those from the BIRL and AVRIL posterior distributions (Figure 3). We then tested KD-BIRL and the related approaches across a variety of true reward functions with varying target states by calculating EVDs for each reward sample. We find that KD-BIRL's posterior distributions result in EVDs that perform equally well, if not better, than those of BIRL and AVRIL across all the reward functions chosen (Table 1).

5.2 Modified Sepsis Environment

In addition to a Gridworld environment, we study the utility of our algorithm in a more real-world setting that models Sepsis treatment. Sepsis, or blood poisoning, arises when the body responds to infection in a way that is harmful to

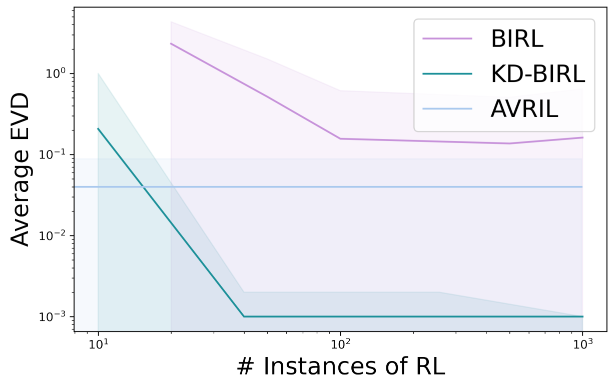

Figure 2: Mean EVD for reward samples from posterior distributions for the reward function $\mathbf{R} = [0.1, 0.2, 0.3, 0.4]$ in a 2×2 Gridworld environment. KD-BIRL requires fewer instances of RL (Q-learning) to generate posterior distribution reward samples that have an EVD of 0. The BIRL reward samples continue to have high EVDs even with many instances of RL, and AVRIL only performs RL once, but the resulting EVDs stagnate.

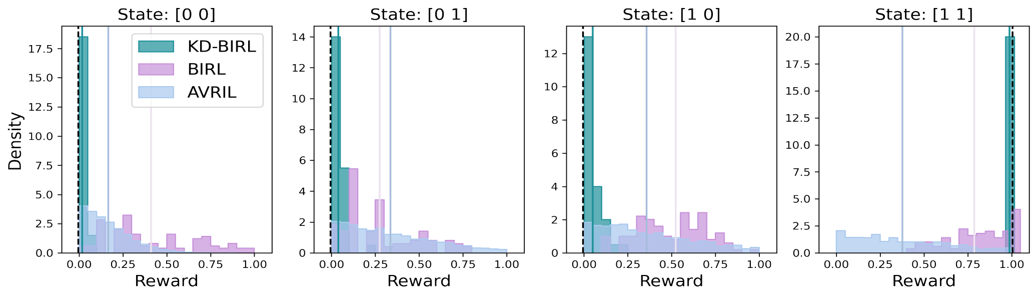

Figure 3: Visualization of the marginal posterior distribution at each state in Gridworld 2×2 environment for the reward function $\mathbf{R} = [0, 0, 0, 1]$. The dashed vertical lines show the true reward for each state. Each algorithm's mean estimated reward in each state is shown using vertical colored lines. KD-BIRL's marginal posterior distributions are more concentrated around the true reward compared to those of BIRL and AVRIL.

its own tissues and organs [10]. This Sepsis treatment environment is a simulator based on de-identified data from MIMIC-III [16, 21], a database of electronic health records (EHR) from the Beth Israel Deaconess Medical Center in Boston, Massachusetts. In the original simulator, each state is a 48-dimensional vector that contains information about various physiological characteristics describing the patient's state. There are 24 possible actions, each corresponding to a different dosage of medication. The transition function is learned using deep RL models [32] and the reward in the original environment is 0 for all timesteps except the last one (+15 if the patient is alive by the time of discharge and -15 otherwise).

We modify this environment slightly in order to effectively apply our algorithm. Sepsis treatment depends heavily on organ failure metrics, as fast increases in these metrics implies organ failure [32]. Since we have observed that, in the Gridworld environment, KD-BIRL can successfully model a reward that is a function of the state space, we modify the Sepsis environment's state space to consist solely of three Sequential Organ Failure Assessment (SOFA) [22] covariates (*sofa*, *quick sofa*, and *quick sofa systolic blood pressure score*). This is similar to the featurized reward vector approach [1], with manually selected features. This also reduces the size of the reward function space R^d to be more manageable, since the cardinality of S in this environment, and most other clinical environments, is likely infinite. Our modified state is now a vector of length 3, and the reward is a linear combination of the difference between the state at timestep t and timestep $t + 1$,

$$R(t) = \begin{bmatrix} a \\ b \\ c \end{bmatrix}^\top \begin{bmatrix} cov1_t - cov1_{t+1} \\ cov2_t - cov2_{t+1} \\ cov3_t - cov3_{t+1}, \end{bmatrix}$$

where a, b, c are the weights, and $cov1, cov2, cov3$ are the three metrics of organ failure. We choose the true (unobserved) weights to be $[a = 0.8, b = 0.6, c = 0.4]$.

Environment	Correct reward $\mathbf{R}^\star$	EVD 95% Confidence Interval		
		KD-BIRL	AVRIL	BIRL
Gridworld	$[0, 0, 0, 1]$	**(0, 0)**	(0.185, 0.467)	**(0, 0)**
Gridworld	$[0.1, 0.2, 0.3, 0.4]$	**(0, 0)**	(0.353, 0.466)	(0.136, 0.181)
Gridworld	$[0, 0, 1, 0]$	**(0, 0)**	(0.348, 0.366)	**(0, 0)**
Gridworld	$[0.2, 0.6, 0.4, 0.8]$	**(0, 0.113)**	(0.122, 0.126)	(0.198, 0.2065)
Gridworld	$[0, 0.5, 0.55, 0.9]$	**(0, 0)**	(0.018, 0.018)	(0.048, 0.049)
Sepsis	[0.8, 0.6, 0.4]	**(0.116, 0.368)**	(0.377, 0.527)	N/A

Table 1: EVD 95% confidence intervals for all environments over several reward functions. The best performing method is in bold for each environment.

We compare our method to AVRIL and avoid fitting BIRL because the process requires several days. In this environment, we use a different procedure to generate reward samples from AVRIL than our approach in the Gridworld environment because the Sepsis environment has a continuous state space. AVRIL parameterizes the estimated reward function using θ, the parameters of a neural network. While it is possible to generate a reward sample from θ, the reward sample is a function of θ, which is not comparable to a set of weights a, b, c, as generated from KD-BIRL. To generate EVDs for AVRIL, we train an AVRIL agent to estimate the reward using a set of 200 expert observations. Then, we generate trajectories using the AVRIL agent's recommended actions starting from an initial state; we calculate EVDs using these trajectories and compare them to those generated using reward samples from KD-BIRL (Table 1). We find that KD-BIRL generates reward samples with lower EVDs than the corresponding trajectories from AVRIL. This indicates that KD-BIRL estimates a posterior distribution that is concentrated around the equivalence class of the true reward $R^\star$ used in the expert behavior demonstrations.

6 Discussion and Conclusion

In this work we present kernel density Bayesian inverse reinforcement learning (KD-BIRL), an efficient Bayesian IRL framework. KD-BIRL improves upon existing methods for IRL by estimating a posterior distribution on the reward function while avoiding Q-learning for every iteration of MCMC sampling, with theoretical guarantees of posterior consistency. We show that KD-BIRL generates more empirically accurate posteriors and is more computationally efficient than competing methods in a Gridworld environment. Additionally, we demonstrate that KD-BIRL is better suited to perform IRL on a complex healthcare task than a related Bayesian IRL method. Taken together, our results suggest that, in complex environments such as medical domains, KD-BIRL can enable a tractable and accurate probabilistic description of clinical objectives that is not possible with current methods.

Several future direction remain. This work is best-suited for on-policy (i.e., simulation) environments, and we believe that additional work must be done to apply it directly to off-policy environments such as retrospective clinical decision-making settings. In particular, we would need behavior demonstrations from multiple agents, so being able to associate actions with an agent (i.e., clinician) is necessary. Additionally, the particular choice of distance metrics and hyperparameters used in the CKDE is likely to depend on the environment and reward function type; we believe that additional experimentation must be done to adapt this to different environments. Furthermore, a limitation of conventional CKDE is that it performs poorly in high dimensions. One solution is to consider a modified version of CKDE to speed it up [19]. Another solution is to replace the CKDE with another nonparametric conditional density estimator [18, 36, 37, 47]. Among these nonparametric alternatives, Gaussian process-based methods are of great interest due to their flexibility and interpretability [15]. Because KD-BIRL is a framework that estimates the likelihood as a conditional density, it can be easily modified to accommodate other choices for the CKDE. Finally, it is of interest to re-parameterize the reward function. Specifically, rather than using a vector where each element represents a possible state, we can use a feature-based reward function where the vector is represented as a latent feature embedding. This is particularly useful for the continuous or infinite state spaces we are interested in.

Acknowledgments and Disclosure of Funding

We thank Alex Chan for providing code associated with the AVRIL method. This work was funded by the Helmsley Trust grant AWD1006624, NIH NCI 5U2CCA233195, NIH NHLBI R01 HL133218, and NSF CAREER AWD1005627. BEE is on the SAB of Creyon Bio, Arrepath, and Freenome. D. Cai was supported in part by a Google Ph.D. Fellowship in Machine Learning.

References

[1] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *The Twenty-First International Conference on Machine Learning*, ser. ICML '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 1.

[2] ——, "Apprenticeship learning via inverse reinforcement learning," in *The twenty-first international conference on Machine learning*, 2004, p. 1.

[3] S. Adams, T. Cody, and P. A. Beling, "A survey of inverse reinforcement learning," *Artificial Intelligence Review*, Feb 2022. [Online]. Available: https://doi.org/10.1007/s10462-021-10108-x

[4] P. H. Amirhossein Kiani, Tianli Ding, "Gymic: An openai gym environment for simulating sepsis treatment for icu patients," https://github.com/akiani/rlsepsis234, 2019.

[5] S. Balakrishnan, Q. P. Nguyen, B. K. H. Low, and H. Soh, "Efficient exploration of reward functions in inverse reinforcement learning via Bayesian optimization," *arXiv preprint arXiv:2011.08541*, 2020.

[6] P. G. Bissiri, C. C. Holmes, and S. G. Walker, "A general framework for updating belief distributions," *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, vol. 78, no. 5, 2016. [Online]. Available: http://www.jstor.org/stable/44682909

[7] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[8] D. S. Brown and S. Niekum, "Efficient probabilistic performance bounds for inverse reinforcement learning," in *AAAI*, 2018.

[9] D. S. Brown, R. Coleman, R. Srinivasan, and S. Niekum, "Safe imitation learning via fast bayesian reward inference from preferences," 2020. [Online]. Available: https://arxiv.org/abs/2002.09089

[10] CDC, "Sepsis is a medical emergency. act fast." Apr 2022. [Online]. Available: https://www.cdc.gov/sepsis/what-is-sepsis.html

[11] J. E. Chacón and T. Duong, *Multivariate kernel smoothing and its applications*. Chapman and Hall/CRC, 2018.

[12] A. J. Chan and M. van der Schaar, "Scalable Bayesian inverse reinforcement learning," in *International Conference on Learning Representations*, 2021.

[13] J. Choi and K.-E. Kim, "Bayesian nonparametric feature construction for inverse reinforcement learning," in *The Twenty-Third International Joint Conference on Artificial Intelligence*, ser. IJCAI '13. AAAI Press, 2013.

[14] J. Choi and K.-e. Kim, "Nonparametric Bayesian inverse reinforcement learning for multiple reward functions," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/140f6969d5213fd0ece03148e62e461e-Paper.pdf

[15] V. Dutordoir, H. Salimbeni, M. Deisenroth, and J. Hensman, "Gaussian process conditional density estimation," *arXiv preprint arXiv:1810.12750*, 2018.

[16] A. L. Goldberger, L. A. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, "Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals," *Circulation*, 2000.

[17] D. Hadfield-Menell, S. Milli, P. Abbeel, S. Russell, and A. Dragan, "Inverse reward design," 2017. [Online]. Available: https://arxiv.org/abs/1711.02827

[18] F. Hinder, V. Vaquet, J. Brinkrolf, and B. Hammer, "Fast non-parametric conditional density estimation using moment trees," in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2021, pp. 1–7.

[19] M. P. Holmes, A. G. Gray, and C. L. Isbell, "Fast nonparametric conditional density estimation," in *The Twenty-Third Conference on Uncertainty in Artificial Intelligence*, ser. UAI'07. Arlington, Virginia, USA: AUAI Press, 2007.

[20] R. Izbicki and A. B. Lee, "Nonparametric conditional density estimation in a high-dimensional regression setting," *Journal of Computational and Graphical Statistics*, vol. 25, no. 4, pp. 1297–1316, 2016. [Online]. Available: https://doi.org/10.1080/10618600.2015.1094393

[21] A. E. Johnson, T. J. Pollard, L. Shen, H. L. Li-Wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark, "Mimic-iii, a freely accessible critical care database," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[22] A. E. Jones, S. Trzeciak, and J. A. Kline, "The sequential organ failure assessment score for predicting outcome in patients with severe sepsis and evidence of hypoperfusion at the time of emergency department presentation," *Critical care medicine*, vol. 37, no. 5, p. 1649–1654, May 2009. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2703722/

[23] J. Z. Kolter, P. Abbeel, and A. Y. Ng, "Hierarchical apprenticeship learning with application to quadruped locomotion," in *Advances in Neural Information Processing Systems*. Citeseer, 2008, pp. 769–776.

[24] S. Levine, Z. Popović, and V. Koltun, "Nonlinear inverse reinforcement learning with Gaussian processes," in *The 24th International Conference on Neural Information Processing Systems*, ser. NIPS'11. Red Hook, NY, USA: Curran Associates Inc., 2011, p. 19–27.

[25] H. B. Mann and A. Wald, "On stochastic limit and order relationships," *The Annals of Mathematical Statistics*, vol. 14, no. 3, pp. 217–226, 1943.

[26] B. Michini and J. P. How, "Bayesian nonparametric inverse reinforcement learning," in *Machine Learning and Knowledge Discovery in Databases*, P. A. Flach, T. De Bie, and N. Cristianini, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 148–163.

[27] ——, "Improving the efficiency of Bayesian inverse reinforcement learning," in *2012 IEEE International Conference on Robotics and Automation*, May 2012, pp. 3651–3656.

[28] B. Michini, M. Cutler, and J. P. How, "Scalable reward learning from demonstration," in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 303–308.

[29] K. Mombaur, A. Truong, and J.-P. Laumond, "From human to humanoid locomotion—an inverse optimal control approach," *Autonomous Robots*, vol. 28, no. 3, pp. 369–383, 2010.

[30] A. Y. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *in Proc. 17th International Conf. on Machine Learning*. Morgan Kaufmann Publishers Inc., 2000, pp. 663–670.

[31] Q. Qiao and P. A. Beling, "Inverse reinforcement learning with Gaussian process," *2011 American Control Conference*, pp. 113–118, 2011.

[32] A. Raghu, M. Komorowski, L. A. Celi, P. Szolovits, and M. Ghassemi, "Continuous state-space models for optimal sepsis treatment: a deep reinforcement learning approach," in *MLHC*, 2017.

[33] D. Ramachandran and E. Amir, "Bayesian inverse reinforcement learning," in *The 20th International Joint Conference on Artifical Intelligence*, 2007, p. 2586–2591.

[34] N. Ratliff, D. Bradley, J. A. Bagnell, and J. Chestnutt, "Boosting structured prediction for imitation learning," in *Advances in Neural Information Processing Systems*, 2006, p. 1153–1160.

[35] N. D. Ratliff, J. A. Bagnell, and M. A. Zinkevich, "Maximum margin planning," in *The 23rd International Conference on Machine Learning*, ser. ICML '06. New York, NY, USA: Association for Computing Machinery, 2006, p. 729–736. [Online]. Available: https://doi.org/10.1145/1143844.1143936

[36] A. L. Rojas, C. R. Genovese, C. J. Miller, R. Nichol, and L. Wasserman, "Conditional density estimation using finite mixture models with an application to astrophysics," *Center for Automatic Learning and Discovery, Department of Statistics, Carnegie Mellon University*, 2005.

[37] J. Rothfuss, F. Ferreira, S. Walther, and M. Ulrich, "Conditional density estimation with neural networks: Best practices and benchmarks," *arXiv preprint arXiv:1903.00954*, 2019.

[38] C. A. Rothkopf and D. H. Ballard, "Modular inverse reinforcement learning for visuomotor behavior," *Biol. Cybern.*, vol. 107, no. 4, p. 477–490, aug 2013.

[39] L. Schwartz, "On bayes procedures," *Zeitschrift für Wahrscheinlichkeitstheorie und Verwandte Gebiete*, vol. 4, pp. 10–26, 1965.

[40] J. Shawe-Taylor and R. C. Williamson, "A PAC analysis of a Bayesian estimator," in *The tenth annual conference on Computational learning theory*, 1997, pp. 2–9.

[41] B. W. Silverman, *Density Estimation for Statistics and Data Analysis*. London: Chapman & Hall, 1986.

[42] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.

[43] S. D. Team, "Stan modeling language users guide and reference manual, version 2.29," 2011. [Online]. Available: https://mc-stan.org/

[44] A. van der Vaart, *Asymptotic Statistics*. Cambridge University Press, 2000, vol. 3.

[45] A. Šošić, A. M. Zoubir, E. Rueckert, J. Peters, and H. Koeppl, "Inverse reinforcement learning via nonparametric spatio-temporal subgoal modeling," *J. Mach. Learn. Res.*, vol. 19, no. 1, p. 2777–2821, jan 2018.

[46] M. P. Wand and M. C. Jones, *Kernel smoothing*. CRC press, 1994.

[47] Z. Wang and D. W. Scott, "Nonparametric density estimation for high-dimensional data—algorithms and applications," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 11, no. 4, p. e1461, 2019.

[48] T. Zhang, "From epsilon-entropy to KL-entropy: Analysis of minimum information complexity density estimation," *The Annals of Statistics*, vol. 34, no. 5, oct 2006. [Online]. Available: https://doi.org/10.1214/009053606000000704

[49] J. Zheng, S. Liu, and L. M. Ni, "Robust Bayesian inverse reinforcement learning with sparse behavior noise," in *Association for the Advancement of Artificial Intelligence*, ser. AAAI'14. AAAI Press, 2014, p. 2198–2205.

[50] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey *et al.*, "Maximum entropy inverse reinforcement learning." in *Association for the Advancement of Artificial Intelligence (AAAI)*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[51] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Human behavior modeling with maximum entropy inverse optimal control," in *AAAI Spring Symposium: Human Behavior Modeling*, 2009.

A Appendix

A.1 Broader impacts

The potential negative societal impacts of our work can be summarized as the harm associated with using IRL. IRL, especially when used in a clinical setting, must be carefully tuned before any conclusions can be made about the agent's objectives. For example, if there can be many reward functions that explain a clinician's behavior, it can be misleading to only identify one of them. Additionally, since there is no ground-truth posterior distribution in these settings, IRL algorithms should only be used with good uncertainty metrics.

A.2 Code

Our experiments were run on an internally-hosted cluster using a 320 NVIDIA P100 GPU whose processor core has 16 GB of memory hosted. Our experiments used a total of approximately 200 hours of compute time. Our code uses the MIT License, and is available at https://github.com/bee-hive/kdbirl.

A.3 Rationality conditions for posterior distribution updates

Bissiri et al. [6] make several assumptions before guaranteeing rational updates to a posterior distribution. These are:

Assumption 1.

$$\psi[l(\theta, x_2), \psi\{l(\theta, x_1), \pi(\theta)\}] \equiv \psi\{l(\theta, x_1) + l(\theta, x_2), \pi(\theta)\}$$

where θ is the parameter of interest, $\pi(\theta)$ is the prior distribution on θ, l is a loss function, ψ is the update function, and x_1, x_2 are data points.

Assumption 2. *For any set $A \subset \Theta$,*

$$\frac{\psi\{l(\theta, x), \pi(\theta)\}}{\int_A \psi\{l(\theta, x), \pi(\theta)\} d\theta} = \psi\{l(\theta, x), \pi_A(\theta)\}$$

where π_A is π normalized to A.

Assumption 3. *Lower evidence for a state should yield smaller posterior probabilities under the same prior. So, if for some $A \subset \Theta$, $l(\theta, x) > l(\theta, y)$ for $\theta \in A \subset \Theta$ and $l(\theta, x) = l(\theta, y)$ for $\theta \in A^c$, then*

$$\int_A \psi\{l(\theta, x), \pi(\theta)\} d\theta < \int_A \psi\{l(\theta, y), \pi(\theta)\} d\theta.$$

Assumption 4. *If $l(\theta, x) \equiv$ constant, then $\psi\{l(\theta, x), \pi(\theta)\} \equiv \pi(\theta)$.*

Assumption 5. *If $\tilde{l}(\theta, x) = l(\theta, x) + c$ for some constant c, then*

$$\psi\{\tilde{l}(\theta, x), \pi(\theta)\} = \psi\{l(\theta, x), \pi(\theta)\}.$$

If ψ is a Q-value function, which can be parameterized by anything from a linear model to a deep neural network, the rationality of the subsequent posterior updates is not discussed in previous work. In particular, it is not possible to verify if appendix A.3 is true.

A.4 Calculating Expected Value Difference (EVD)

The procedure to calculate EVD confidence intervals varies depending on the method. For all methods, this process requires a set of reward samples. Because KD-BIRl and BIRL both use MCMC sampling, we can use the reward samples generated from each iteration. Because AVRIL does not use MCMC, we first use the AVRIL agent to estimate the variational mean and standard deviation of the reward vector. Using these statistics, we then assume that the reward samples arise from a multivariate normal distribution and generate samples according to the mean and standard deviation.

Once we have samples, we can then calculate EVD and 95% confidence intervals. Recall that EVD is defined as $|V^*(r^A) - V^{\pi^*(r^L)}(r^A)|$ where r^A is the ground truth known reward and r^L is the learned reward. For a given method, we calculate 95% confidence intervals across the EVD for each of the reward samples. Given a reward sample, we can calculate EVD, where r^L is the sample, and r^A is the known reward. To determine the value of the policy optimizing for a particular reward function, we train an optimal agent for that reward function, and generate 200 demonstrations characterizing its behavior; the value is then the average reward received across those demonstrations. Finally, we calculate the difference between the value of the policy for r^A and r^L, and report confidence intervals across these values for all samples for a given method.

A.5 Choosing bandwidth hyperparameters for KD-BIRL

Here, we visualize the effect of the bandwidth hyperparameters h and h' on KD-BIRL's posterior distribution for the reward function $\mathbf{R} = [0, 0, 0, 1]$, marginalized at state $[1, 1]$ (Figure 4). Recall that h and h' correspond to the bandwidth for the state-action pairs and reward functions, respectively. Note that the density of reward samples varies more with h' than h, but that it is important to tune both these hyperparameters because the resulting posterior distributions can change substantially.

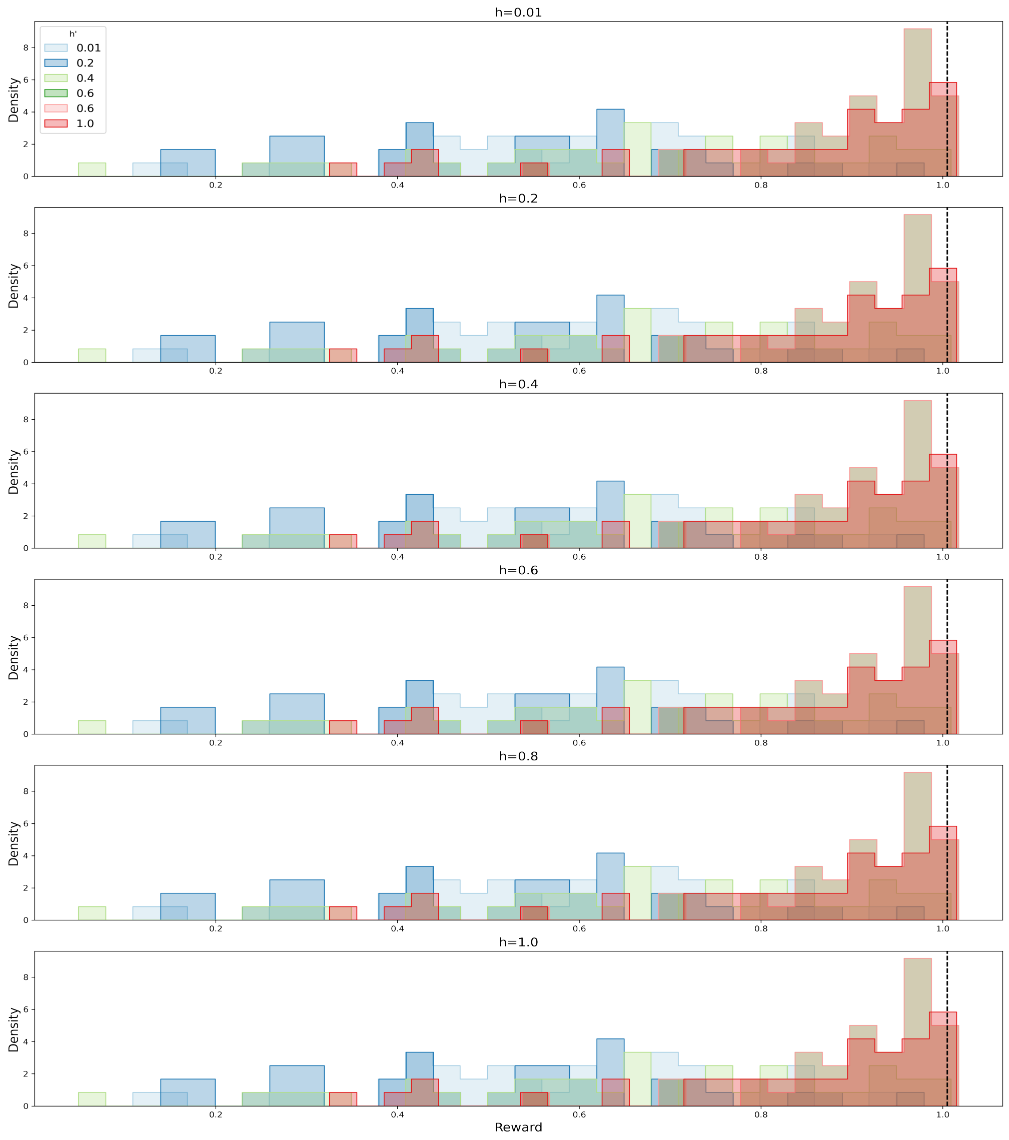

Figure 4: Density plots of the KD-BIRL posterior distributions for the reward function $\mathbf{R} = [0, 0, 0, 1]$ when varying bandwidth hyperparameters, marginalized at state $[1, 1]$. We vary h and h', two hyperparameters that correspond to the bandwidth for the state-action pairs and reward functions.

A.6 Trace plots for KD-BIRL

Here, we visualize trace plots for the posterior distribution from KD-BIRL corresponding to the Gridworld reward functions $\mathbf{R}[0, 0, 0, 1]$ (Figure 5) and $\mathbf{R} = [0.1, 0.2, 0.3, 0.4]$ (Figure 6). Our trace plots display how MCMC samples numerically vary across iterations; these plots indicate that MCMC has converged for both reward functions.

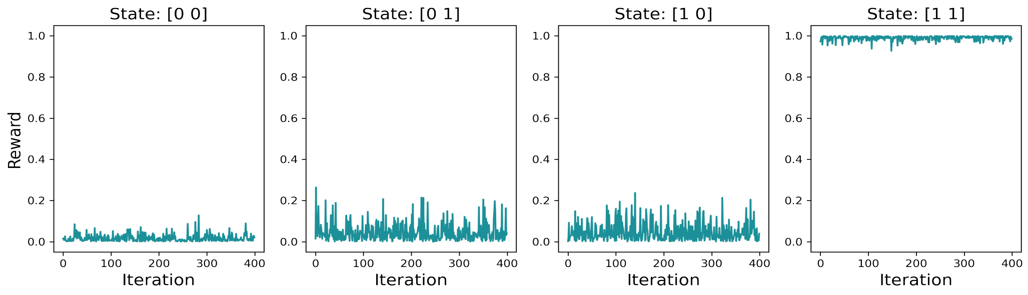

Figure 5: Trace plots for KD-BIRL posterior distribution for Gridworld 2×2 environment reward function $\mathbf{R} = [0, 0, 0, 1]$.

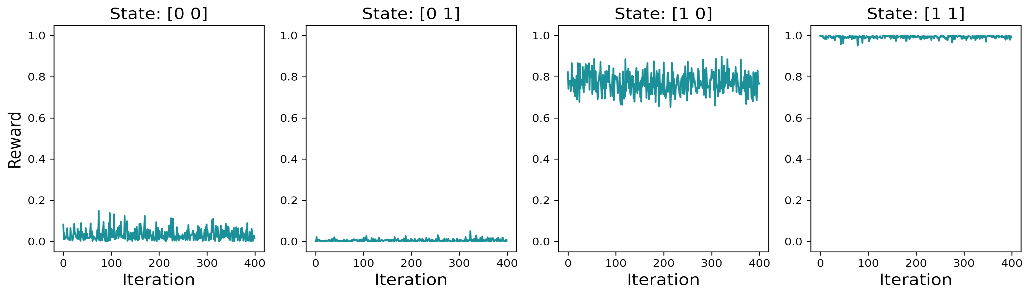

Figure 6: Trace plots for KD-BIRL posterior distribution for Gridworld 2×2 environment reward function $\mathbf{R} = [0.1, 0.2, 0.3, 0.4]$.

A.7 Joint Distribution Plots

Here, we show the pairwise joint distribution across all states in a Gridworld 2×2 environment for the posterior distributions from all three methods.

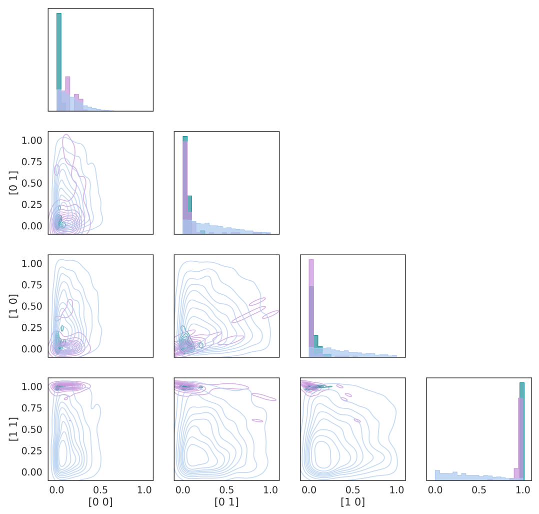

Figure 7: Contour plots to visualize the pairwise joint density for the posteriors from all methods for states in the Gridworld 2×2 environment. Here, the environment's true data generating function is $\mathbf{R} = [0, 0, 0, 1]$. The diagonal shows the marginal distribution of each state and the rest of the plots show the joint distribution between two states. The y-axis for the histograms on the diagonal corresponds to density. KD-BIRl and BIRL's contours are more concentrated than those of AVRIL.

A.8 Asymptotic expansion of the mean integrated squared error Theorem 1

Theorem A.1. *(i) The integrated squared bias of the kernel density estimator can be expanded as*

$$ISB\{\hat{f}(\cdot; \boldsymbol{H})\} = \frac{1}{4} c_2(K)^2 vec^\top \boldsymbol{R}(D^{\otimes 2} f)(vec\, \boldsymbol{H})^{\otimes 2} + o(||vec\, \boldsymbol{H}||^2). \tag{7}$$

(ii) The integrated variance of the kernel density estimator can be expanded as

$$IV\{\hat{f}(\cdot; H)\} = m^{-1} |\boldsymbol{H}|^{-1/2} R(K) + o(m^{-1} |\boldsymbol{H}|^{-1/2}). \tag{8}$$

Here, $\hat{f}$ is the estimated density function, $\mathbf{H}$ is a matrix of bandwidth values, $c_2(K) = \int_{\mathcal{R}^d} z_i^2 K(z) dz$ for all $i = 1, \ldots, d$ is the variance of the kernel function K, and m is the size of the training dataset. In our work, $\mathbf{H}$ is a diagonal matrix where every element on the diagonal is the same bandwidth h_m. In this work, we assume that f is square integrable and twice differentiable, and that the bandwidth matrix $\mathbf{H} \to 0$ as $m \to \infty$. Because we use a Gaussian kernel for K, we know that it is square integrable, spherically symmetric, and has a finite second order moment.

A.9 Wand and Jones, Equation 4.16

The mean integrated squared error (MISE) of a multivariate kernel density estimator is defined as:

$$MISE\{\hat{f}(\cdot; \mathbf{H})\} = n^{-1}(4\pi)^{\frac{-d}{2}} |\mathbf{H}|^{\frac{-1}{2}} + w^\top \{(1 - n^{-1})\Omega_2 - 2\Omega_1 + \Omega_0\} w$$

where $\mathbf{H}$ is a matrix of bandwidth values, n is the size of the dataset, Ω_a denotes the $k \times k$ matrix with (l, l') entry equal to $\phi_a \mathbf{H} + \Sigma_l + \Sigma_{l'}(\mu_l - \mu_{l'})$, ϕ_d is a d-variate Normal kernel, $w = (w_1, \ldots, w_k)^\top$ is a vector of positive numbers summing to 1, and for each $l = 1, \ldots, k$, μ_l is a $d \times 1$ vector and Σ_l is a $d \times d$ covariance matrix.

A.10 Schwartz's Theorem

If $p_0 \in KL(\Pi)$ and for every neighborhood $\mathcal{U}$ of p_0 there exist tests ϕ_n such that $P_0^n \phi_n \to 0$ and $sup_{p \in \mathcal{U}^c} P^n(1 - \phi_n) \to 0$, then the posterior distribution $\Pi_n(\cdot|X_1, \ldots, X_n)$ in the model $X_1, \ldots, X_n|p \sim^{iid} p$ and $p \sim \Pi$ is strongly consistent at p_0.

A.11 Asymptotic Statistics, Lemma 10.6

Lemma A.2. *Suppose that Θ is σ-compact, $P_\theta \neq P_{\theta'}$ for every pair $\theta \neq \theta'$, and the maps $\theta \to P_\theta$ are continuous for the total variation norm. Then there exists a sequence of estimators that is uniformly consistent on every compact subset of Θ.*

Here, Θ is the space of parameters, P is the probability density function, and $\theta \in \Theta$ is a parameter.