Unsupervised Inverse Reinforcement Learning with Noisy Data

Amit Surana

Abstract—In this paper we propose an approach for unsupervised Inverse Reinforcement Learning (IRL) with noisy data using a hidden variable Markov Decision Processes (hMDP) representation. hMDP accounts for observation uncertainty by using a hidden state variable. We develop a nonparametric Bayesian IRL technique for hMDP based on Dirichlet Processes mixture model. We provide an efficient Markov Chain Monte Carlo based sampling algorithm whereby one can automatically cluster noisy data into different behaviors, and estimate the underlying reward parameters per cluster. We demonstrate our approach for unsupervised learning, and prediction and classification of agent behaviors in a simulated surveillance scenario.

I. INTRODUCTION

One of the most important goals of surveillance systems is to analyze human activities in order to detect anomalies, and predict future activities before they occur. Deviations from the normal behavior can be effectively detected by exploring the motion patterns [1]. There is a wide literature covering these topics, e.g., [2], [3] which use object level representation and rely on trajectory modeling, while [4], [5], [6] use group level representation for crowd behavior.

In this paper we focus on trajectory modeling techniques based on Markov Decision Processes (MDP) and its variants for behavior learning, prediction and classification in surveillance applications. MDPs have been a popular approach for modeling sequential decision making [7]. They offer several advantages for human behavioral modeling in surveillance scenarios [8], [9]. Firstly, rational human activities can be thought of as being driven by immediate rewards, expected future rewards and goals, which can be naturally captured as a MDP. Secondly, MDP representation encapsulates behaviors in terms of physical scene features and not physical location, and so has the ability to generalize to novel scenes. Thirdly, MDP representation can be reliably learned using limited data compared to e.g. location based Markov model which degenerates to random walk when not enough training data is available [8].

Application of MDP framework for trajectory based behavior learning in computer vision applications require additional considerations. Firstly, rewards/preferences which drive the agent behavior in a particular environment may not be known apriori, and have to be learned from trajectories extracted from video data. Secondly, the trajectories which are output from a tracking algorithm are typically noisy. As a result the true state (e.g. position) of agent is not directly observable. Thirdly, in videos there are typically many agents, and the number of different behaviors and the behavior labels for each agent trajectory may not be known a priori, and should also be learned in addition to the rewards.

For the first problem, Inverse Reinforcement Learning techniques can be used, which given the environment model and the data, determine reward function that explains the observed behavior. A variety of IRL algorithms exist in literature which can be broadly categorized into optimization based techniques [10], [11], [12], [13], [14] and Bayesian approaches [15], [16]. All, these approaches however assume that the underlying state of MDP is fully observable. For IRL with noisy data, authors in [8] introduced the concept of hidden variable MDP (hMDP). In hMDP observation uncertainty is modeled via a hidden state variable like in Partially Observable Markov Decision Process (POMDP) [17]. However, hMDP is different than POMDP in the sense that the agent is not uncertain about its own state (and does not have to account for that uncertainty in making decisions), its only the observer who has noisy observations of agent’s state. For hMDP IRL, authors extended the causal entropy IRL approach [13] for learning and forecasting behavior from noisy trajectories, assuming that the trajectories are labeled. The third problem has been considered in context of standard MDP IRL setting (i.e. non-noisy case) in [18], where authors use a Bayesian nonparametric (BNP) approach which does not require the knowledge of number of distinct behavior clusters, and the assignment of trajectories into those clusters.

In this paper we propose a framework which can deal with all the three problems simultaneously. We argue that it is natural to develop this framework in a Bayesian IRL (BIRL) setting. Accordingly, we first develop hMDP BIRL (hBIRL) techniques, assuming noisy trajectory labels are given. For this we exploit the fact that for a fixed policy, hMDP reduces to a Hidden Markov Model (HMM). Hence, Markov Chain Monte Carlo (MCMC) methods developed for parameter learning in HMMs [19], [20], can be employed. In particular, we describe two approaches for hBIRL: one is based on likelihood recursion, which marginalizes over the hidden state sequence in the underlying HMM, and the other uses forward-backward Gibbs sampling. The latter approach is preferred as it leads to a faster mixing Markov chain. We next extend the hBIRL framework to nonparametric setting for which we employ a Dirichlet Processes (DP) mixture model as a prior over the behavior clusters, and use a MCMC sampling procedure. During this sampling, the clusters, reward parameters per cluster, and the underlying state sequence per trajectory are sampled sequentially utilizing Chinese Restaurant Process representation of the DP mixture model. This BNP approach automatically partitions the trajectories without the need to specify a priori the number...
of distinct behaviors present in the dataset. We demonstrate the application of this nonparametric-hBIRL (NP-hBIRL) for unsupervised behavior learning in a simulated urban surveillance scenario. We also propose a recursive Bayesian approach for behavior classification and prediction with the hMDP models, and illustrate it in the scenarios considered.

The paper is organized into seven sections. We summarize the mathematical preliminaries for hMDPs, and review existing BIRL techniques in Sections II and III, respectively. Section IV describes the BIRL for hMDPs, and its unsupervised extension using BNP framework is covered in Section V. We present an approach for behavior classification and prediction based on the hMDP representation in Section VI. In Section VII we present a numerical example, and conclude the paper in Section VIII with directions for future work.

II. Hidden Variable MDP

The concept of hidden variable MDP (hMDP) extends the notion of MDPs to model an observer who has noisy observations of state, and was introduced in [8]. Formally, hMDP is $M^h = (S, A, \mathcal{Y}, \mathcal{P}, \mathcal{P}^w, \mathcal{R}, \chi, \nu)$, where

- $S = \{1, \cdots, N_s\}$ is finite state space
- $A = \{1, \cdots, N_a\}$ is finite action space
- $\mathcal{Y} \subset \mathbb{R}^n$ is the space of observed variables
- $\mathcal{P}(s, a, s')$ is the action dependent state transition dynamics
- $\mathcal{P}^w(s, y)$ is the likelihood of observing $y \in \mathcal{Y}$ given the state $s$
- $\nu$ is the discount factor
- $\chi$ is initial distribution on $S$
- $\mathcal{R} : S \times A \to \mathbb{R}$ is the reward function

A policy $\pi : S \times A \to [0, 1]$ is a mapping from states to probabilities of taking each available action, i.e., $\pi(s, a)$ is the probability with which the policy $\pi$ chooses actions $a \in A$ in state $s$. In this paper it will suffice to restrict to the class of stationary Markovian randomized policies [7].

A hMDP $M^h$ has an underlying MDP which we shall denote by $M = (S, A, \mathcal{P}, \mathcal{R}, \chi, \nu)$. As in MDP, the state-value function $V^\pi(s)$ for an hMDP is given by

$$V^\pi(s) = E^\pi \left[ \sum_{t=0}^{\infty} \nu^t R(s_t, a_t) | s_0 = s \right],$$

and is the infinite horizon expected reward for starting in state $s$ and following the policy $\pi$ thereafter. Similarly, action-value function $Q^\pi(s, a)$ is defined as the infinite-horizon expected reward starting in state $s$, taking action $a$ and thereafter following the policy $\pi$. For all $s \in S, a \in A, V^\pi$ and $Q^\pi$ satisfy the Bellman Equations [7].

$$V^\pi(s) = \sum_{a \in A} \pi(s, a) \left[ R(s, a) + \nu \sum_{s' \in S} \mathcal{P}(s, a, s') V^\pi(s') \right]$$

$$Q^\pi(s, a) = R(s, a) + \nu \sum_{s' \in S} \mathcal{P}(s, a, s') V^\pi(s').$$

The optimal action-value function $Q^*$ is such that $Q^*(s, a) = \max_{\pi} Q^\pi(s, a), \ \forall (s, a) \in S \times A$.

We shall denote by $\pi^*$ the corresponding optimal policy. Given $M$, the optimal policy $\pi^*$ can be computed using various well known methods such as value iteration, policy iteration, and via linear programming [7]. In our computations, we use a linear programming approach, and we will refer to this operation by MDPLPPolicy($M$).

A. Reward Parameterized hMDP

In this paper we will consider reward paramertized hMDPs $M^h_\theta = (S, A, \mathcal{Y}, \mathcal{P}^w, \mathcal{R}, \chi, \nu)$ with a linear parameterization for reward function

$$R(s, a) = \theta_1 \phi_1(s, a) + \cdots + \theta_N \phi_N(s, a),$$

where, $\phi_i : S \times A \to \mathbb{R}, i = 1, \cdots, N$ are the reward basis function, and $\theta = (\theta_1, \cdots, \theta_N)^T \in \mathbb{R}^N$ are real valued coefficients. For a given $\theta$ we will denote by $\pi^*_\theta$ the optimal policy, and by $Q^*_\theta$ and $V^*_\theta$ the corresponding optimal $Q-$function and the value function, respectively.

We will use $M^h_{\theta} = (S, A, \mathcal{P}^w, \chi, \nu)$ to denote the case when the reward parameters $\theta$ are unknown, and analogously $M^h_{\{\theta\}}$.

III. Bayesian IRL for MDP

Let $M_{\theta}$ be a reward parametrized MDP, and let $D^s = \{s_0, T_i, \cdots, s_M \}$, where the trajectories $s_0, T_i, s_{i+1}, \cdots, s_M$ are assumed to be generated using some unknown policy $\pi^E$. The reward parameters $\theta$ which $\pi^E$ optimizes are also not known. Given the behaviour data $D^s$ and the MDP $M^h_{\{\theta\}}$, the IRL problem involves finding a policy $\pi^L$, such that it is as good as the unknown policy $\pi^E$.

IRL problem is inherently ill-posed, since there are infinitely many reward functions that may yield the policy $\pi^E$ as optimal [10], [16]. To address this uniqueness issue, different approaches have been proposed in the literature to introduce preferences over the reward or policy function spaces. These approaches can be broadly categorised as Optimization based IRL, and Bayesian IRL. Optimization based IRL approaches encode the preferences by using appropriate objective functions and/or constraints [10], [21], [11], [13], [22]; while Bayesian approach capture the preferences in form of a prior distribution, and behavior compatibility as a likelihood function.

In Bayesian IRL (BIRL), the reward learning task is cast as a standard Bayesian inference problem [15], [16]. By encoding the reward parameter preference as the prior $P(\theta)$, and the optimality confidence of the behaviour data $D^s$ as the likelihood $L(D^s; M_{\theta})$, the posterior $P(\theta|D^s)$ over reward parameters can be determined by applying Bayes Theorem,

$$P(\theta|D^s) = \frac{L(D^s; M_{\theta}) P(\theta)}{P(D^s)},$$

where, $P(D^s)$ is a normalizing constant. We will use parameterized distribution $H(\theta; \lambda)$ (where $\lambda$ are the hyper parameters) such as uniform, Gaussian or a Laplacian for the
prior $P(\theta)$. The likelihood $\mathcal{L}(D^*; M_\theta)$ in BIRL is typically defined as an independent exponential distribution:

$$
\mathcal{L}(D^*; M_\theta) = \Pi_i^M \mathcal{L}(s^i_{0:T}; M_\theta),
$$

(4)

where

$$
\mathcal{L}(s_{0:T}; M_\theta) = \chi(s_0) \prod_{t=1}^T \mathcal{P}_\pi(s_{t-1}, s_t),
$$

(5)

$$
\mathcal{P}_\pi(s_{t-1}, s_t) = \sum_a \mathcal{P}(s_{t-1}, a, s_t) \pi(s_{t-1}, a),
$$

(6)

and

$$
\pi_\theta^\eta(s, a) = \frac{\exp(\eta Q_\theta^*(s, a))}{\sum_{a\in A} \exp(\eta Q_\theta^*(s, a))},
$$

(7)

with $\eta$ being a parameter that is equivalent to the inverse of temperature in the Boltzmann distribution and represents our degree of confidence in the agents ability to behave in a way that maximizes the reward, see [15].

For the MDP likelihood model (4), the posterior $P(\theta|D^*)$ cannot be computed in a closed form, and one has to resort to Markov chain Monte Carlo (MCMC) methods [23]. The MCMC methods obtain a representation of the posterior by iteratively generating samples from it, see Algo. 1.

**Algorithm 1 Prototypical BIRL**

$$
\{\theta^i\}_{i=0}^{N_{\theta}} = \text{BIRL}(M_\theta, H, \tau, N_{sp}, D^*)
$$

1. Sample $\theta^0 \sim H(\cdot; \lambda)$ and $Q_{\theta^0}^* = \text{MDPLP}(M_{\theta_0})$.
2. For $i = 1, \cdots, N_{sp}$ do following,
   
   $\theta^i = \text{SampleTheta}(\theta^{i-1}, M_\theta, H, \tau, D^*)$.

3. Return $\{\theta^i\}_{i=0}^{N_{\theta}}$.

**Algorithm 2 Sample Theta for MDP**

$$
\theta = \text{SampleTheta}(\theta^0, M_\theta, H, \tau, D^*)
$$

1. Sample $\epsilon \sim \text{Normal}(0, 1)$, and let $\tilde{\theta} = \theta^0 + \tau \epsilon$.
2. Compute new policy $\tilde{Q}^* = \text{MDPLP}(M_{\tilde{\theta}})$.
3. Set $\hat{\theta} = \tilde{\theta}$ with probability
   
   $$
   \min \left( 1, \frac{\mathcal{L}(D^*; M_{\hat{\theta}})H(\hat{\theta}; \lambda)}{\mathcal{L}(D^*; M_{\theta^0})H(\theta^0; \lambda)} \right),
   $$

   else $\hat{\theta} = \theta^0$.
4. Return $\hat{\theta}$.

Note that Algo. 1 does not require knowledge of the normalizing constants (see step 3 in Algo. 2), and so the likelihood and the prior are only required to be known up to a constant. It should also be noted that step 3 in Algo. 2 requires solving an optimal policy for a MDP with parameters $\hat{\theta}$ which can become a computational bottleneck for large state space MDPs. To alleviate this, several approaches have been proposed in [15], [24], [16].

Once the samples from the posterior $P(\theta|D^*)$ are available, one can compute an empirical posterior mean/median which is optimal under the mean square/linear loss function, respectively [15]. A gradient method for maximum-a-posterior (MAP) estimation based on the (sub)differentiability of the posterior distribution is given in [16].

**IV. BIRL FOR HMDP**

In hMDP case the state trajectories $s_{0:T}$ are not observed directly. Only noisy measurements of states are observed, i.e. $y_{0:T} = \{y_t \sim \mathcal{P}_\theta(y_t|s_t) : t = 0, \cdots, T\}$. Under these circumstances, Eq. (5) cannot be used directly for the likelihood computation in Algo. 2.

However, note that for a given policy $\pi$, hMDP reduces to a standard HMM, $H_\pi = (S, Y, \mathcal{P}_\pi, \mathcal{P}_\theta, \chi)$, where $\mathcal{P}_\pi$ is transition matrix defined in Eq. (6). We will denote by $H_\theta = H_{\pi^\eta}$, where $\pi^\eta$ is the policy defined in Eq. (7). Thus, the likelihood $\mathcal{L}(y_{0:T}; M^h_\theta) = \mathcal{L}(y_{0:T}, H_\theta)$ can be computed as

$$
\mathcal{L}(y_{0:T}; M^h_\theta) = \mathcal{L}(y_{0:T}, H_\theta) = \sum_{s_{0:T}} P(y_{0:T}, s_{0:T}|H_\theta),
$$

(8)

where

$$
P(y_{0:T}, s_{0:T}|H_\theta) = \chi(s_0) \mathcal{P}_\theta(y_0, s_{0:T}) \prod_{t=1}^T \mathcal{P}_{\pi^\eta}(s_{t-1}, s_t) \mathcal{P}_\theta(s_t, y_t).
$$

(9)

The sum in (8) is over $S^{T+1}$ elements, so it quickly becomes infeasible to compute even for small size of $S$. Several approaches developed for HMM can be used to avoid this explosion: Likelihood Recursion, Direct Gibbs Sampling, and Forward-Backward Gibbs sampling, see reviews [19], [20]. We next describe BIRL approaches for hMDPs (which we refer to as hBIRL) based on likelihood recursion and Forward-Backward Gibbs sampling.

**A. hBIRL with Likelihood Recursion**

Given $H_\theta$, the likelihood recursion employs forward variable,

$$
\alpha_t(s) = P(y_{0:t}, s_t = s),
$$

(10)

which can be recursively computed as:

$$
\alpha_t(s) = \mathcal{P}_\theta(y_t, s) \sum_r \mathcal{P}_{\pi^\eta}(r, s) \alpha_{t-1}(r),
$$

(11)

with the initialization, $\alpha_0(s) = \chi(s) \mathcal{P}_\theta(s, y_0)$. Hence, data likelihood can be recovered as

$$
\mathcal{L}(y_{0:T}; H_\theta) = \sum_s \alpha_T(s).
$$

(12)

It takes $O(|S|^{2T})$ operations to compute likelihood using above recursion.

Let $D^\theta = \{y^i_{0:T} : i = 1, \cdots, M\}$ be the training dataset, then

$$
\mathcal{L}(y^i_{0:T}; H_\theta) = \Pi_i^M \mathcal{L}(y^i_{0:T}; H_\theta),
$$

(13)

where, $\mathcal{L}(y^i_{0:T}; H_\theta)$ is computed using Eq. (12). Using this likelihood recursion, Algo. 3 describes a hBIRL algorithm.
where in step 2, \( h_{\text{SampleTheta}} \) is computed using the Algorithm 2 with the probability ratio in step 3 (of Algorithm 2) replaced by:

\[
\min \left( 1, \frac{L(D^y; \mathcal{M}_h^b, H(\theta; \lambda))}{L(D^v; \mathcal{M}_w^b, H(\theta^v; \lambda))} \right).
\]

**Algorithm 3 hBIRL using Likelihood Recursion**

\[
\{\theta^i\}_{i=0}^{N_{sp}} = \text{hBIRL} \left( \mathcal{M}_h^b, H, \tau, N_{sp}, D^v \right)
\]

1. Sample \( \theta^0 \sim H(\cdot, \lambda) \) and \( Q^*_{00} = \text{MDPLP}(\mathcal{M}_{00}) \).
2. For \( i = 1, \cdots, N_{sp} \) do following,
   \[
   \theta^i = h_{\text{SampleTheta}}(\theta^{i-1}, \mathcal{M}_h^b, H, \tau, D^v).
   \]
3. Return \( \{\theta^i\}_{i=0}^{N_{sp}} \).

Metropolis-Hastings (MH) type MCMC procedure used in Algorithm 3 which average over \( s_{0:T} \) are useful, but there are reasons to prefer sampling \( s_{0:T} \) instead of integrating it out. When the dimension of \( \theta \) is large, the MH sampling procedure can move slowly through the parameter space, leading to poor performance [20]. Moreover, highly correlated elements of \( \theta \) under \( P(\theta|y_{0:T}) \) are often nearly independent under \( P(\theta|s_{0:T}, y_{0:T}) \). Next section discusses Gibbs type MCMC sampling which alleviates this problem.

**B. hBIRL with Gibbs Sampling**

In our application, in order to sample from \( P(\theta, s_{0:T}, y_{0:T}, \lambda) \) the HMMs (i.e. \( \mathcal{H}_0 \)) missing data structure can be exploited which admits posterior samplers that alternate between simulating \( P(s_{0:T}|y_{0:T}, \theta, \lambda) = P(s_{0:T}|y_{0:T}, \theta) \) and simulating \( P(\theta|y_{0:T}, s_{0:T}, \lambda) = P(\theta|s_{0:T}, \lambda) \) given \( s_{0:T} \).

Direct Gibbs (DG) [25] and Forward-Backward Gibbs (FBG) [26], [27] sampling have been proposed in literature for sampling \( P(s_{0:T}|y_{0:T}, \theta) \). We focus on FBG, as it introduces fewer components into the Gibbs Markov chain and promotes rapid mixing than DG, see [20] for details. Algorithm 4 describes the FBG procedure for hBIRL. Once, the sample \( s_{0:T} \) is available, \( \theta \) can be sampled from \( P(\theta|s_{0:T}, \lambda) \) using the MH procedure as described in Algorithm 2.

The state sequence \( s_{0:T} \) is sampled using Forward-Backward procedure for HMMs as described in Algorithm 5.

**Algorithm 4 hBIRL using Forward-Backward Gibbs**

\[
\{\theta^i\}_{i=0}^{N_{sp}} = \text{hBIRL} \left( \mathcal{M}_h^b, H, \tau, N_{sp}, D^v \right)
\]

1. Sample \( \theta^0 \sim H(\lambda) \) and \( Q^*_{00} = \text{MDPLP}(\mathcal{M}_{00}) \).
2. For \( i = 1, \cdots, N_{sp} \) do following,
   - For \( j = 1, \cdots, M \)
     \[
     s_{0:T_j} = \text{SampleStateSeq}(\theta^{i-1}, \mathcal{M}_h^b, y_{0:T_j}).
     \]
   - Sample new \( \theta^i \),
     \[
     \theta^i = \text{SampleTheta}(\theta^{i-1}, \mathcal{M}_h^b, H, \tau, D^v).
     \]
3. Return \( \{\theta^i\}_{i=0}^{N_{sp}} \).

**Algorithm 5 Forward Backward Procedure for State Sampling**

\[
s_{0:T} = \text{SampleStateSeq}(\theta, \mathcal{M}_h^b, y_{0:T})
\]

1. Let \( \mathcal{H}_0 \) be HMM corresponding to \( \mathcal{M}_0^b \).
2. Compute the forward variables \( \alpha_t, t = 0, \cdots, T \) using Eq. (11).
3. Moving backward sample \( s_t, t = T - 1, \cdots, 0 \) using
   \[
   s_t \sim P(s) \propto \alpha_t(s) P_{\pi_0}(s, s_{t+1})
   \]
   with the initialization \( s_T \sim \alpha_T \).

This approach uses a Bayesian nonparametric (BNP) framework that does not require the knowledge of the number of distinct behavior clusters. We first review briefly the BNP approach for unsupervised clustering.

**A. Bayesian Nonparametrics**

Classical parametric methods for inferring the latent clusters in data, often rely on fixing the number of clusters. On the other hand, BNP methods provide a much richer and flexible model representation, and learning using Bayesian techniques [28]. Rather than comparing models that vary in complexity (and choose the best one) like in classical approaches, the BNP approach is to fit a single model that can adapt its complexity to grow as more data is observed. This is essential in complex settings, where the space of models to be searched is difficult to efficiently enumerate and explore. BNP framework achieves such as representation and flexibility by defining distributions on functions spaces such as that of probability measures. Here we briefly review Dirichlet Process (DP) family of BNP methods, see [28], [29] for details.

DP denoted by \( \text{DP}(\gamma, H) \), is a distribution over distribution parameterized by a base distribution \( H \) and a concentration parameter \( \gamma \) [30]. A sample \( G_0 \sim \text{DP}(\gamma, H) \) is given by

\[
G_0 = \sum_{k=1}^{\infty} \beta_k \delta_{\theta_k}, \quad \theta_k|H \sim H, k = 1, \cdots
\]
where, $\delta$ is the delta function, and $\beta \sim \text{GEM}(\gamma)$ denotes a stick breaking construction

$v_k|\gamma \sim \text{Beta}(1, \gamma), \beta_k = v_k \prod_{j=1}^{k-1}(1 - v_j), \ k = 1, \cdots.$

There are other alternative representations of DP such as Polya’s Urn Predictions, and Chinese Restaurant Process (CRP) which facilitate inference with these models [31]. DP is commonly used as a prior on the parameters of a mixture model with a random and potentially countably infinite number of mixture components. A generative DP mixture model for a dataset $\{y_1, \cdots, y_M\}$ using a set of latent parameters $\theta_i, i = 1, \cdots, K$ can be defined as:

$$G \sim \text{DP}(\gamma, H),$$

$$\theta_i|G \sim G, \ i = 1, 2, \cdots,$$

$$y_i|\{\theta_i\}_{i=1}^\infty \sim F(\theta_i), \ i = 1, \cdots, M, \ (17)$$

where, $F(\theta_i) = F(\cdot; \theta_i)$ is a parametrized distribution for data $y_i$. Note that the parameters with which an observation is associated implicitly partitions the data into different clusters. Thus, one does not have to assume a priori the number of clusters in the data, allowing the data to drive the complexity of the inferred model. One can also obtain DP mixture model as the weak infinite limit (i.e. $K \to \infty$) of a sequence of finite mixture models (with $K < \infty$ clusters)

$$p(\alpha) \sim \text{Dir}(\alpha/K, \cdots, \alpha/K)$$

$$z_i|p \sim \text{Mult}(p), \ i = 1, \cdots, M$$

$$\theta_k \sim H, \ k = 1, \cdots, K$$

$$y_i|z_i, \{\theta_j\}_{j=1}^K \sim F(\theta_{z_i}), \ i = 1, \cdots, M, \ (18)$$

where, Dir is the standard Dirichlet distribution with concentration parameter $\alpha/K, p = \{p_1, \cdots, p_K\}$ is the mixing proportion for the clusters, Mult is the standard Multinomial distribution, and $z_i \in \{1, \cdots, K\}$ is the cluster assignment of $y_i$ such that $y_i$ is assigned to cluster $k$ when $z_i = k$.

**B. NP-hBIRL Generative Model and Inference**

We will use DP as a generative model for the behavior data $D^b$ similar to approach used in [18]. The generative process is as follows:

- Sample $C = \{c_1, \cdots, c_M\}$ from DP mixture model,

$$p(\alpha) \sim \text{Dir}(\alpha/K, \cdots, \alpha/K)$$

$$c_i|p \sim \text{Mult}(p), \ i = 1, \cdots, M$$

- Sample $\Theta = \{\theta_1, \cdots, \theta_K\}$ from prior $H(\cdot; \lambda)$

- Draw state and observation pairs $(s_{0:T}^i, y_{0:T}^i)$ from the HMM $\mathcal{H}_{\theta_i}$, for $i = 1, \cdots, M$ and set

$$D^b = \{y_{0:T}^i|i = 1, \cdots, M\},$$

$$S = \{s_{0:T}^i|i = 1, \cdots, M\}.$$

For $C$, we will denote by $C_{-m}$ as

$$C_{-m} = \{c_i|i \neq m, i = 1, \cdots, M\}, \ (19)$$

with similar interpretation for $S_{-m}$ and $\Theta_{-m}$. The joint posterior can be expressed as:

$$P(c, \Theta, S|D^b, \lambda, \alpha) \propto P(c|\alpha) \prod_{k=1}^K P(\theta_k, S_k|D_k^b, \lambda) \ (20)$$

where,

$$D_k^b = \{y_{0:T}^i \in D^b|c_i = k, i = 1, \cdots, M\}, \ (21)$$

is the set of trajectories assigned to the $k$–th cluster, and

$$S_k = \{s_{0:T}^i \in S|c_i = k, i = 1, \cdots, M\}, \ (22)$$

are the corresponding state sequences. The conditional marginals can be derived as follows:

- For $c_m$, the marginal is of the form

$$P(c_m|C_{-m}, D^b, \Theta, S, \lambda, \alpha) \propto P(y^m_{0:T_m}, s^m_{0:T_m}|c_m) P(c_m|C_{-m}, \alpha),$$

where,

$$P(c|C_{-m}, \alpha) = \left\{ \begin{array}{ll}
\frac{n_{-m,c_i}}{n_c c = c_j \text{ for some } j} \\
\alpha c \neq c_j \text{ for all } j
\end{array} \right. \ (23)$$

is the Chinese Restaurant Process (CRP) representation of the DP mixture model. Here $n_{-m,c_i} = |\{c_i = c_j|i \neq m, i = 1, \cdots, M\}|$ is number of trajectories excluding $y_{0:T}^j$ assigned to cluster $c_j$. Note that in Eq. (23), if the sampled $c \neq c_j \forall j = 1, \cdots, M$ then $y^m_{0:T_m}$ is assigned to a new cluster.

- For $\theta_k, k = 1, \cdots, K$, the marginals are given by

$$P(\theta_k|c, D^b, \Theta_{-k}, S, \lambda, \alpha) = P(\theta_k|D_k^b, S_k, \lambda), \ (24)$$

$$= P(\theta_k|S_k, \lambda). \ (25)$$

- For $s^m_{0:T_m}$, $m = 1 \cdots, M$, marginals become

$$P(s^m_{0:T_m}|c, D^b, \Theta, S, \lambda, \alpha) = P(s^m_{0:T_m}|y^m_{0:T_m}, \theta_{c_m}).$$

Using the above marginals, posterior samples of cluster indices and reward parameters can be iteratively sampled as described in the Algo. 6 which we refer to as nonparametric-hBIRL (NP-hBIRL).

**VI. BEHAVIOR PREDICTION AND CLASSIFICATION WITH hMDPs**

In this section we describe a framework to predict and classify agent behavior. Let $B_i, i = 1, \cdots, N_b$ be a class of different behaviors, each represented by a corresponding hMDP model $M_i^b; i = 1, \cdots, N_b$. We assume that each agent follows one of the behaviors $B_i$, which is unknown with a prior $P(B_i), i = 1, \cdots, N_b$. Given a noisy trajectory $y_t = y_{0:t}$ for an agent, the goal is to determine which behavior the agent is most likely following, and predict agent’s future trajectory. We take a Bayesian viewpoint, and describe procedures to determine the posterior $P(B_i|Y_t)$ and predict agent’s expected future trajectory.
Algorithm 6 NP-hBIRL

1: Let $C$, $\Theta$ and $S$ be samples from previous step.
2: For $m = 1$ to $M$
   • Let $c_m$ be current cluster of trajectory $y_{0:T_m}^m$, with
     $\theta_{c_m}$ being the reward vector for that cluster.
   • Sample $c_{m}^{*} \sim P(c|C_{-m}, \alpha)$ using (23).
   • If $c_{m}^{*} \notin C_{-m}$, then sample new $\theta_{c_{m}^{*}} \sim H(\cdot; \lambda)$ and
     the corresponding state sequence,
     \[
     s_{0:T_m}^m = \text{SampleStateSeq}(\theta_{c_{m}^{*}}, M^h_0|y_{0:T_m}^m),
     \]
   • Accept $(c_m, \theta_{c_m}) \leftarrow (c_{m}^{*}, \theta_{c_{m}^{*}})$ with probability
     \[
     \min \left(1, \frac{P(s_{0:T_m}^m, y_{0:T_m}^m|\theta_{c_{m}^{*}})}{P(s_{0:T_m}^m, y_{0:T_m}^m|\theta_{c_m})} \right)
     \]
     where, $P(s_{0:T_m}^m, y_{0:T_m}^m|\theta_{c_{m}^{*}})$ is computed using Eq. (9).
3: Using $\Theta$ from previous sampling step, obtain new samples
   \[\theta_k = \text{SampleTheta}(\theta_k, M^h_\theta, H, \tau, S_k),\]
   for $k = 1$ to $K$.
4: For $m = 1, \cdots, M$, sample new state sequences
   \[s_{0:T_m}^m = \text{SampleStateSeq}(\theta_{c_m}, M^h_{\theta}, y_{0:T_m}^m)\]
5: Return new samples of $C$, $\Theta$ and $S$.

A. Classification

For behavior classification, the posterior $P(B_i|Y_t)$ can be computed using Bayes rule,
\[
P(B_i|Y_t) = \frac{P(Y_t|B_i)P(B_i)}{\sum_{j=1}^{N_b} P(Y_t|B_j)P(B_j)},
\]  \hspace{1cm} (25)
where, $P(Y_t|B_i) = \sum_s \alpha^i_t(s)$ and $\alpha^i_t(s)$ is the forward variable for the model $M^h_i$ computed using recursion described in the Section IV-A.

B. Prediction

Let $\mathcal{O}(s|Y_t)$ be expected future occupancy map,
\[
\mathcal{O}(s|Y_t) = E_{P(s_{t+1:T}|Y_t)} \left[ \sum_{\tau=t+1}^{T} \mathcal{I}(s_\tau = s)|Y_t \right],
\]
which measures the expected number of times agent visits different states $s \in S$ over the horizon $[t+1, T]$. In order to compute $\mathcal{O}(s|Y_t)$, we condition on the latent behaviors $B_i$, $i = 1, \cdots, N_b$, as follows
\[
\mathcal{O}(s|Y_t) = \sum_{s_{t+1:T}} \sum_{\tau=t+1}^{T} \mathcal{I}(s_\tau = s)P(s_{t+1:T}|Y_t),
\]
\[
= \sum_{i=1}^{N_b} \sum_{s_{t+1:T}} \sum_{\tau=t+1}^{T} \mathcal{I}(s_\tau = s)P(s_{t+1:T}|Y_t, B_i)P(B_i|Y_t),
\]
\[
= \sum_{i=1}^{N_b} \sum_{\tau=t+1}^{T} \sum_{s_{t+1:T}} P(s_\tau = s|Y_t, B_i)P(B_i|Y_t),
\]
where, $P(B_i|Y_t)$ is computed using Eq. (25). To compute $P(s_\tau = s|Y_t, B_i) = P(s_\tau = s|Y_t, B_i)$ $\tau = t+2, \cdots, T$ (where we suppress dependence on the model $B_i$), we define
\[
\gamma_t(s_\tau) = P(s_\tau|Y_t) = \sum_{\tau-1} P(s_{\tau-1}, s_\tau|Y_t)
\]
\[
= \sum_{\tau-1} P_{\pi_0}(s_{\tau-1}, s_\tau)\gamma_{\tau-1}(s_{\tau-1}),
\]
with the initialization
\[
\gamma_{t+1}(s_{t+1}) = \sum_{s_{t}} P_{\pi_0}(s_t, s_{t+1})\alpha_t(s_t).
\]  \hspace{1cm} (27)

VII. NUMERICAL DEMONSTRATION

In this section, we demonstrate our hMDP based Bayesian nonparametric behavior learning, prediction and classification in a simulated urban surveillance scenario. In this scenario the goal-oriented agents move around to reach their desired destinations near the buildings, denoted by A, B and C in the Figure 1 a). In order to generate behavioral data, we resorted to Agent Based Modeling (ABM) [32], as it provides a flexible framework to emulate behaviors in a controlled setting. We developed an ABM simulator for our urban scenario. The urban environment is represented in form of a discretized elevation map indicating the computational cells occupied by the buildings. Agent state is represented by the cell it occupies, and there are 4 available actions: move north, south, east and west. The reward function for
each agent is parameterized in terms of four basis functions: \( \phi_i, i = 1, \ldots, 4 \), with \( \phi_1 \) and \( \phi_2 \) shown in the Figure 1 a). The function \( \phi_1 \) penalizes areas occupied with buildings which need to be avoided, while \( \phi_2, \phi_3, \phi_4 \) have high values near the goal destinations A, B and C, respectively. Accordingly, we assume there are 3 types of agent behaviors: \( \text{BehA} \), \( \text{BehB} \) and \( \text{BehC} \). Here, \( \text{BehA} \) denotes the behavior where agent has preference for destination A, with similar interpretation of \( \text{BehB} \) and \( \text{BehC} \).

Based on these reward preferences we compute optimal policies and generate noisy trajectories for each type of behavior. The observation model \( \mathcal{P}^y \) is taken be Gaussian distribution, i.e. \( \mathcal{P}^y(s, y) = \mathcal{N}(y, \Sigma) \), where \( y = (x, y) \) is the observed noisy agent position, and \( \mathcal{N} \) is normal distribution with covariance \( \Sigma = \sigma^2 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \) and mean \( y = (\bar{x}, \bar{y}) \) which is the cartesian coordinate of the center of cell corresponding to the discrete cell \( s \). We consider two noise levels: \( \sigma^2 = 1 \) which we refer to as Low Noise case, and \( \sigma^2 = 10 \) which corresponds to High Noise case.

### A. Behavior Learning

We now demonstrate the performance of our NP-hBIRL Algo. 6 for learning hMDP models from the unlabeled trajectory data. The results provided are based on a training set comprising of 15 trajectories per behavior. Figure 1b) shows a subset of the data. Figures 2a) and b) show posterior samples from a MCMC run for the two noise levels. One can see 3 distinct clusters in parameter space \((\theta_2, \theta_3, \theta_4)\) corresponding to \( \text{BehA} \), \( \text{BehB} \) and \( \text{BehC} \). As expected, for the low noise case these clusters are more prominent, while for high noise case they become fuzzy.

![Figure 2](image)

**Fig. 2.** Subplots a) and b) show posterior samples of reward parameter (red triangles are true reward parameters) from a MCMC run for two different noise levels. Subplots c) and d) show the corresponding hamming distance.

Figures 2 c) and d) illustrates the MCMC convergence in terms of Hamming distance which measures the error between true labels and estimated trajectory labels. Hamming distance is computed using Munkres algorithm as discussed in [29]. Again, compared to the high noise case the convergence for low noise is better. We ran several MCMC runs with different initializations, and picked the posterior sample which had the highest likelihood as point estimates for the learned hMDP models. We next demonstrate how these learned hMDP models can be used for behavior prediction and classification using techniques described in Section VI.

### B. Behavior Prediction and Classification

Given the trajectories of the different agents which are updated over time, the goal is to detect which agents are most likely moving towards the critical destination B, and predict their most likely future paths towards it. We assume that the agents follow one of the behaviors \( \text{BehA} \), \( \text{BehB} \) or \( \text{BehC} \), which is a priori unknown.

We first consider an agent following the behavior \( \text{BehB} \), and perform prediction based on agent’s observed noisy trajectory under the high noise case. Figure 3 a)-d) show the prediction of expected future occupancy map (denoted by yellow) at different time instants. Initially there is ambiguity in what behavior the agent is following. Despite the noisy observations, as more trajectory is seen, we are able to correctly classify behavior to be of type \( \text{BehB} \) in much advance, and thus accurately predict the future behavior. Note that if the noisy trajectory was directly discretized into cells and used with a non-noisy MDP model representation, the predictions will be highly inaccurate since noise will lead to transitions which are not consistent with the state transition matrix.

![Figure 3](image)

**Fig. 3.** Behavior prediction based on latent goal detection for the high noise case. Blue curve is the true trajectory, while the green dashed curve is the observed noisy trajectory.

Finally, we consider several agents moving towards their goal location near different buildings, see left plot in the Figure 4. Here the objective is to classify which agents if following \( \text{BehB} \). The right sub-figure in 4 shows the likelihood of different agents heading towards B. Clearly, agents 1 and 2 shown by red/green tracks are classified correctly as moving towards B in much advance before they
behaviors actually reach that destination.

VIII. CONCLUSION

In this paper we developed a nonparametric Bayesian IRL framework based on hMDP representation to address unsupervised learning in noisy environments, and described several MCMC procedures to efficiently learn the posterior given the noisy behavior data. We demonstrated our approach for learning, prediction and classification of agent behaviors in a simulated surveillance scenario. In future we plan to evaluate our NP-hBIRL framework for activity learning and analysis in realistic video datasets, and extend our approach in context of much richer MDP representation such as ones developed in [9].

ACKNOWLEDGEMENTS

This research was supported by AFOSR contract FA9550-11-C-0100. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the AFOSR. The author will like to thank Kunal Srivastava for discussions and feedback.

REFERENCES