Inverse Reinforcement Learning via Deep Gaussian Process

Ming Jin[*]
UC Berkeley, USA
jinming@berkeley.edu

Andreas Damianou[†]
Amazon.com, Cambridge, UK
damianou@amazon.com

Pieter Abbeel
UC Berkeley, USA
pabbeel@berkeley.edu

Costas Spanos
UC Berkeley, USA
spanos@berkeley.edu

Abstract

We propose a new approach to inverse reinforcement learning (IRL) based on the deep Gaussian process (deep GP) model, which is capable of learning complicated reward structures with few demonstrations. Our model stacks multiple latent GP layers to learn abstract representations of the state feature space, which is linked to the demonstrations through the Maximum Entropy learning framework. Incorporating the IRL engine into the nonlinear latent structure renders existing deep GP inference approaches intractable. To tackle this, we develop a non-standard variational approximation framework which extends previous inference schemes. This allows for approximate Bayesian treatment of the feature space and guards against overfitting. Carrying out representation and inverse reinforcement learning simultaneously within our model outperforms state-of-the-art approaches, as we demonstrate with experiments on standard benchmarks ("object world","highway driving") and a new benchmark ("binary world").

1 INTRODUCTION

The problem of inverse reinforcement learning (IRL) is to infer the latent reward function that the agent subsumes by observing its demonstrations or trajectories in the task. It has been successfully applied in scientific inquiries, e.g., animal and human behavior modeling (Ng et al., 2000; Konstantakopoulos et al., 2017; 2016), as well as practical challenges, e.g., navigation (Ratliff et al., 2006; Abbeel et al., 2008; Ziebart et al., 2008) and intelligent building controls (Barrett and Linder, 2015). By learning the reward function, which provides the most succinct and transferable definition of a task, IRL has enabled advancing the state of the art in the robotic domains (Abbeel and Ng, 2004; Kolter et al., 2007).

Previous IRL algorithms treat the underlying reward as a linear (Abbeel and Ng, 2004; Ratliff et al., 2006; Ziebart et al., 2008; Syed and Schapire, 2007; Ratliff et al., 2009) or non-parametric function (Levine et al., 2010; 2011) of the state features. Main formulations within the linearity category include maximum margin (Ratliff et al., 2006), which presupposes that the optimal reward function leads to maximal difference of expected reward between the demonstrated and random strategies, and feature expectation matching (Abbeel and Ng, 2004; Syed et al., 2008), based on the observation that it suffices to match the feature expectation of a policy to the expert in order to guarantee similar performances. The reward function can be also regarded as the parameters for the policy class, such that the likelihood of observing the demonstrations is maximized with the true reward function, e.g., the maximum entropy approach (Ziebart et al., 2008).

As the representation power is limited by the linearity assumption, nonlinear formulations (Levine et al., 2010) are proposed to learn a set of composite features based on logical conjunctions. Non-parametric methods, pioneered by (Levine et al., 2011) based on Gaussian Processes (GPs) (Rasmussen and Williams, 2006), greatly enlarge the function space of latent reward to allow for non-linearity, and have been shown to achieve the state of the art performance on benchmark tests, e.g., object world and simulated highway driving (Abbeel and Ng, 2004; Syed and Schapire, 2007; Levine et al., 2010; 2011). Nevertheless,

[*] This research is funded by the Republic of Singapore's National Research Foundation through a grant to the Berkeley Education Alliance for Research in Singapore (BEARS) for the Singapore-Berkeley Building Efficiency and Sustainability in the Tropics (SinBerBEST) Program. BEARS has been established by the University of California, Berkeley as a center for intellectual excellence in research and education in Singapore.

[†] Work done while this author was at the University of Sheffield.

the heavy reliance on predefined or handcrafted features becomes a bottleneck for the existing methods, especially when the complexity or essence of the reward can not be captured by the given features. Finding such features automatically from data would be highly desirable.

In this paper, we propose an approach which performs feature and inverse reinforcement learning simultaneously and coherently within the same model by incorporating deep learning. The success of deep learning in a wide range of domains has drawn the community's attention to its structural advantages that can improve learning in complicated scenarios, e.g., Mnih et al. (2013) recently achieved a deep reinforcement learning (RL) breakthrough. Nevertheless, most deep models require massive data to be properly trained and can become impractical for IRL. On the other hand, deep Gaussian processes (deep GPs) (Damianou and Lawrence, 2013; Damianou, 2015) not only can they learn abstract structures with *smaller* data sets, but they also retain the non-parametric properties which Levine et al. (2011) demonstrated as important for IRL.

A deep GP is a deep belief network comprising a hierarchy of latent variables with Gaussian process mappings between the layers. Analogously to how gradients are propagated through a standard neural network, deep GPs aim at propagating uncertainty through Bayesian learning of latent posteriors. This constitutes a useful property for approaches involving stochastic decision making and also guards against overfitting by allowing for noisy features. However, previous methodologies employed for approximate Bayesian learning of deep GPs (Damianou and Lawrence, 2013; Hensman and Lawrence, 2014; Bui et al., 2015; Mattos et al., 2016) fail when diverging from the simple case of fixed output data modeled through a Gaussian regression model. In particular, in the IRL setting, the reward (output) is only revealed through the demonstrations, which is guided by the policy given by the reinforcement learning (Damianou, 2015).

The main contributions of our paper are below.

- We extend the deep GP framework to the IRL domain (Fig. 1), allowing for learning latent rewards with more complex structures from limited data.
- We derive a variational lower bound on the marginal log likelihood using an innovative definition of the variational distributions. This methodological contribution enables Bayesian learning in our model and can be applied to other scenarios where the observation layer's dynamics cause similar intractabilities.
- We compare the proposed deep GP for IRL with existing approaches in benchmark tests as well as newly defined tests with more challenging rewards.

Table 1: Summary of notations.

t	time step $t \in \{1, 2, \cdots, T\}$
h	index of demonstrations $h \in \{1, \cdots, H\}$
s, a	state $s \in \mathcal{S}$, action $a \in \mathcal{A}$
γ	RL discount factor, $\gamma \in (0, 1)$
$\boldsymbol{r}$	reward vector for all states $s \in \mathcal{S}$
$r(\cdot)$	reward function for states $\mathcal{S} \mapsto \mathbb{R}$
$Q(\cdot)$	Q-value function $\mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$
$\mathcal{Q}$	variational distribution
$V(\cdot)$	state value function $\mathcal{S} \mapsto \mathbb{R}$
$\pi, \hat{\pi}, \pi^*$	policy $\mathcal{S} \mapsto \mathcal{A}$, and the corresponding estimated and optimal versions
$\mathbf{X}$	m_0-dimensional feature matrix for $\lvert\mathcal{S}\rvert = n$ discrete states, $\mathbf{X} \in \mathbb{R}^{n,m_0}$
$\mathbf{x}_i, \mathbf{x}^m$	features for ith state, $[\mathbf{X}]_{i,:} = \mathbf{x}_i$, the mth feature for all states, $[\mathbf{X}]_{:,m} = \mathbf{x}^m$
$k_{\boldsymbol{\theta}}$	covariance function parametrized by $\boldsymbol{\theta}$
$K_{\mathbf{X},\mathbf{X}}$	covariance matrix, $[K_{\mathbf{X},\mathbf{X}}]_{i,j} = k_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j)$
$\mathbf{B}$	latent state matrix with m_1-dimensional features, $\mathbf{B} \in \mathbb{R}^{n,m_1}$
$\mathbf{D}$	$\mathbf{B}$ with Gaussian noise, $\mathbf{D} \in \mathbb{R}^{n,m_1}$
$\mathbf{W}, \mathbf{Z}$	inducing inputs, $\mathbf{W} \in \mathbb{R}^{K,m_0}$, $\mathbf{Z} \in \mathbb{R}^{K,m_1}$
$\mathbf{V}, \boldsymbol{f}$	inducing outputs, $\mathbf{V} \in \mathbb{R}^{K,m_1}$, $\boldsymbol{f} \in \mathbb{R}^K$
tilde $\tilde{}$	mean of the corresponding variable distribution, e.g., $\tilde{\boldsymbol{f}}, \tilde{\mathbf{v}}, \tilde{\mathbf{D}}$

In the following, we review the problem of inverse reinforcement learning (IRL). The list of notations used throughout the paper is summarized in Table 1.

1.1 Inverse Reinforcement Learning

The Markov Decision Process (MDP) is characterized by $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, \boldsymbol{r}\}$, which represents the state space, action space, transition model, discount factor, and reward function, respectively.

Take robot navigation as an example. The goal is to travel to the goal spot while avoiding stairwells. The state describes the current location and heading. The robot can choose actions from going forward or backward, turning left or right. The transition model specifies $p(s_{t+1}|s_t, a_t)$, i.e., the probability of reaching the next state given the current state and action, which accounts for the kinematic dynamics. The reward is +1 if it achieves the goal, -1 if it ends up in the stairwell, and 0 otherwise. The discount factor, γ, is a positive number less than 1, e.g., 0.9, to discount the future rewards. The optimal policy is then given by maximizing the expected reward, i.e.,

$$\pi^* = \arg\min_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)|\pi\right]. \tag{1}$$

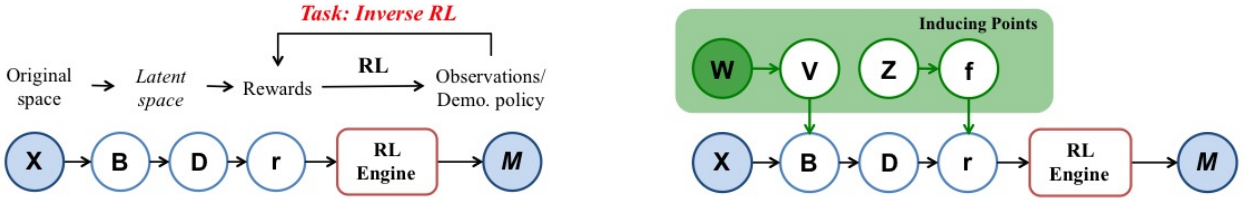


Figure 1: *Left:* The proposed deep GP model for IRL. The two layers of GPs are stacked to generate the latent reward $\boldsymbol{r}$, which is input to the reinforcement learning (RL) engine to produce an optimal control policy and demonstrations. $\mathbf{X}$ denotes the initial feature representation of states. *Right:* Illustration of DGP-IRL augmented with inducing outputs $\boldsymbol{f}, \mathbf{V}$ and corresponding inputs $\mathbf{Z}, \mathbf{W}$.

The IRL task is to find the reward function r^* such that the induced optimal policy matches the demonstrations, given $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma\}$ and $\mathcal{M} = \{\zeta_1, ..., \zeta_H\}$, where $\zeta_h = \{(s_{h,1}, a_{h,1}), ..., (s_{h,T}, a_{h,T})\}$ is the demonstration trajectory, consisting of state-action pairs. Under the **linearity assumption**, the feature representation of states forms the linear basis of reward, $r(s) = \mathbf{w}^\top \phi(s)$, where $\phi(s) : \mathcal{S} \mapsto \mathbb{R}^{m_0}$ is the m_0-dimensional mapping from the state to the feature vector. From this definition, the *expected reward* for policy π is given by:

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t)|\pi\right] = \mathbf{w}^\top \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi\right]$$

where $\mu(\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t)|\pi\right]$ is the feature expectation for policy π. The reward parameter $\mathbf{w}^*$ is learned such that

$$\mathbf{w}^{*\top} \mu(\pi^*) \geq \mathbf{w}^{*\top} \mu(\pi), \forall \pi \tag{2}$$

a prevalent idea that appears in the maximum margin planning (MMP) (Ratliff et al., 2006) and feature expectation matching (Syed and Schapire, 2007).

Motivated by the perspective of expected reward that parametrizes the policy class, the maximum entropy (MaxEnt) model (Ziebart et al., 2008) considers a stochastic decision model, where *the optimal policy randomly chooses the action* according to the associated reward:

$$p(a|s) = \exp\{Q^*(s, a; \boldsymbol{r}) - V^*(s; \boldsymbol{r})\} \tag{3}$$

where $V(s; \boldsymbol{r}) = \log \sum_a \exp(Q(s, a; \boldsymbol{r}))$ follows the Bellman equation, $Q(s, a; \boldsymbol{r})$ and $V(s; \boldsymbol{r})$ are measures of how desirable are the corresponding state s and state-action pair (s, a) under rewards $\boldsymbol{r}$. In principle, for a given state s, the best action corresponds to the highest Q-value, which represents the "desirability" of the action. Assuming independence among state-action pairs from demonstrations, the likelihood of the demonstration corresponds to the joint probability of taking a sequence of actions $a_{i,t}$ under states $s_{h,t}$ according to the Bellman equation:

$$\begin{aligned} p(\mathcal{M}|\boldsymbol{r}) &= \prod_{h=1}^{H} \prod_{t=1}^{T} p(a_{h,t}|s_{h,t}) \\ &= \exp\left(\sum_{h=1}^{H} \sum_{t=1}^{T} \left(Q(s_{h,t}, a_{h,t}; \boldsymbol{r}) - V(s_{h,t}; \boldsymbol{r})\right)\right). \end{aligned} \tag{4}$$

Though directly optimizing the above criteria with respect to $\boldsymbol{r}$ is possible, it does not lead to generalized solutions transferrable in a new test case where no demonstrations are available; hence, we need a "model" of $\boldsymbol{r}$. MaxEnt assumes linear structure for rewards, while GPIRL (Levine et al., 2011) uses GPs to relate the states to rewards. In Section 2.1, we will give a brief overview of GPs and GPIRL.

2 THE MODEL

In this section, we start by discussing the reward modeling through Gaussian processes (GPs) following (Levine et al., 2011), proceed to incorporate the representation learning module as additional GP layers, then develop our variational training framework and, finally, develop the transfer between tasks.

2.1 Gaussian Process Reward Modeling

We consider the setup of *discretizing the world* into n states. Assume that observed state-action pairs (demonstrations) $\mathcal{M} = \{\zeta_1, \ldots, \zeta_h\}$ are *generated* by a set of m_0-dimensional state features $\mathbf{X} \in \mathbb{R}^{n, m_0}$ through the reward function r. Throughout this paper we denote points (rows of a matrix) as $[\mathbf{X}]_{i,:} = \mathbf{x}_i$, features (columns) as $[\mathbf{X}]_{:,m} = \mathbf{x}^m$ and single elements as $[\mathbf{X}]_{i,m} = x_i^m$.

In this modeling framework, the reward function r plays the role of an unknown mapping, thus we wish to treat it as *latent* and keep it flexible and non-linear. Therefore, we follow (Levine et al., 2011) and model it with a zero-mean GP prior (Rasmussen and Williams, 2006):

$$r \sim \mathcal{GP}(0, k_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j)),$$

where $k_{\boldsymbol{\theta}}$ denotes the covariance function, e.g. $k_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_k^2 e^{-\frac{\xi}{2}(\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{x}_i - \mathbf{x}_j)}$, $\boldsymbol{\theta} = \{\sigma_k, \xi\}$. Given a finite amount of data, this induces the probability $\boldsymbol{r}|\mathbf{X}, \boldsymbol{\theta} \sim \mathcal{N}(\mathbf{0}, K_{\mathbf{XX}})$, where $\boldsymbol{r} \triangleq r(\mathbf{X})$ and the covariance matrix is obtained by $[K_{\mathbf{XX}}]_{i,j} = k_{\boldsymbol{\theta}}(\mathbf{x}_i, \mathbf{x}_j)$. The GPIRL training objective comes from integrating out the latent reward:

$$p(\mathcal{M}|\mathbf{X}) = \int p(\mathcal{M}|\boldsymbol{r}) p(\boldsymbol{r}|\mathbf{X}, \boldsymbol{\theta}) \mathrm{d}\boldsymbol{r} \tag{5}$$

and maximizing over $\boldsymbol{\theta}$, which we drop from our expressions from now on.

The above integral is intractable, because $p(\mathcal{M}|\boldsymbol{r})$ has the complicated expression of (4) (this is in contrast to the tranditional GP regression where $\mathcal{M}|\boldsymbol{r}$ is a Gaussian or other simple distribution). This can be alleviated using the approximation of (Levine et al., 2011). We will describe this approximation in the next section, as it is also used by our approach. Notice that all latent function instantiations are linked through a joint multivariate Gaussian. Thus, prediction of the function value $r^* = r(\mathbf{x}^*)$ at a test input $\mathbf{x}^*$ is found through the conditional

$$r^*|\boldsymbol{r}, \mathbf{X}, \mathbf{x}^* \sim \mathcal{N}\big(K_{\mathbf{x}^*\mathbf{X}} K_{\mathbf{XX}}^{-1} \boldsymbol{r}, k_{\mathbf{x}^*\mathbf{x}^*} - K_{\mathbf{x}^*\mathbf{X}} K_{\mathbf{XX}}^{-1} K_{\mathbf{X}\mathbf{x}^*}\big)$$

As can be seen, the prediction $r(\mathbf{x}^*)$ is reliant on the *effectiveness of feature representation*: states with features close in Euclidean distance are assumed to be associated with similar rewards. This motivates our novel deep GP-IRL method which is obtained by considering additional layers, as we will describe next.

2.2 Incorporating the Representation Learning Layers

The traditional model-based IRL approach is to learn the latent reward $\boldsymbol{r}$ (operating on fixed state features $\mathbf{X}$) that best explains the demonstrations $\mathcal{M}$. In this paper we wish to additionally and simultaneously uncover a highly descriptive state feature representation. To achieve this, we introduce a *latent* state feature representation $\mathbf{B} = \big[\mathbf{b}^1, ..., \mathbf{b}^{m_1}\big] \in \mathbb{R}^{n, m_1}$. $\mathbf{B}$ constitutes the instantiations of an introduced function b which is learned as a non-linear GP transformation from $\mathbf{X}$. To account for noise we further introduce $\mathbf{D}$ as the noisy versions of $\mathbf{B}$, i.e., $d_i^m = b_i^m + \epsilon$, $\epsilon \sim \mathcal{N}(0, \lambda^{-1})$.

Importantly, rather than performing two steps of learning separately (for the GPs on r and on b), we nest them into a single objective function, to maintain the flow of information during optimization. This results in a deep GP whose top layers perform representation learning and lower layers perform model-based IRL. Fig. 1 outlines our model, Deep Gaussian Process for Inverse Reinforcement Learning (**DGP-IRL**). By using $\mathbf{x}^m$ to represent the m-th column of $\mathbf{X}$, and similarly for $\mathbf{D}, \mathbf{B}$ the full generative model is written as follows:

$$\begin{aligned}
&p(\mathcal{M}, \boldsymbol{r}, \mathbf{D}, \mathbf{B}|\mathbf{X}) \\
&= \underbrace{p(\mathcal{M}|\boldsymbol{r})}_{\text{IRL}} \underbrace{p(\boldsymbol{r}|\mathbf{D})}_{\mathcal{GP}(\mathbf{0}, k^r(\mathbf{d}_i, \mathbf{d}_j))} \underbrace{p(\mathbf{D}|\mathbf{B})}_{\text{Gaussian noise}} \underbrace{p(\mathbf{B}|\mathbf{X})}_{\mathcal{GP}(\mathbf{0}, k^b(\mathbf{x}_i, \mathbf{x}_j))} \\
&= e^{\sum_{h=1}^H \sum_{t=1}^T \left(Q(s_{h,t}, a_{h,t}; \boldsymbol{r}) - V(s_{h,t}; \boldsymbol{r})\right)} \mathcal{N}(\boldsymbol{r}|\mathbf{0}, K_{\mathbf{DD}}) \\
&\qquad \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{d}^m|\mathbf{b}^m, \lambda^{-1}\mathbf{I}) \mathcal{N}(\mathbf{b}^m|\mathbf{0}, K_{\mathbf{XX}}),
\end{aligned} \tag{6}$$

where the IRL term $p(\mathcal{M}|\boldsymbol{r})$ takes the form of (4) as suggested by Ziebart et al. (2008). $K_{\mathbf{XX}}$ and $K_{\mathbf{DD}}$ are the covariance matrices in each layer, constructed with covariance functions k^b and k^r respectively. Compared to GPIRL the proposed framework has substantial gain in flexibility by introducing the abstract representation of states in the hidden layers $\mathbf{B}, \mathbf{D}$. Note that the model in Fig. 1 can be extended in depth by introducing additional hidden layers and connecting them with additional GP mappings; it is only for illustration simplicity that we base our derivation on the two layered structure.

We can compress the statistical power of each generative layer into a set of auxiliary variables within a *sparse GP framework* (Snelson and Ghahramani, 2005). Specifically, we introduce *inducing outputs and inputs*, denoted by $\boldsymbol{f} \in \mathbb{R}^K$ and $\mathbf{Z} \in \mathbb{R}^{K, m_1}$ respectively for the lower layer and by $\mathbf{V} \in \mathbb{R}^{K, m_1}$ and $\mathbf{W} \in \mathbb{R}^{K, m_0}$ for the top layer (as in Fig. 1). The inducing outputs and inputs are related with the same GP prior appearing in each layer. For example, $\boldsymbol{f}|\mathbf{Z} \sim \mathcal{N}(\mathbf{0}, K_{\mathbf{ZZ}})$ with $K_{\mathbf{ZZ}} = k^r(\mathbf{Z}, \mathbf{Z})$. By relating the original and inducing variables through the conditional Gaussian distribution, the auxiliary variables are learned to be *sufficient statistics* of the GP. The augmented model, shown in Fig. 1, has the following definition:

$$\begin{aligned}
&p(\mathcal{M}, \boldsymbol{r}, \boldsymbol{f}, \mathbf{B}, \mathbf{D}, \mathbf{V}|\mathbf{X}, \mathbf{Z}, \mathbf{W}) \\
&= p(\mathcal{M}|\boldsymbol{r}) p(\boldsymbol{r}|\boldsymbol{f}, \mathbf{D}, \mathbf{Z}) p(\boldsymbol{f}|\mathbf{Z}) p(\mathbf{D}|\mathbf{B}) p(\mathbf{B}|\mathbf{V}, \mathbf{X}, \mathbf{W}) \\
&= e^{\sum_{h=1}^H \sum_{t=1}^T \left(Q(s_{h,t}, a_{h,t}; \boldsymbol{r}) - V(s_{h,t}; \boldsymbol{r})\right)} \cdot \\
&\quad \mathcal{N}(\boldsymbol{r}|K_{\mathbf{DZ}} K_{\mathbf{ZZ}}^{-1} \boldsymbol{f}, \boldsymbol{\Sigma}_r) \mathcal{N}(\boldsymbol{f}|\mathbf{0}, K_{\mathbf{ZZ}}) \cdot \\
&\quad \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{d}^m|\mathbf{b}^m, \lambda^{-1}\mathbf{I}) \mathcal{N}(\mathbf{b}^m|K_{\mathbf{XW}} K_{\mathbf{WW}}^{-1} \mathbf{v}^m, \boldsymbol{\Sigma}_B)
\end{aligned} \tag{7}$$

where we adopt the Fully Independent Training Conditional (FITC) to preserve the exact variances in $\mathbf{\Sigma}_B = diag\left(K_{\mathbf{XX}} - K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}\right)$, and the Deterministic Training Conditional (DTC) (Quiñonero-Candela and Rasmussen, 2005) in $\mathbf{\Sigma}_r = \mathbf{0}$ as in GPIRL to facilitate the integration of $\boldsymbol{r}$ in the training objective (see next section).

In the following, we will omit the inducing inputs $\mathbf{W}, \mathbf{Z}$ in the conditions, with the convention to treat them as model parameters (Damianou and Lawrence, 2013; Damianou, 2015; Kandemir, 2015). By selecting $K \ll n$ the complexity reduces from $\mathcal{O}(n^3)$ to $\mathcal{O}(nK^2)$. While DGP-IRL resolves the case when the outputs have complex dependencies with the latent layers, the training of the model based on variational inference requires gradients for the parameters, as in backpropagation, whose convergence can be improved by leveraging advancements in deep learning.

Additionally, in DGP-IRL, the role of auxiliary variables goes further than just introducing scalability. Indeed, as we shall see next, the auxiliary variables play a distinct role in our model, by forming the base of a variational framework for Bayesian inference.

2.3 Variational Inference

We wish to optimize the model evidence for training:

$$p(\mathcal{M}|\mathbf{X}) = \int p(\mathcal{M}, \boldsymbol{f}, \boldsymbol{r}, \mathbf{V}, \mathbf{D}, \mathbf{B}|\mathbf{X})\mathrm{d}(\boldsymbol{f}, \boldsymbol{r}, \mathbf{V}, \mathbf{D}, \mathbf{B})$$

However, this quantity is intractable. Firstly because the latent variables $\mathbf{D}$ appear nonlinearly in the inverse of covariance matrices. Secondly, because the latent rewards $\boldsymbol{f}, \boldsymbol{r}$ relate to the observation $\mathcal{M}$ through the reinforcement learning layer; the choice of $\mathbf{\Sigma}_r = \mathbf{0}$ in (7) does not completely solve this problem because in DGP-IRL there is additional uncertainty propagated by the latent layers. This indicates that Laplace approximation is not practical, neither is the variational method employed for deep GP (Damianou and Lawrence, 2013; Kandemir, 2015), where the output is related to the latent variable in a simple regression framework.

To this end, we show that we can derive an analytic lower bound on the model evidence by constructing a variational framework using the following special form of variational distribution:

$$\mathcal{Q} = q(\boldsymbol{f})q(\mathbf{D})q(\mathbf{B})q(\mathbf{V}), \text{ with :} \tag{8}$$

$$q(\boldsymbol{f}) = \delta(\boldsymbol{f} - \tilde{\boldsymbol{f}})$$

$$q(\mathbf{B}) = p(\mathbf{B}|\mathbf{V}, \mathbf{X}) \tag{9}$$

$$q(\mathbf{D}) = \prod_{m=1}^{m_1} \delta\left(\mathbf{d}^m - K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\tilde{\mathbf{v}}^m\right) \tag{10}$$

$$q(\mathbf{V}) = \prod_{m=1}^{m_1} \mathcal{N}\left(\mathbf{v}^m|\tilde{\mathbf{v}}^m, \mathbf{G}^m\right) \tag{11}$$

The delta distribution is equivalent to taking the mean of normal distributions for prediction, which is reasonable in the context of reinforcement learning (Levine et al., 2011). Also note that the delta distribution is *applied only in the bottom layer and not repeatedly*; therefore, representation learning is indeed being manifested in the latent layers.

In addition, $q(\mathbf{B})$ matches the exact conditional $p(\mathbf{B}|\mathbf{V}, \mathbf{X})$ so that these two terms cancel in the fraction of (13) and the number of variational parameters is minimized, as in (Titsias and Lawrence, 2010). As for $q(\mathbf{D})$, it is chosen as delta distributions such that combined with $\mathbf{\Sigma}_r = \mathbf{0}$ the IRL term $p(\mathcal{M}|\boldsymbol{r})p(\boldsymbol{r}|\boldsymbol{f}, \mathbf{D})$ in (12) becomes tractable and information can flow through the latent layers $\mathbf{B}, \mathbf{D}$.

The variational marginal $q(\mathbf{V})$ is factorized across its dimensions with fully parameterized normal densities, as in (Damianou and Lawrence, 2013). Notice that $\tilde{\boldsymbol{f}}$ and $\tilde{\mathbf{v}}$ are the mean of the inducing outputs (Table 1), corresponding to pseudo-inputs $\mathbf{Z}$ and $\mathbf{W}$, where $\mathbf{Z}$ (initialized with random numbers from uniform distributions (Sutskever et al., 2013)) be learned to further maximize the marginaliked likelihood, and $\mathbf{W}$ is chosen as a subset of $\mathbf{X}$.

The variational means of $\mathbf{D}$ can be augmented with input data, $\mathbf{X}$, to improve stability during training (Duvenaud et al., 2014).

The variational lower bound, $\mathcal{L}$, follows from the Jensen's inequality, and can be derived analytically due to the choice of variational distribution $\mathcal{Q}$ (see Section 2 of the Appendix for details):

$$\log p(\mathcal{M}|\mathbf{X}) = \log \int \underbrace{p(\mathcal{M}|\boldsymbol{r})p(\boldsymbol{r}|\boldsymbol{f}, \mathbf{D})}_{p(\mathcal{M}|\boldsymbol{r}=K_{\mathbf{DZ}}K_{\mathbf{ZZ}}^{-1}\boldsymbol{f}) \text{ by DTC: } \mathbf{\Sigma}_r = \mathbf{0}} p(\boldsymbol{f})p(\mathbf{D}|\mathbf{B})p(\mathbf{B}|\mathbf{V}, \mathbf{X})p(\mathbf{V})\mathrm{d}(\boldsymbol{r}, \boldsymbol{f}, \mathbf{V}, \mathbf{D}, \mathbf{B}) \tag{12}$$

$$\geq \int q(\boldsymbol{f})q(\mathbf{D})p(\mathbf{B}|\mathbf{V}, \mathbf{X})q(\mathbf{V}) \log \frac{p(\mathcal{M}|K_{\mathbf{DZ}}K_{\mathbf{ZZ}}^{-1}\boldsymbol{f})p(\boldsymbol{f})p(\mathbf{D}|\mathbf{B})p(\mathbf{V})}{q(\boldsymbol{f})q(\mathbf{D})q(\mathbf{V})} \text{ by Jensen's ineq.} \tag{13}$$

$$= \mathcal{L}_{\mathcal{M}} + \mathcal{L}_{\mathcal{G}} - \mathcal{L}_{\mathrm{KL}} + \mathcal{L}_{\mathcal{B}} - \frac{nm_1}{2}\log(2\pi\lambda^{-1}) \tag{14}$$

where

$$\mathcal{L}_{\mathcal{M}} = \log p(\mathcal{M}|K_{\tilde{\mathbf{D}}\mathbf{Z}}K_{\mathbf{Z}\mathbf{Z}}^{-1}\tilde{\boldsymbol{f}})) = \sum_{h=1}^{H}\sum_{t=1}^{T}\Big(Q(s_{h,t},a_{h,t};K_{\tilde{\mathbf{D}}\mathbf{Z}}K_{\mathbf{Z}\mathbf{Z}}^{-1}\tilde{\boldsymbol{f}}) - V(s_{h,t};K_{\tilde{\mathbf{D}}\mathbf{Z}}K_{\mathbf{Z}\mathbf{Z}}^{-1}\tilde{\boldsymbol{f}})\Big) \tag{15}$$

$$\mathcal{L}_{\mathcal{G}} = \log p(\boldsymbol{f} = \tilde{\boldsymbol{f}}|\mathbf{Z}) = \log \mathcal{N}(\boldsymbol{f} = \tilde{\boldsymbol{f}}|\mathbf{0}, K_{\mathbf{Z}\mathbf{Z}}) \tag{16}$$

$$\mathcal{L}_{KL} = \mathrm{KL}(q(\mathbf{V})||p(\mathbf{V}|\mathbf{W})) = \sum_{m=1}^{m_1}\mathrm{KL}(\mathcal{N}(\mathbf{v}^m|\tilde{\mathbf{v}}^m,\mathbf{G}^m)||\mathcal{N}(\mathbf{v}^m|\mathbf{0},K_{\mathbf{W}\mathbf{W}}))$$

$$\mathcal{L}_{\mathcal{B}} = -\frac{\lambda}{2}\sum_{m=1}^{m_1}\mathrm{Tr}(\boldsymbol{\Sigma}_{\mathbf{B}} + K_{\mathbf{X}\mathbf{W}}K_{\mathbf{W}\mathbf{W}}^{-1}\mathbf{G}^m K_{\mathbf{W}\mathbf{W}}^{-1}K_{\mathbf{W}\mathbf{X}}) \tag{17}$$

where $|K_{\mathbf{W}\mathbf{W}}|$ is the determinant of $K_{\mathbf{W}\mathbf{W}}$. $\tilde{\mathbf{D}} = [\tilde{\mathbf{d}}^1, ..., \tilde{\mathbf{d}}^{m_1}]$, where $\tilde{\mathbf{d}}^m = K_{\mathbf{X}\mathbf{W}}K_{\mathbf{W}\mathbf{W}}^{-1}\tilde{\mathbf{v}}^m$. $\mathcal{L}_{\mathcal{M}}$ is the term associated with RL. $\mathcal{L}_{\mathcal{G}}$ is the Gaussian prior on inducing outputs $\boldsymbol{f}$. $\mathcal{L}_{KL}$ denotes the Kullback – Leibler (KL) divergence between the variational posterior $q(\mathbf{V})$ to the prior $p(\mathbf{V})$, acting as a regularization term. The lower bound $\mathcal{L}$ can be optimized with gradient-based methods, which are computed by backpropagation. In addition, we can find the optimal fixed-point equations for the variational distribution parameters $\tilde{\mathbf{v}}^m, \mathbf{G}^m$ for $q(\mathbf{V})$ using variational calculus, in order to raise the variational lower bound $\mathcal{L}$ further (refer to Section 3 of the supplement for this derivation).

Notice that the approximate marginalization of all hidden spaces, in (14), approximates a Bayesian training procedure, according to which model complexity is automatically balanced through the Bayesian Occam's razor principle. Optimizing the objective $\mathcal{L}$ turns the variational distribution $\mathcal{Q}$ into an approximation to the true model posterior.

2.4 Transfer to New Tasks

The inducing points provide a succinct summary of the data, by the property of FITC (Quiñonero-Candela and Rasmussen, 2005), which means only the inducing points are necessary for prediction. Given a set of new states $\mathbf{X}^*$, DGP-IRL can infer the latent reward through the full Bayesian treatment:

$$p(\boldsymbol{r}^*|\mathbf{X}^*,\mathbf{X}) = \int \Big\{p(\boldsymbol{r}^*|\boldsymbol{f},\mathbf{D}^*)q(\boldsymbol{f})p(\mathbf{D}^*|\mathbf{B}^*) \; p(\mathbf{B}^*|\mathbf{V},\mathbf{X}^*)q(\mathbf{V})\Big\}\mathrm{d}(\boldsymbol{f},\mathbf{B}^*,\mathbf{D}^*,\mathbf{V}) \tag{18}$$

Given that the above integral is computationally intensive to evaluate, a practical alternative adopted in our implementation is to use point estimates for latent variables; hence, the rewards are given by:

$$\boldsymbol{r}^* = K_{\mathbf{D}^*\mathbf{Z}}K_{\mathbf{Z}\mathbf{Z}}^{-1}\tilde{\boldsymbol{f}} \tag{19}$$

where $\mathbf{D}^* = [\mathbf{d}_*^1, ..., \mathbf{d}_*^{m_1}]$, with $\mathbf{d}_*^m = K_{\mathbf{X}^*\mathbf{W}}K_{\mathbf{W}\mathbf{W}}^{-1}\tilde{\mathbf{v}}^m$. The above formulae suggest that instead of making inference based on $\mathbf{X}$ layer directly as in Levine et al. (2011), DGP-IRL first estimates the latent representation of the states, $\mathbf{D}^*$, then makes GP regression using the latent variables.

3 EXPERIMENTS

For the experimental evaluation, we employ the *expected value difference* (EVD) as a metric of optimality, given by:

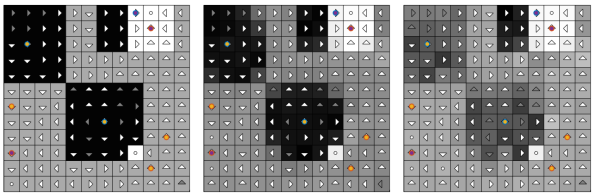
$$\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t)|\pi^*\right] - \mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t)|\hat{\pi}\right], \tag{20}$$

which is the difference between the expected reward earned under the optimal policy, π^*, given by the true rewards, and the policy derived from the IRL rewards, $\hat{\pi}$. Our software implementation is included in the supplementary.

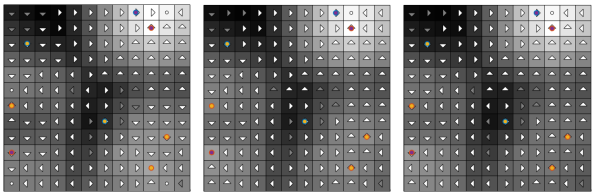
3.1 Object World Benchmark

The Object World (OW) benchmark, originally implemented by Levine et al. (2011), is a $N \times N$ gridworld where dots of *primary colors*, e.g., red and blue, and *secondary colors*, e.g., purple and green, are placed on the grid at random, as shown in Fig. 2. Each state, i.e., grid block, is described by the shortest distances to dots among each color group. The latent reward is assigned such that if a block is 1 step within a red dot and 3 steps within a blue dot, the reward is +1; if it is 3 steps within a blue dot only, the reward is -1, and the reward is 0 otherwise. The agent maximizes its expected discounted reward by following a policy which provides the probabilities of actions (moving up/down/left/right, or stay still) at each state, subject to a transition probability.

The objective of the experiment is to compare the performances of DGP-IRL with previous methods as the



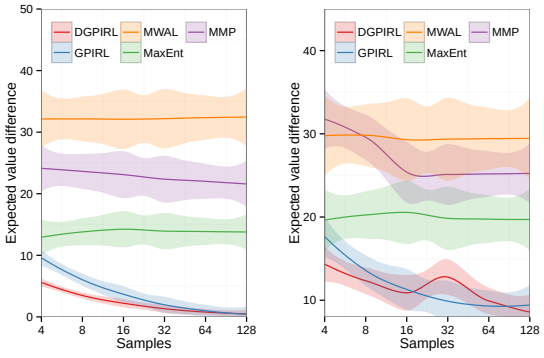
(a) Ground Truth (b) DGP-IRL (c) GPIRL



(d) MWAL (e) MaxEnt (f) MMP

Figure 2: OW benchmark for IRL, evaluated for (a) DGP-IRL, (b) GPIRL, (c) MWAL, (d) MaxEnt, and (e) MMP, with 64 demonstrations and continuous features. Except for DGP-IRL, all the other algorithms are evaluated with the toolbox by Levine et al. (2011).

number of demonstrations varies. Candidates that are evaluated include the Multiplicative Weights for Apprenticeship Learning (MWAL) (Syed and Schapire, 2007), MaxEnt, MMP, which assume a linear reward function, and GPIRL, which is the state-of-the-art method on the benchmark. Linear models, as is shown in Fig. 2, cannot capture the complex structure, while GPIRL learns more accurate yet still noisy rewards, as limited by feature discriminability; DGP-IRL, on the contrary, makes inference closest to the ground truth even with limited data, thanks to the increased representational power and robust training through variational inference.



(a) Training (b) Transfer

Figure 3: Plots of EVD in the training (a) and transfer (b) tests for the OW benchmark, where continuous features are employed.

Additionally, the transferability test is carried out by examining EVD in a new world where no demonstrations are available, which requires the ability of knowledge transfer from the previous learning scenario. DGP-IRL outperforms GPIRL and other models in both the training and transfer cases, and the improvement is obvious as more data is accessible (Fig. 3). The shaded area (Fig. 3, 6, 7(a)) are the standard deviation of EVD among independent experiments, which reflect that DGP-IRL and GPIRL are more reliable.

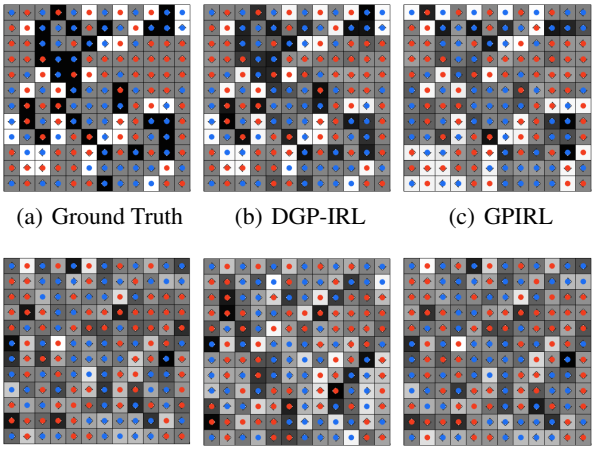
3.2 Binary World Benchmark

Though the rewards in OW are nonlinear functions of the features, they form separated clusters in the subspace spanned by two dimensions, i.e., distances to the nearest red and blue dots. Binary world (BW) is a benchmark introduced by Wulfmeier et al. (2015) whose reward depends on combinatorics of features. More specifically, in a world of $N \times N$ plane where each block is randomly assigned with either a blue or red dot, the state is associated with the +1 reward if there are 4 blues in the 3×3 *neighborhood*, -1 for 5 blues, and 0 otherwise. The feature represents the color of the 9 dots in the neighborhood. BW sets up a challenging scenario, where states that are maximally separated in feature space can have the same rewards, yet those that are close in euclidean distance may have opposite rewards.

The task is to learn the latent rewards of states given limited demonstrations, as is shown in Fig. 4. While linear models are limited by their capacity of representation, the results of GPIRL also deviate from the latent rewards as it cannot generalize from training data with the convoluted features. DGP-IRL, nevertheless, is able to recover the ground truth with the highest fidelity.

By successively warping the original feature space through the latent layers, DGP-IRL can learn an abstract representation that reveals the reward structure. As is illustrated in Fig. 5, though the points are mixed up in the input space, making it impossible to separate those with the same rewards, their positions in the latent space clearly form clusters, which indicates that DGP-IRL has remarkably uncovered the mechanism of reward generation by simply observing the traces of actions.

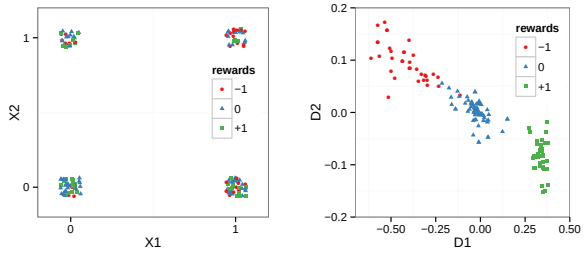
The advantage of simultaneous representation and inverse reinforcement learning is demonstrated in Fig. 6, which plots the EVD for the training and transfer cases. As the features are interlinked not only with the reward but also with themselves in a very nonlinear way, this scenario is particularly challenging for linear models, such as LEARCH (Ratliff et al., 2009), MaxEnt, and MMP. While both GPIRL and DGP-IRL have satisfactory performance in the training case, DGP-IRL significantly outperforms GPIRL in the transfer task, which indicates that



(a) Ground Truth (b) DGP-IRL (c) GPIRL

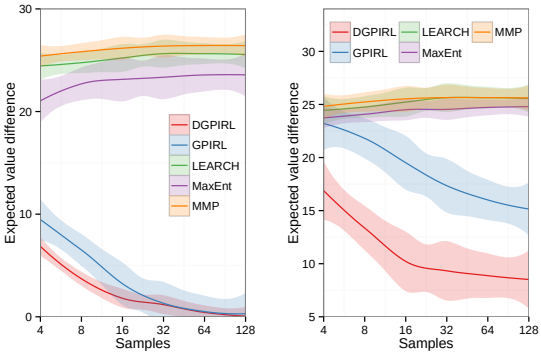
(d) LEARCH (e) MaxEnt (f) MMP

Figure 4: BW benchmark evaluated with 128 demonstrated traces for DGP-IRL, GPIRL, LEARCH Ratliff et al. (2009), MaxEnt, and MMP.



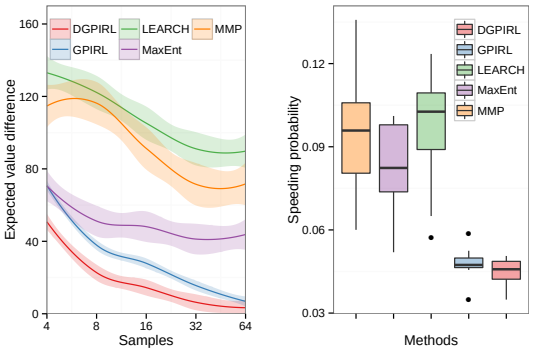
(a) Input space $\mathbf{X}$ (b) Latent space $\mathbf{D}$

Figure 5: Visualization of points along two arbitrary dimensions in the (a) input space $\mathbf{X}$ and (b) latent space $\mathbf{D}$ of DGP-IRL. The points represent features of states and are color-coded with the associated rewards. The rewarads are entangled in the input space $\mathbf{X}$ but separated in the latent space $\mathbf{D}$.



(a) Training (b) Transfer

Figure 6: Plots of EVD in the training (a) and transfer (b) tests for the BW benchmark as the number of training samples varies.



(a) Training EVD (b) Speeding Prob.

Figure 7: Plots of EVD in the training (a) and the probability of speeding (with 64 demonstrations) (b) in the highway driving simulation benchmark, with three lanes and 32 car lengths.

the learned latent space is transferrable across different scenarios.

3.3 Highway Driving Behaviors

Highway driving behavior modeling, as investigated by Levine et al. (2010; 2011), is a concrete example to examine the capacity of IRL algorithms in learning the underlying motives from demonstrations based on a simple simulator. In a three-lane highway, vehicles of specific class (civilian or police) and category (car or motorcycle) are positioned at random, driving at the same constant speed. The robot car can switch lanes and navigate at up to three times the traffic speed. The state is described by a continuous feature which consists of the closest distances to vehicles of each class and category in the same lane, together with the left, right, and any lane, both in the front and back of the robot car, in addition to the current speed and position.

The goal is to navigate the robot car as fast as possible, but avoid speeding by checking that the speed is no greater than twice the current traffic when the police car is 2 car lengths nearby. As the reward is a nonlinear function determined by the current speed and distance to the police, linear models are outrun by GPIRL and DGP-IRL. Performance generally improves with more demonstrations, and DGP-IRL remains to yield the policy closest to the optimal in EVD, and with minimal probability of speeding, as illustrated in Fig. 7(a) and 7(b), respectively.

4 CONCLUSION AND FUTURE WORK

DGP-IRL is proposed as a solution for IRL based on deep GPs. By extending the structure with additional GP layers to learn the abstract representation of the state features, DGP-IRL has more representational capability than linear-based and GP-based methods. Meanwhile, the Bayesian training approach through variational inference guards against overfitting, bringing the advantage of automatic capacity control and principled uncertainty handling.

Our proposed DGP-IRL outperforms state-of-the-art methods in our experiments, and is shown to learn efficiently even from few demonstrations. For future work, the unique properties of DGP-IRL enable easy incorporation of side knowledge (through priors on the latent space) to IRL, but our work also opens up the way for combining deep GPs with other complicated inference engines, e.g., selective attention models (Gregor et al., 2015). We plan to investigate these ideas in the future. Another promising future direction is to construct transferable models where the latent layer is relied on for knowledge sharing. Finally, we plan to investigate some of the many applications where DGP-IRL can prove especially beneficial, such as intelligent transportation (Gu et al., 2016), building and grid controls (Jin et al., 2017a;b), multiobjective control (Jia et al., 2017), and human-in-the-loop gamification (Ratliff et al., 2014).

References

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, page 1. ACM, 2004.

Pieter Abbeel, Dmitri Dolgov, Andrew Y Ng, and Sebastian Thrun. Apprenticeship learning for motion planning with application to parking lot navigation. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 1083–1090. IEEE, 2008.

Enda Barrett and Stephen Linder. Autonomous HVAC control, a reinforcement learning approach. In *Machine Learning and Knowledge Discovery in Databases*, pages 3–19. Springer, 2015.

Thang D Bui, José Miguel Hernández-Lobato, Yingzhen Li, Daniel Hernández-Lobato, and Richard E Turner. Training deep Gaussian processes using stochastic expectation propagation and probabilistic backpropagation. *arXiv preprint arXiv:1511.03405*, 2015.

Andreas Damianou. *Deep Gaussian processes and variational propagation of uncertainty*. PhD thesis, University of Sheffield, 2015.

Andreas Damianou and Neil Lawrence. Deep Gaussian processes. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pages 207–215, 2013.

David Duvenaud, Oren Rippel, Ryan P. Adams, and Zoubin Ghahramani. Avoiding pathologies in very deep networks. In *Artificial Intelligence and Statistics*, 2014.

Karol Gregor, Ivo Danihelka, Alex Graves, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

Weixi Gu, Ming Jin, Zimu Zhou, Costas J Spanos, and Lin Zhang. Metroeye: Smart tracking your metro trips underground. In *Proceedings of the 13th International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, pages 84–93. ACM, 2016.

James Hensman and Neil D Lawrence. Nested variational compression in Deep Gaussian Processes. *arXiv preprint arXiv:1412.1370*, 2014.

Ruoxi Jia, Roy Dong, S Shankar Sastry, and Costas J Spanos. Privacy-enhanced architecture for occupancy-based HVAC control. In *Proceedings of the 8th International Conference on Cyber-Physical Systems*, pages 177–186. ACM, 2017.

Ming Jin, Wei Feng, Ping Liu, Chris Marnay, and Costas Spanos. Mod-dr: Microgrid optimal dispatch with demand response. *Applied Energy*, 187:758–776, 2017a.

Ming Jin, Ruoxi Jia, and Costas Spanos. Virtual occupancy sensing: Using smart meters to indicate your presence. *IEEE Transactions on Mobile Computing*, 99:1, 2017b.

Melih Kandemir. Asymmetric transfer learning with Deep Gaussian Processes. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 730–738, 2015.

J Zico Kolter, Pieter Abbeel, and Andrew Y Ng. Hierarchical apprenticeship learning with application to quadruped locomotion. In *Advances in Neural Information Processing Systems*, pages 769–776, 2007.

I. C. Konstantakopoulos, L. J. Ratliff, M. Jin, S. S. Sastry, and C. J. Spanos. A robust utility learning framework via inverse optimization. *IEEE Transactions on Control Systems Technology*, PP(99):1–17, 2017. doi: 10.1109/TCST.2017.2699163.

Ioannis C Konstantakopoulos, Lillian J Ratliff, Ming Jin, Costas J Spanos, and S Shankar Sastry. Inverse modeling of non-cooperative agents via mixture of utilities. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 6327–6334. IEEE, 2016.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Feature construction for inverse reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1342–1350, 2010.

Sergey Levine, Zoran Popovic, and Vladlen Koltun. Nonlinear inverse reinforcement learning with Gaussian processes. In *Advances in Neural Information Processing Systems*, pages 19–27, 2011.

César Lincoln C Mattos, Zhenwen Dai, Andreas Damianou, Jeremy Forth, Guilherme A Barreto, and Neil D Lawrence. Recurrent Gaussian processes. *International Conference on Learning Representations (ICLR)*, 2016.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, pages 663–670, 2000.

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005.

Carl Edward Rasmussen and Christopher KI Williams. *Gaussian processes for machine learning*, volume 1. MIT press Cambridge, 2006.

Lillian J Ratliff, Ming Jin, Ioannis C Konstantakopoulos, Costas Spanos, and S Shankar Sastry. Social game for building energy efficiency: Incentive design. In *52nd Annual Allerton Conference on Communication, Control, and Computing, 2014*, pages 1011–1018, 2014.

Nathan D Ratliff, J Andrew Bagnell, and Martin A Zinkevich. Maximum margin planning. In *Proceedings of the 23rd international conference on Machine learning*, pages 729–736. ACM, 2006.

Nathan D Ratliff, David Silver, and J Andrew Bagnell. Learning to search: Functional gradient techniques for imitation learning. *Autonomous Robots*, 27(1):25–53, 2009.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2005.

Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1139–1147, 2013.

Umar Syed and Robert E Schapire. A game-theoretic approach to apprenticeship learning. In *Advances in neural information processing systems*, volume 20, pages 1–8, 2007.

Umar Syed, Michael Bowling, and Robert E Schapire. Apprenticeship learning using linear programming. In *Proceedings of the 25th international conference on Machine learning*, pages 1032–1039. ACM, 2008.

Michalis K Titsias and Neil D Lawrence. Bayesian Gaussian process latent variable model. In *International Conference on Artificial Intelligence and Statistics*, pages 844–851, 2010.

Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, and Anind K Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438, 2008.

Appendix of:
Inverse Reinforcement Learning via Deep Gaussian Process

1 Background: Inverse Reinforcement Learning and DGP-IRL

The Markov Decision Process (MDP) is characterized by $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma, \boldsymbol{r}\}$, which represents the state space, action space, transition model, discount factor, and reward function, respectively.

The IRL task is to find the reward function r^* such that the induced optimal policy matches the demonstrations, given $\{\mathcal{S}, \mathcal{A}, \mathcal{T}, \gamma\}$ and $\mathcal{M} = \{\zeta_1, ..., \zeta_h\}$, where $\zeta_i = \{(s_{i,1}, a_{i,1}), ..., (s_{i,T}, a_{i,T})\}$ is the demonstration trajectory, consisting of state-action pairs.

Deep Gaussian process for inverse reinforcement learning (DGP-IRL) extends the deep Gaussian process (deep GP) framework to the IRL domain, as shown in Fig. 1. DGP-IRL learns an abstract representation that reveals the reward structure by warping the original feature space through the latent layers, $\mathbf{D}, \mathbf{B}$.

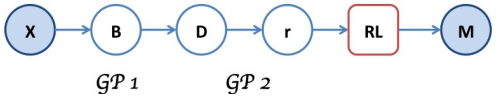


Figure 1: The proposed deep GP model for IRL, where latent Gaussian processes are introduced to learn a representation of the world for the latent reward $\boldsymbol{r}$. The rewards are provided to the reinforcement learning (RL) engine to generate a set of observable trajectories $\mathcal{M}$.

For a set of observed trajectories $\mathcal{M}$, our objective is to optimize the corresponding marginalized log-likelihood given the states in the world as represented by $\mathbf{X}$:

$$\log p(\mathcal{M}|\mathbf{X}) = \log \int p(\mathcal{M}|\boldsymbol{r}) p(\boldsymbol{r}|\mathbf{D}) p(\mathbf{D}|\mathbf{B}) p(\mathbf{B}|\mathbf{X}) d(\boldsymbol{r}, \mathbf{D}, \mathbf{B}) \tag{1}$$

where the integration is with respect to the latent layers, including the reward vector $\boldsymbol{r}$. As introduced in the main paper, $\mathbf{d}^m \in \mathbb{R}^n$ is the m-th column of the latent layer $\mathbf{D} = \begin{bmatrix}\mathbf{d}^1 & \cdots & \mathbf{d}^{m_1}\end{bmatrix}$, and similarly for $\mathbf{B} = \begin{bmatrix}\mathbf{b}^1 & \cdots & \mathbf{b}^{m_1}\end{bmatrix}$:

$$p(\mathcal{M}|\boldsymbol{r}) = \sum_{i=1}^{h} \sum_{t=1}^{T} \left(Q(s_{i,t}, a_{i,t}; \boldsymbol{r}) - V(s_{i,t}; \boldsymbol{r})\right) \tag{2}$$

$$p(\boldsymbol{r}|\mathbf{D}) = \mathcal{N}(\boldsymbol{r}|\mathbf{0}, K_{\mathbf{DD}}) \tag{3}$$

$$p(\mathbf{D}|\mathbf{B}) = \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{d}^m|\mathbf{b}^m, \lambda^{-1}\mathbf{I}) \tag{4}$$

$$p(\mathbf{B}|\mathbf{X}) = \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{b}^m|\mathbf{0}, K_{\mathbf{XX}}) \tag{5}$$

where $p(\mathcal{M}|\boldsymbol{r})$ represents the reinforcement learning term, given by:

$$\log p(\mathcal{M}|\boldsymbol{r}) = \sum_i \sum_t \left(Q(s_{i,t}, a_{i,t}; \boldsymbol{r}) - V(s_{i,t}; \boldsymbol{r})\right) \tag{6}$$

$$= \sum_t \sum_t \left(\boldsymbol{r}_{s_{i,t}, a_{i,t}} - V(s_{i,t}; \boldsymbol{r}) + \sum_{s'} \gamma \mathcal{T}_{s'}^{s_{i,t}, a_{i,t}} V(s'; \boldsymbol{r})\right) \tag{7}$$

The Q-value $Q(s_{i,t}, a_{i,t}; \boldsymbol{r})$ used above is a measure of how desirable is the corresponding state-action pair $(s_{i,t}, a_{i,t})$ under rewards $\boldsymbol{r}$ for all the world states, and is defined by:

$$Q(s_{i,t}, a_{i,t}; \boldsymbol{r}) = \boldsymbol{r}_{s_{i,t}, a_{i,t}} + \sum_{s'} \gamma \mathcal{T}_{s'}^{s_{i,t}, a_{i,t}} V(s'; \boldsymbol{r})$$

where $\boldsymbol{r}_{s_{i,t},a_{i,t}} = r(s_{i,t}, a_{i,t}) \in \mathbb{R}$ is the reward for $(s_{i,t}, a_{i,t})$, γ is the discount factor, $\mathcal{T}_{s'}^{s_{i,t},a_{i,t}} = P(s'|s_{i,t}, a_{i,t})$ is the transition probability by the transition model, and $V(s_{i,t}; \boldsymbol{r})$ is the value associated with state $s_{i,t}$, obtained by the modified Bellman backup operator:

$$V(s_{i,t}; \boldsymbol{r}) = \log \sum_{a \in \mathcal{A}} \exp\left(\boldsymbol{r}_{s_{i,t},a_{i,t}} + \sum_{s'} \gamma \mathcal{T}_{s'}^{s_{i,t},a} V(s'; \boldsymbol{r}) \right)$$

where we apply a **soft-max function** $V(s_{i,t}; \boldsymbol{r}) = \log \sum_{a \in \mathcal{A}} \exp\left(Q(s_{i,t}, a; \boldsymbol{r})\right)$ for the Q-values with all possible actions $a \in \mathcal{A}$. The value function $V(s; \boldsymbol{r})$ for state s can be obtained by repeatedly applying the above Bellman backup operator. For simplicity of notations, we use $V(s_{i,t}; \boldsymbol{r}), Q(s_{i,t}, a_{i,t}; \boldsymbol{r})$ to denote the solution after Bellman backup operators, unlike some literature that uses $V^*(s_{i,t}; \boldsymbol{r}), Q^*(s_{i,t}, a_{i,t}; \boldsymbol{r})$ to denote the difference. Detailed derivation of the above relationships can be found in [4].

2 Variational Lower Bound for DGP-IRL

It is intractable to perform the integration as in (1) for the marginal log-likelihood. In addition to $p(\mathcal{M}|\boldsymbol{r})$, which involves the latent variable $\boldsymbol{r}$ in a way which requires Q-value iterations, the term $p(\boldsymbol{r}|\mathbf{D}) = \mathcal{N}(\boldsymbol{r}|\mathbf{0}, K_{\mathbf{DD}})$ has a nonlinear dependency on $\mathbf{D}$ in the kernel matrix.

To tackle this issue, we introduce inducing outputs $\boldsymbol{f}, \mathbf{V}$ and their corresponding inputs $\mathbf{Z}, \mathbf{W}$, as shown in Fig. 2. The resulting model follows the main paper:

$$p(\mathcal{M}|\boldsymbol{r}) = \sum_{i=1}^{h} \sum_{t=1}^{T} \left(Q(s_{i,t}, a_{i,t}; \boldsymbol{r}) - V(s_{i,t}; \boldsymbol{r}) \right) \tag{8}$$

$$p(\boldsymbol{r}|\boldsymbol{f}, \mathbf{D}, \mathbf{Z}) = \mathcal{N}(\boldsymbol{r}|K_{\mathbf{DZ}} K_{\mathbf{ZZ}}^{-1} \boldsymbol{f}, \mathbf{0}) \tag{9}$$

$$p(\boldsymbol{f}|\mathbf{Z}) = \mathcal{N}(\boldsymbol{f}|\mathbf{0}, K_{\mathbf{ZZ}}) \tag{10}$$

$$p(\mathbf{D}|\mathbf{B}) = \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{d}^m|\mathbf{b}^m, \lambda^{-1}\mathbf{I}) \tag{11}$$

$$p(\mathbf{B}|\mathbf{V}, \mathbf{X}, \mathbf{W}) = \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{b}^m|K_{\mathbf{XW}} K_{\mathbf{WW}}^{-1} \mathbf{v}^m, \boldsymbol{\Sigma}_B) \tag{12}$$

We also design the variation distribution as illustrated in the main paper:

$$\begin{aligned}
\mathcal{Q} &= q(\boldsymbol{f}) q(\mathbf{D}) p(\mathbf{B}|\mathbf{V}, \mathbf{X}) q(\mathbf{V}), \text{ with :} \\
q(\boldsymbol{f}) &= \delta(\boldsymbol{f} - \tilde{\boldsymbol{f}}) \\
q(\mathbf{D}) &= \prod_{m=1}^{m_1} \delta\left(\mathbf{d}^m - K_{\mathbf{XW}} K_{\mathbf{WW}}^{-1} \tilde{\mathbf{v}}^m\right) \\
q(\mathbf{V}) &= \prod_{m=1}^{m_1} \mathcal{N}\left(\mathbf{v}^m|\tilde{\mathbf{v}}^m, \mathbf{G}^m\right),
\end{aligned}$$

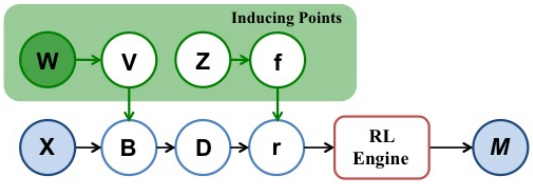


Figure 2: Illustration of DGP-IRL with the inducing outputs $\boldsymbol{f}, \mathbf{V}$ and the corresponding inputs $\mathbf{Z}, \mathbf{W}$.

where the variational distribution $\mathcal{Q}$ is to not be confused with the notation for Q-values, Q. Using the above distribution $\mathcal{Q}$, we can derive the variational lower bound as follows:

$$\log p(\mathcal{M}|\mathbf{X}, \mathbf{Z}, \mathbf{W}) = \log \int p(\mathcal{M}, \boldsymbol{r}, \boldsymbol{f}, \mathbf{V}, \mathbf{D}, \mathbf{B}|\mathbf{Z}, \mathbf{W}, \mathbf{X}) d(\boldsymbol{r}, \boldsymbol{f}, \mathbf{V}, \mathbf{D}, \mathbf{B}) \tag{13}$$

$$= \log \int \underbrace{p(\mathcal{M}|\boldsymbol{r}) p(\boldsymbol{r}|\boldsymbol{f}, \mathbf{D}, \mathbf{Z})}_{p(\mathcal{M}|K_{\mathbf{DZ}} K_{\mathbf{ZZ}}^{-1} \boldsymbol{f})} p(\boldsymbol{f}|\mathbf{Z}) p(\mathbf{D}|\mathbf{B}) p(\mathbf{B}|\mathbf{V}, \mathbf{W}, \mathbf{X}) p(\mathbf{V}|\mathbf{W}) d(\boldsymbol{r}, \boldsymbol{f}, \mathbf{V}, \mathbf{D}, \mathbf{B}) \tag{14}$$

$$\geq \int q(\boldsymbol{f}) q(\mathbf{D}) p(\mathbf{B}|\mathbf{V}, \mathbf{W}, \mathbf{X}) q(\mathbf{V}) \log \frac{p(\mathcal{M}|K_{\mathbf{DZ}} K_{\mathbf{ZZ}}^{-1} \boldsymbol{f}) p(\boldsymbol{f}|\mathbf{Z}) p(\mathbf{D}|\mathbf{B}) p(\mathbf{V}|\mathbf{W})}{q(\boldsymbol{f}) q(\mathbf{D}) q(\mathbf{V})} \tag{15}$$

$$= \log p(\mathcal{M}|K_{\tilde{\mathbf{D}}\mathbf{Z}} K_{\mathbf{ZZ}}^{-1} \tilde{\boldsymbol{f}}) + \log p(\boldsymbol{f} = \tilde{\boldsymbol{f}}|\mathbf{Z}) + \int q(\mathbf{V}) q(\mathbf{D}) p(\mathbf{B}|\mathbf{V}, \mathbf{W}, \mathbf{X}) \log \frac{p(\mathbf{D}|\mathbf{B}) p(\mathbf{V}|\mathbf{W})}{q(\mathbf{V})} d(\mathbf{D}, \mathbf{B}, \mathbf{V}) \tag{16}$$

In the above derivation, the combination of $p(\mathcal{M}|\boldsymbol{r}) p(\boldsymbol{r}|\boldsymbol{f}, \mathbf{D}, \mathbf{Z})$ in (14) uses the deterministic training conditional (DTC) assumption [2], i.e., $p(\boldsymbol{r}|\boldsymbol{f}, \mathbf{D}, \mathbf{Z}) = \delta(\boldsymbol{r} - K_{\mathbf{DZ}} K_{\mathbf{ZZ}}^{-1} \boldsymbol{f})$, (15) applies Jensen's inequality with the variational distribution Q, (16) is a direct consequence of the choice of Q, and $\tilde{\mathbf{D}} = \begin{bmatrix} \tilde{\mathbf{d}}^1 & \cdots & \tilde{\mathbf{d}}^{m_1} \end{bmatrix}$, with $\tilde{\mathbf{d}}^m = K_{\mathbf{XW}} K_{\mathbf{WW}}^{-1} \tilde{\mathbf{v}}^m$.

Utility 1 (Gaussian identities) *If the marginal and conditional Gaussian distributions for $\boldsymbol{f}$ and $\mathbf{v}$ are in the form:*

$$p(\boldsymbol{f}|\mathbf{v}) = \mathcal{N}(\boldsymbol{f}|\mathbf{M}\mathbf{v} + \mathbf{m}, \boldsymbol{\Sigma}_{\boldsymbol{f}})$$
$$p(\mathbf{v}) = \mathcal{N}(\mathbf{v}|\boldsymbol{\mu}_{\mathbf{v}}, \boldsymbol{\Sigma}_{\mathbf{v}})$$

Then the marginal distribution of $\boldsymbol{f}$ is:

$$p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}|\mathbf{M}\boldsymbol{\mu}_{\mathbf{v}} + \mathbf{m}, \boldsymbol{\Sigma}_{\boldsymbol{f}} + \mathbf{M}\boldsymbol{\Sigma}_{\mathbf{v}}\mathbf{M}^\top) \tag{17}$$

Using the Gaussian identities, the derivation of $\int q(\mathbf{V}) p(\mathbf{B}|\mathbf{V}, \mathbf{W}, \mathbf{X}) d\mathbf{V}$ is as follows:

$$\int q(\mathbf{V}) p(\mathbf{B}|\mathbf{V}, \mathbf{W}, \mathbf{X}) d\mathbf{V} = \int \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{v}^m|\tilde{\mathbf{v}}^m, \mathbf{G}^m) \mathcal{N}(\mathbf{b}^m|K_{\mathbf{XW}} K_{\mathbf{WW}}^{-1} \mathbf{v}^m, \boldsymbol{\Sigma}_{\mathbf{B}}) d\mathbf{V}$$
$$= \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{b}^m| \underbrace{K_{\mathbf{XW}} K_{\mathbf{WW}}^{-1} \tilde{\mathbf{v}}^m}_{\tilde{\mathbf{b}}^m}, \underbrace{\boldsymbol{\Sigma}_{\mathbf{B}} + K_{\mathbf{XW}} K_{\mathbf{WW}}^{-1} \mathbf{G}^m K_{\mathbf{WW}}^{-1} K_{\mathbf{WX}}}_{\tilde{\boldsymbol{\Sigma}}_{\mathbf{B}}^m})$$

Therefore, we can obtained a closed form integration for the last term in (16) as follows:

$$
\begin{aligned}
&\int q(\mathbf{V})q(\mathbf{D})p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})\log p(\mathbf{D}|\mathbf{B})d(\mathbf{D},\mathbf{B},\mathbf{V}) \\
&= \int \left(\int q(\mathbf{V})p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})d\mathbf{V}\right) q(\mathbf{D})\log p(\mathbf{D}|\mathbf{B})d(\mathbf{D},\mathbf{B}) \\
&= \int \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{b}^m|\tilde{\mathbf{b}}^m,\tilde{\mathbf{\Sigma}}_{\mathbf{B}}^m)\log \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{d}^m=\tilde{\mathbf{d}}^m|\mathbf{b}^m,\lambda^{-1}\mathbf{I})d\mathbf{B} \\
&= \int \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{b}^m|\tilde{\mathbf{b}}^m,\tilde{\mathbf{\Sigma}}_{\mathbf{B}}^m)\log \prod_{m=1}^{m_1} \left((2\pi)^{-n/2}|\lambda^{-1}\mathbf{I}|^{-1/2}e^{-\frac{\lambda}{2}(\tilde{\mathbf{d}}^m-\mathbf{b}^m)^\top(\tilde{\mathbf{d}}^m-\mathbf{b}^m)}\right) d\mathbf{B} \\
&= \int \prod_{m=1}^{m_1} \mathcal{N}(\mathbf{b}^m|\tilde{\mathbf{b}}^m,\tilde{\mathbf{\Sigma}}_{\mathbf{B}}^m)\left(-\frac{nm_1}{2}\log(2\pi\lambda^{-1}) - \frac{\lambda}{2}\sum_{m=1}^{m_1}(\tilde{\mathbf{d}}^m-\mathbf{b}^m)^\top(\tilde{\mathbf{d}}^m-\mathbf{b}^m)\right) d\mathbf{B} \\
&= -\frac{nm_1}{2}\log(2\pi\lambda^{-1}) - \frac{\lambda}{2}\sum_{m=1}^{m_1}\left(Tr(\tilde{\mathbf{\Sigma}}_{\mathbf{B}}^m) + (\tilde{\mathbf{d}}^m-\tilde{\mathbf{b}}^m)^\top(\tilde{\mathbf{d}}^m-\tilde{\mathbf{b}}^m)\right)
\end{aligned}
$$

where $\tilde{\mathbf{\Sigma}}_B^m = \mathbf{\Sigma}_B + K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\mathbf{G}^m K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}$, $\tilde{\mathbf{b}}^m = K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\tilde{\mathbf{v}}^m$, and $\tilde{\mathbf{d}}^m = K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\tilde{\mathbf{v}}^m$, according to the variational distribution Q.

We now express the variational lower bound of the log likelihood as follow:

$$\mathcal{L} = \mathcal{L}_M + \mathcal{L}_G - \mathcal{L}_{KL} + \mathcal{L}_B - \frac{nm_1}{2}\log(2\pi\lambda^{-1}) \tag{18}$$

where

$$\mathcal{L}_M = \log p(\mathcal{M}|K_{\tilde{\mathbf{D}}\mathbf{Z}}K_{\mathbf{ZZ}}^{-1}\tilde{\boldsymbol{f}}) \tag{19}$$

$$\mathcal{L}_G = \log p(\boldsymbol{f}=\tilde{\boldsymbol{f}}|\mathbf{Z}) = \log\mathcal{N}(\boldsymbol{f}=\tilde{\boldsymbol{f}}|0,K_{\mathbf{ZZ}}) \tag{20}$$

$$= -\frac{1}{2}\tilde{\boldsymbol{f}}^\top K_{\mathbf{ZZ}}^{-1}\tilde{\boldsymbol{f}} - \frac{n_{inducing}}{2}\log(2\pi) - \frac{1}{2}\log|K_{\mathbf{ZZ}}| \tag{21}$$

$$\mathcal{L}_{KL} = KL(q(\mathbf{V})||p(\mathbf{V}|\mathbf{W})) = \sum_{m=1}^{m_1} KL(\mathcal{N}(\mathbf{v}^m|\tilde{\mathbf{v}}^m,\mathbf{G}^m)||\mathcal{N}(\mathbf{v}^m|0,K_{\mathbf{WW}})) \tag{22}$$

$$= \sum_{m=1}^{m_1}\frac{1}{2}\left(Tr(K_{\mathbf{WW}}^{-1}(\mathbf{G}^m+\tilde{\mathbf{v}}^m\tilde{\mathbf{v}}^{m\top}) - n_{inducing} + \log\left(\frac{|K_{\mathbf{WW}}|}{|\mathbf{G}^m|}\right)\right) \tag{23}$$

$$\mathcal{L}_B = -\frac{\lambda}{2}\sum_{m=1}^{m_1} Tr(\mathbf{\Sigma}_{\mathbf{B}} + K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\mathbf{G}^m K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}) \tag{24}$$

which is also described in the main paper. The learning of the model involves optimizing over the variational parameters, including $\tilde{\boldsymbol{f}}, \tilde{\mathbf{v}}^m, \mathbf{G}^m$, inducing inputs $\mathbf{Z}$, as well as hyperparameters for the kernel functions, which is performed through backpropagation based on the gradients of the variational lower bound (18) with respect to these parameters.

3 Optimizing the Variational Distribution $q(\mathbf{V})$

As can be seen, the variational lower bound (18) depends on the parameters of the variational distribution $q(\mathbf{V}) = \prod_{m=1}^{m_1}\mathcal{N}(\mathbf{v}^m|\tilde{\mathbf{v}}^m,\mathbf{G}^m)$, which can be optimized to improve the lower bound further. For the last term in (16), we have

$$
\begin{aligned}
&\int q(\mathbf{V})q(\mathbf{D})p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})\log\frac{p(\mathbf{D}|\mathbf{B})p(\mathbf{V}|\mathbf{W})}{q(\mathbf{V})}d(\mathbf{D},\mathbf{B},\mathbf{V})\\
&=\int q(\mathbf{V})\left(\int q(\mathbf{D})p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})\log\frac{p(\mathbf{D}|\mathbf{B})p(\mathbf{V}|\mathbf{W})}{q(\mathbf{V})}d(\mathbf{D},\mathbf{B})\right)d\mathbf{V}\\
&=\int q(\mathbf{V})\left(\int p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})\log\frac{p(\mathbf{D}=\tilde{\mathbf{D}}|\mathbf{B})p(\mathbf{V}|\mathbf{W})}{q(\mathbf{V})}d\mathbf{B}\right)d\mathbf{B}\\
&=\int q(\mathbf{V})\log\frac{e^{\langle\log p(\mathbf{D}=\tilde{\mathbf{D}}|\mathbf{B})\rangle_{p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})}}p(\mathbf{V}|\mathbf{W})}{q(\mathbf{V})}d\mathbf{V}
\end{aligned}
$$

where we have $\tilde{\mathbf{D}}=\begin{bmatrix}\tilde{\mathbf{d}}^1 & \cdots & \tilde{\mathbf{d}}^{m_1}\end{bmatrix}$, with $\tilde{\mathbf{d}}^m=K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\tilde{\mathbf{e}}^m$, and $\tilde{\mathbf{e}}^m$ for $m=1,...,m_1$ are variational parameters to optimize.

To maximize the above quantity, we can reverse the Jensen's inequality to obtin the condition that:

$$
\log q(\mathbf{V})=const+\langle\log p(\mathbf{D}=\tilde{\mathbf{D}}|\mathbf{B})\rangle_{p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})}+\log p(\mathbf{V}|\mathbf{W})
$$

Now for the term $\langle\log p(\mathbf{D}=\tilde{\mathbf{D}}|\mathbf{B})\rangle_{p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})}$, we have:

$$
\begin{aligned}
&\langle\log p(\mathbf{D}=\tilde{\mathbf{D}}|B)\rangle_{p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})}=\sum_{m=1}^{m_1}\langle\log\mathcal{N}(\mathbf{d}^m=\tilde{\mathbf{d}}^m|\mathbf{b}^m,\lambda^{-1}I)\rangle_{p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})}\\
&=const+\sum_{m=1}^{m_1}\left\langle-\frac{\lambda}{2}Tr\left(\tilde{\mathbf{d}}^m\tilde{\mathbf{d}}^{m\top}+\mathbf{b}^m\mathbf{b}^{m\top}-2\tilde{\mathbf{d}}^m\mathbf{b}^{m\top}\right)\right\rangle_{\mathcal{N}(\mathbf{b}^m|K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\mathbf{v}^m,\mathbf{\Sigma_B})}\\
&=const+\sum_{m=1}^{m_1}\left(-\frac{\lambda}{2}Tr\left(\tilde{\mathbf{d}}^m\tilde{\mathbf{d}}^{m\top}+\mathbf{\Sigma_B}+\mathbf{v}^{m\top}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\mathbf{v}^m-2\mathbf{v}^{m\top}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}\tilde{\mathbf{d}}^m\right)\right)
\end{aligned}
$$

Therefore, we have:

$$
\log q(\mathbf{v}^m)=const-\frac{1}{2}\left(\lambda\mathbf{v}^{m\top}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\mathbf{v}^m-2\lambda\mathbf{v}^{m\top}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}\tilde{\mathbf{d}}^m+\mathbf{v}^{m\top}K_{\mathbf{WW}}^{-1}\mathbf{v}^m\right)
$$

Therefore by completing the squares we have $q(\mathbf{v}^m)=\mathcal{N}(\mathbf{v}^m|\tilde{\mathbf{v}}_*^m,\mathbf{\Sigma}_{\mathbf{v}*}^m)$:

$$
\begin{aligned}
\mathbf{\Sigma}_{\mathbf{v}*}^m&=(K_{\mathbf{WW}}^{-1}+\lambda K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1})^{-1}\\
&=\lambda^{-1}K_{\mathbf{WW}}(\lambda^{-1}K_{\mathbf{WW}}+K_{\mathbf{WX}}K_{\mathbf{XW}})^{-1}K_{\mathbf{WW}}\\
\tilde{\mathbf{v}}_*^m&=\lambda\mathbf{\Sigma}_{\mathbf{v}*}^mK_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}\tilde{\mathbf{d}}^m\\
&=K_{\mathbf{WW}}\underbrace{(\lambda^{-1}K_{\mathbf{WW}}+K_{\mathbf{WX}}K_{\mathbf{XW}})^{-1}}_{\mathbf{\Gamma}}K_{\mathbf{WX}}\tilde{\mathbf{d}}^m
\end{aligned}
$$

With the above optimized variational parameters for $q(\mathbf{v}^m)$, we first obtain:

$$
\begin{aligned}
&\int q(\mathbf{v}^m)\langle\log p(\mathbf{d}^m=\tilde{\mathbf{d}}^m|\mathbf{b}^m)\rangle_{p(\mathbf{b}^m|\mathbf{v}^m,\mathbf{W},\mathbf{X})}d\mathbf{v}^m=\\
&-\frac{n}{2}\log(2\pi\lambda^{-1})-\frac{\lambda}{2}Tr\left(\tilde{\mathbf{d}}^m\tilde{\mathbf{d}}^{m\top}+\mathbf{\Sigma_B}+K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}(\mathbf{\Sigma}_{\mathbf{v}*}^m+\tilde{\mathbf{v}}_*^m\tilde{\mathbf{v}}_*^{m\top})-2\tilde{\mathbf{v}}_*^{m\top}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}\tilde{\mathbf{d}}^m\right)
\end{aligned}
$$

Next, we calculate $\int q(\mathbf{v}^m)\log p(\mathbf{v}^m|\mathbf{W})d\mathbf{v}^m$:

$$
\int q(\mathbf{v}^m)\log p(\mathbf{v}^m|\mathbf{W})=-\frac{n}{2}\log(2\pi)-\frac{1}{2}\log|K_{\mathbf{WW}}|-\frac{1}{2}Tr(K_{\mathbf{WW}}^{-1}(\mathbf{\Sigma}_{\mathbf{v}*}^m+\tilde{\mathbf{v}}_*^m\tilde{\mathbf{v}}_*^{m\top}))
$$

Finally we have:

$$
H(q(\mathbf{v}^m))=q(\mathbf{v}^m)\log\frac{1}{q(\mathbf{v}^m)}=\frac{n}{2}\log(2\pi)+\frac{1}{2}\log|\mathbf{\Sigma}_{\mathbf{v}*}^m| \tag{25}
$$

Summarizing, we have:

$$\int q(\mathbf{V})q(\mathbf{D})p(\mathbf{B}|\mathbf{V},\mathbf{W},\mathbf{X})\log\frac{p(\mathbf{D}|\mathbf{B})p(\mathbf{V}|\mathbf{W})}{q(\mathbf{V})}d(\mathbf{D},\mathbf{B},\mathbf{V})$$
$$\leq\sum_{m=1}^{m_1}\Bigg[-\frac{n}{2}\log(2\pi\lambda^{-1})-\frac{1}{2}\log|K_{\mathbf{WW}}|-\frac{1}{2}Tr(K_{\mathbf{WW}}^{-1}(\mathbf{\Sigma}_{\mathbf{v}*}^m+\tilde{\mathbf{v}}_*^m\tilde{\mathbf{v}}_*^{m\top}))+\frac{1}{2}\log|\mathbf{\Sigma}_{\mathbf{v}*}^m|$$
$$-\frac{\lambda}{2}Tr\Bigg(\tilde{\mathbf{d}}^m\tilde{\mathbf{d}}^{m\top}+\mathbf{\Sigma_B}+K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}(\mathbf{\Sigma}_{\mathbf{v}*}^m+\tilde{\mathbf{v}}_*^m\tilde{\mathbf{v}}_*^{m\top})-2\tilde{\mathbf{v}}_*^{m\top}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}\tilde{\mathbf{d}}^m\Bigg)\Bigg]$$

We now express the variational lower bound of the log likelihood as follow:

$$\mathcal{L}=\mathcal{L}_M+\mathcal{L}_G+\mathcal{L}_{DBV} \tag{26}$$

where

$$\mathcal{L}_M=\log p(\mathcal{M}|K_{\tilde{\mathbf{D}}\mathbf{Z}}K_{\mathbf{ZZ}}^{-1}\tilde{\boldsymbol{f}}) \tag{27}$$
$$\mathcal{L}_G=\log p(u=\tilde{u}|Z)=\log\mathcal{N}(u=\tilde{u}|0,K_{ZZ}) \tag{28}$$
$$=-\frac{1}{2}\tilde{u}^\top K_{ZZ}^{-1}\tilde{u}-\frac{K}{2}\log(2\pi)-\frac{1}{2}\log|K_{ZZ}| \tag{29}$$
$$\mathcal{L}_{DBV}=\sum_{m=1}^{m_1}\Bigg[-\frac{n}{2}\log(2\pi\lambda^{-1})-\frac{1}{2}\log|K_{\mathbf{WW}}|-\frac{1}{2}Tr(K_{\mathbf{WW}}^{-1}(\mathbf{\Sigma}_{\mathbf{v}*}^m+\tilde{\mathbf{v}}_*^m\tilde{\mathbf{v}}_*^{m\top}))+\frac{1}{2}\log|\mathbf{\Sigma}_{\mathbf{v}*}^m|$$
$$-\frac{\lambda}{2}Tr\Bigg(\tilde{\mathbf{d}}^m\tilde{\mathbf{d}}^{m\top}+\mathbf{\Sigma_B}+K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}(\mathbf{\Sigma}_{\mathbf{v}*}^m+\tilde{\mathbf{v}}_*^m\tilde{\mathbf{v}}_*^{m\top})-2\tilde{\mathbf{v}}_*^{m\top}K_{\mathbf{WW}}^{-1}K_{\mathbf{WX}}\tilde{\mathbf{d}}^m\Bigg)\Bigg] \tag{30}$$

where $\tilde{\mathbf{d}}^m=K_{\mathbf{XW}}K_{\mathbf{WW}}^{-1}\tilde{\mathbf{e}}^m$, $\mathbf{\Gamma}=(\lambda^{-1}K_{\mathbf{WW}}+K_{\mathbf{WX}}K_{\mathbf{XW}})^{-1}$,$\Sigma_{\mathbf{v}*}^m=\lambda^{-1}K_{\mathbf{WW}}\mathbf{\Gamma}K_{\mathbf{WW}}$.

The parameters we need to learn in this case include the variational parameters $\tilde{\boldsymbol{f}}$, and $\tilde{\mathbf{e}}^m$ for $m=1,...,m_1$, inducing inputs $\mathbf{Z}$, as well as hyperparameters for kernel functions.

4 Parameters Learning by Derivatives

In this section, we will obtain the derivatives of the marginal log likelihood $\mathcal{L}$ in (26) with respect to the variational parameters $\tilde{\boldsymbol{f}}$, $\tilde{\mathbf{e}}^m$ and inducing inputs $\mathbf{Z}$. The derivative of the reinforcement learning term, $p(\mathcal{M}|\boldsymbol{r})$ in (7), with respect to the reward vectors $\boldsymbol{r}$, is given by:

$$\frac{\partial}{\partial\boldsymbol{r}}\log p(\mathcal{M}|\boldsymbol{r})=\sum_i\sum_t\left(\frac{\partial}{\partial\boldsymbol{r}}\boldsymbol{r}_{s_{i,t},a_{i,t}}-\frac{\partial}{\partial\boldsymbol{r}}V_{s_{i,t}}^{\boldsymbol{r}}+\sum_{s'}\gamma\mathcal{T}_{s'}^{s_{i,t},a_{i,t}}\frac{\partial}{\partial\boldsymbol{r}}V_{s'}^{\boldsymbol{r}}\right) \tag{31}$$

The first term, $\sum_i\sum_t\frac{\partial}{\partial\boldsymbol{r}}\boldsymbol{r}_{s_{i,t},a_{i,t}}$, is simply a vector that counts the number of state-action pairs in the demonstrations $\hat{\mu}$, whose entry corresponding to (s,a) is given by: $\hat{\mu}_{s,a}=\sum_i\sum_t 1_{s_{i,t}=s\wedge a_{i,t}=a}$. The second term involves the derivative of the value function at state s with respect to rewards, as indicated in [4], equal to the expected visitation count of each state-action pair when starting from state s and following the optimal stochastic policy, i.e., $\frac{\partial}{\partial\boldsymbol{r}}V_s^{\boldsymbol{r}}=E[\mu|s]$, where μ is a vector with each entry $\mu_{s,a}$ corresponding to the expected visitation count for (s,a). Therefore, (31) can be written as:

$$\frac{\partial}{\partial\boldsymbol{r}}\log p(\mathcal{M}|\boldsymbol{r})=\hat{\mu}-\sum_i\sum_t E[\mu|s_{i,t}]+\sum_i\sum_t\sum_{s'}\gamma\mathcal{T}_{s'}^{s_{i,t},a_{i,t}}E[\mu|s_{i,t}]$$
$$=\hat{\mu}-\sum_s\hat{\nu}_sE[\mu|s]$$

where $\hat{\nu}_s=\sum_a\hat{\mu}_{s,a}-\sum_i\sum_t\gamma\mathcal{T}_{s'}^{s_{i,t},a_{i,t}}$. The term $\sum_s\hat{\nu}_sE[\mu|s]$ can be computed efficiently by a simple iterative algorithm described in [4], which we do not recount here. Note that the above derivation follows from [1].

For the variational parameters $\tilde{\boldsymbol{f}}$, we need to consider only two terms that involve it, i.e., $\mathcal{L}_{\mathcal{M}}, \mathcal{L}_G$:

$$\frac{\partial \mathcal{L}_M}{\partial \tilde{\boldsymbol{f}}} = \frac{\partial \boldsymbol{r}}{\partial \tilde{\boldsymbol{f}}} \frac{\partial \mathcal{L}_{\mathcal{M}}}{\partial \boldsymbol{r}} = K_{\tilde{\mathbf{D}}\mathbf{Z}} K_{\mathbf{Z}\mathbf{Z}}^{-1} \frac{\partial \log p(\mathcal{M}|\boldsymbol{r})}{\partial \boldsymbol{r}}$$
$$\frac{\partial \mathcal{L}_G}{\partial \tilde{\boldsymbol{f}}} = -K_{\mathbf{Z}\mathbf{Z}}^{-1} \tilde{\boldsymbol{f}}$$

where $\boldsymbol{r} = K_{\tilde{\mathbf{D}}\mathbf{Z}} K_{\mathbf{Z}\mathbf{Z}}^{-1} \tilde{\boldsymbol{f}}$ is the reward vector that we use for reinforcement learning.

For the variational parameters $\tilde{\mathbf{e}}^m$, let $\tilde{\mathbf{D}} = \left[K_{\mathbf{X}\mathbf{W}} K_{\mathbf{W}\mathbf{W}}^{-1} \tilde{\mathbf{e}}^1, ..., K_{\mathbf{X}\mathbf{W}} K_{\mathbf{W}\mathbf{W}}^{-1} \tilde{\mathbf{e}}^{m_1}\right] \in \mathbb{R}^{n \times m_1}$, and $\mathbf{E} = [\tilde{\mathbf{e}}^1, ..., \tilde{\mathbf{e}}^{m_1}] \in \mathbb{R}^{K \times m_1}$:

$$\frac{\partial \mathcal{L}_{\mathcal{M}}}{\partial \mathbf{E}} = \frac{\partial \tilde{\mathbf{D}}}{\partial \mathbf{E}} \frac{\partial K_{\tilde{\mathbf{D}}\mathbf{Z}}}{\partial \tilde{\mathbf{D}}} \frac{\partial \boldsymbol{r}}{\partial K_{\tilde{\mathbf{D}}\mathbf{Z}}} \frac{\partial \mathcal{L}_{\mathcal{M}}}{\partial \boldsymbol{r}}$$

In addition, by applying matrix derivatives,

$$\frac{\partial \mathcal{L}_{DBV}}{\partial \mathbf{e}^m} = -\frac{\lambda}{2} \Bigg(2K_{\mathbf{W}\mathbf{W}}^{-1} K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} K_{\mathbf{W}\mathbf{W}}^{-1} + 2K_{\mathbf{W}\mathbf{W}}^{-1} K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} \boldsymbol{\Gamma} K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} \Gamma K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} K_{\mathbf{W}\mathbf{W}}^{-1}$$
$$-4K_{\mathbf{W}\mathbf{W}}^{-1} K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} \boldsymbol{\Gamma} K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} K_{\mathbf{W}\mathbf{W}}^{-1} \Bigg) \mathbf{e}^m - K_{\mathbf{W}\mathbf{W}}^{-1} K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} \boldsymbol{\Gamma} K_{\mathbf{W}\mathbf{W}} \boldsymbol{\Gamma} K_{\mathbf{W}\mathbf{X}} K_{\mathbf{X}\mathbf{W}} K_{\mathbf{W}\mathbf{W}}^{-1} \mathbf{e}^m$$

The gradients are provided to minFunc [3], which calls a quasi-Newton strategy, where limited-memory BFGS updates with Shanno-Phua scaling are used in computing the step direction, and a bracketing line-search for a point satisfying the strong Wolfe conditions is used to compute the step direction.

References

[1] S. Levine, Z. Popovic, and V. Koltun. Nonlinear inverse reinforcement learning with Gaussian processes. In *Advances in Neural Information Processing Systems*, pages 19–27, 2011.

[2] J. Quiñonero-Candela and C. E. Rasmussen. A unifying view of sparse approximate gaussian process regression. *The Journal of Machine Learning Research*, 6:1939–1959, 2005.

[3] M. Schmidt. minfunc: unconstrained differentiable multivariate optimization in matlab. *URL http://www. di. ens. fr/mschmidt/Software/minFunc. html*, 2012.

[4] B. D. Ziebart, J. A. Bagnell, and A. K. Dey. Modeling interaction via the principle of maximum causal entropy. In *International Conference on Machine Learning (ICML)*, pages 1247–1254, 2010.