Computer Science and Artificial Intelligence Laboratory

Technical Report

MIT-CSAIL-TR-2018-015 May 17, 2018

Learning Models of Sequential Decision-Making without Complete State Specification using Bayesian Nonparametric Inference and Active Querying

Vaibhav V. Unhelkar and Julie A. Shah

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Learning Models of Sequential Decision-Making without Complete State Specification using Bayesian Nonparametric Inference and Active Querying

Vaibhav V. Unhelkar
Massachusetts Institute of Technology
Cambridge, MA 02139
unhelkar@csail.mit.edu

Julie A. Shah
Massachusetts Institute of Technology
Cambridge, MA 02139
julie_a_shah@csail.mit.edu

Abstract

Learning models of decision-making behavior during sequential tasks is useful across a variety of applications, including human-machine interaction. In this paper, we present an approach to learning such models within Markovian domains based on observing and querying a decision-making agent. In contrast to classical approaches to behavior learning, we do not assume complete knowledge of the state features that impact an agent's decisions. Using tools from Bayesian nonparametric inference and time series of agents' decisions, we first provide an inference algorithm to identify the presence of any unmodeled state features that impact decision making, as well as likely candidate models. In order to identify the best model among these candidates, we next provide an active querying approach that resolves model ambiguity by querying the decision maker. Results from our evaluations demonstrate that, using the proposed algorithms, an observer can identify the presence of latent state features, recover their dynamics, and estimate their impact on decisions during sequential tasks.

Keywords: Decision Making, Graphical Models, Human-AI Collaboration

1 Introduction

Knowledge of another agent's decision-making model is useful in a variety of scenarios: for example, in intelligent tutoring applications, human-robot teamwork, and generating transparent AI systems. However, such a model is typically difficult to observe and specify. In such cases, an estimate of the decision-making models can be obtained either through prior domain knowledge, observations of the agent's behavior, or some combination of both.

Here, we consider the problem of inferring an agent's decision-making model during sequential tasks, without complete specification of the features that impact that agent's decisions (i.e., state features). For ease of exposition throughout this paper, we refer to the agent whose behavior is being inferred as the *decision maker*, and the agent who seeks to infer a model for these decisions as the *observer*. For instance, within an intelligent tutoring application, the observer would be the AI tutor attempting to understand the rationale behind the decisions of a human student Aleven et al. (2015); Ramachandran et al. (2017). In the context of human-robot teamwork, the observer is a robot predicting the decisions of its human collaborators in order to improve the fluency of collaboration Thomaz et al. (2016). Lastly, in problems requiring explainable and transparent AI systems, the observer is an introspective agent attempting to generate an interpretable model of its own decisions Hayes and Shah (2017).

In the above examples, while the agent's decisions can usually be observed, the model used for decision making is unavailable. A common approach, then, is to specify via prior domain knowledge the

features (state vector) that impact the agent's decisions, and to recover the agent's policy (mapping from state to action) using data related to the agent's decisions Ziebart et al. (2008). This approach has proven successful across a variety of applications (e.g., see Argall et al. (2009)) in which the objective of the observer is prediction or imitation of the decision maker's behavior. However, this method may not recover the agent's true model variables (for instance, reward function), and requires a complete specification of the features that impact the agent's decisions.

In several applications, however (including those discussed above), a model of the agent's decision making is usually desired, and a *complete* specification of the features that impact the decisions may not be available. Hence, in this paper, we first provide an inference approach that utilizes observations of an agent's decisions and pre-specified (known) features to identify the presence of any unmodeled (latent) features that impact the agent's decisions. We assume a Markovian model of the decision maker's behavior, and use tools from Bayesian nonparametric inference to sample likely agent models and jointly infer the agent's latent state features, the state transition model, and the agent's policy.

While our sampling-based inference approach is able to identify the presence of latent states and generate samples of likely agent models, we observe in our evaluation that, in general, it is not possible to identify the correct model from the likely samples using only observations of agent's decisions. To alleviate this ambiguity, we provide an active learning approach that generates queries for the decision maker given a limited budget of queries and a set of candidate agent models. We demonstrate in simulation experiments that our approach can identify the presence of latent state features, recover their dynamics, and estimate their impact on decisions during sequential tasks.

2 Background

There has been significant interest in understanding and modeling the decision-making behavior of humans Newell et al. (1972); Kahneman (2003) and artificial agents Georgeff et al. (1998) alike. While various models for describing decision-making behavior exist, due to our focus on sequential decision-making tasks, we adopt a formalism inspired by Markov decision processes (MDPs) Puterman (2014).

Concisely, an MDP is defined by a state space S specifying a finite set of states s, an action space A specifying a finite set of actions a, a Markovian state transition function $T(s'|s,a)$, an immediate reward function $R(s,a,s')$, and a discount factor γ. An MDP agent is assumed to be optimizing a reward criteria – commonly, the discounted cumulative reward $\sum_{t=0}^{\infty}\gamma^t R(s_t,a_t,s_{t+1})$. The agent's decision-making behavior is summarized by its policy $\pi(a|s)$, which maps states to actions. The optimal decision-making behavior π^* can be obtained using planning (when model specifications are known) or reinforcement learning Sutton and Barto (1998).

In this paper, we consider scenarios wherein the decision-maker agent is operating within a Markovian world with known dynamics (state transition function) but an unspecified reward function (cf. the MDP/R notation used by Abbeel and Ng (2004)). In contrast to the canonical definition of MDP, the agent's reward function, reward optimization criteria (if any) and its policy are assumed to be unknown. In addition, some features that impact the agent's decisions are also assumed to be unknown: the agent's policy might include unmodeled dynamic features x, and is given by $\pi(a|s,x)$ (instead of $\pi(a|s)$). These differences are motivated by the facts that (a) human and artificial agents may not behave optimally Simon (1982), and (b) models generated using prior domain knowledge for describing behavior, though useful, might exclude some relevant factors that impact agent decisions. Next, we provide an example scenario motivating our problem, and mathematically formalize these aspects of our model and the problem in the subsequent section.

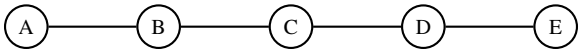

Figure 1: Schematic of a transit system. The nodes represent stations of the transit system.

Motivating example: Consider an agent using a public transit system in a city depicted in Fig. 1. Assuming trains moving in both east and west directions are always available, the set of decisions available to the agent is known and includes the following: taking a train moving east, taking a train moving west, or waiting at a station. While the dynamics of the agent's motion – which depend upon

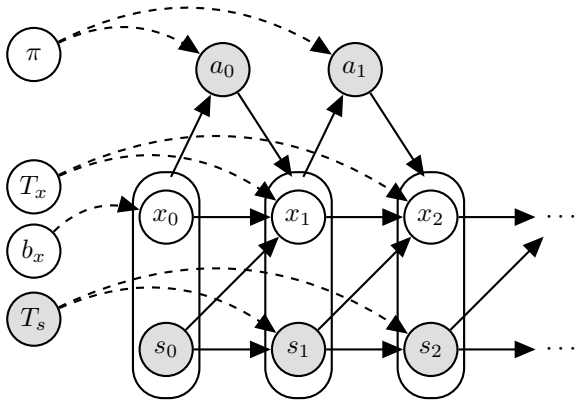

Figure 2: A graphical model of decision-making behavior. Each node represents a random variable, and the observed variables are depicted in grey. The known state features are represented by s, and the unknown state features by x. Both s and x (included in the oval supernode) impact the decision choices a, based on the policy $\pi(a|s, x)$. The transition functions T_x and T_s specify the probability for the next state given the previous state and the action, and the start distribution b_x models the probability of the initial latent state x_0.

the public transit system and the agent's decisions – are known, the rationale behind the agent's decisions is not. These decisions depend upon a variety of factors, including known, observable features (such as the current location) and unmodeled, latent features (such as the intended location). In this paper, we are interested in identifying the presence of any unmodeled features in an agent's decision making, their dynamics, and their impact on that agent's decisions.

3 iAMM: Bayesian Nonparametric Model for Markovian Decision-Making

In this section, we formalize our decision-making model and define the problems of interest. We consider a scenario in which an observer intends to model the behavior of a decision-maker agent. The decision maker's decisions depend upon state features $f \in F$. These features include known components denoted by $s \in S$ and unknown components denoted by $x \in X$ – i.e., $f = [x, s]$. The decision maker has full knowledge and observability of state f; however, the observer is only aware of the known component, s.

The state transition function for the known features $T_s(s'|s, a)$, which specifies the probability of the next observable state given the previous observable state and action, is known to the observer agent. However, the set of latent states X, their initial distribution $b_x(x)$, and their transition dynamics $T_x(x'|x, s, a)$ are unknown to the observer. Lastly, the decision maker's behavior is summarized by the policy $\pi(a|f) = \pi(a|x, s)$, which provides the probability of choosing action a while the agent is in state f. The decision maker has knowledge of all model variables, including X, b_x, T_x and π, while the observer does not.

We summarize this Markovian model of agent's decisions, denoted as AMM, as a graphical model in Fig. 2. In applications of interest, the unmodeled states could correspond to agent preferences (e.g., the goal location in the motivating example). The known state transition function T_s, given the agent's decision (action), is independent of the latent state features, and models the fact that latent states impact observable state dynamics only via actions. Note that this model does not make any assumptions regarding the reward function used by the agent or the reward optimization criteria; however, as we will address in Section 3.2, if this information is available, it can be incorporated as probability priors in our model.

3.1 Problem Definition

Formally, we consider the following two problems for the observer agent to characterize the behavior of the decision-maker agent:

1. Given execution traces – i.e., time series of known, observable state features s and actions (decisions) a, – identify the presence of any latent state features x.

2. If latent states exist, estimate
   (a) their transition dynamics T_x and
   (b) the state-dependent policy $\pi(a|s, x)$.

3.2 Priors for Latent State Features

In order to model the presence of latent state features and to incorporate any available prior knowledge, we employ tools from Bayesian modeling. Specifically, we extend our model with priors for latent state features, their dynamics, and impact on the decision-maker's policy to arrive at a generative model of decisions with Bayesian nonparameteric priors.

As the set of latent states X is unknown a priori, the number of latent states is also unknown and must be inferred from the execution traces of the decision maker. This naturally motivates the use of nonparametric priors for number of states $|X|$, initial distribution b_x and transition function T_x. Specifically, we use hierarchical Dirichlet process (HDP) priors inspired by the HDP-HMM model of Teh et al. (2006). Under this HDP prior, the popularity of the latent states is generated from a Dirichlet process (DP). The base distribution over possible latent states, given by β, can be obtained using a stick-breaking construction with hyper-parameter γ, i.e.,

$$\beta(\cdot) \sim \text{GEM}(\gamma) \tag{1}$$

The initial distribution b_x and the rows of the transition function T_x are also generated using a Dirichlet process (DP) with base distribution β and scaling parameter α. This allows us to share parameters (in our case, the state space of latent states) between the rows of transition function and the initial belief. The corresponding generative model is given as follows:

$$b_x(\cdot) \sim \text{DP}(\alpha, \beta) \tag{2}$$

$$T_x(\cdot|x, s, a) \sim \text{DP}(\alpha, \beta), \qquad x = 1, 2, \cdots \tag{3}$$

In case the hyper-parameters α and γ also need to be estimated, $Gamma$ priors are used for these variables.

3.3 Priors for Decision-Maker's Policy

In addition to priors for latent states and their dynamics, our model includes a prior for the latent state-dependent policy $\pi(a|s, x)$. This prior is modeled as a probability distribution over the decision maker's action space and is identical for each of the potentially countably infinite latent states x. This policy prior allows us to incorporate prior domain knowledge (if any) regarding the agent's policy and can vary based on the known state feature s. For our experiments, unless otherwise specified, we used a Dirichlet distribution as the policy prior,

$$\pi(a|s, x) \sim \textsc{Dirichlet}(\rho_{1:|A|}), \qquad x = 1, 2, \cdots \tag{4}$$

The graphical model of Fig. 2 along with the priors specified in Eq.1-4 describes our model denoted as iAMM, a Bayesian nonparametric generative model of decision making within Markovian domains.

4 Identifying Presence of Latent States

We first address the problem of identifying presence of any latent states that impact agent decisions. With regard to our generative model iAMM, this problem corresponds to inference of the latent state space X given the known model parameters S, A, T_s and decision-maker's execution traces $a_{0:T}, s_{0:T}$. Both sampling based and variational inference approaches can be developed for this inference problem. In this paper, we provide a sampling based approach - namely, beam sampling for iAMM - inspired by a similar algorithm for HDP-HMM developed by Van Gael et al. (2008). Along with identifying latent states, the beam sampling approach jointly infers other latent variables, and consequently provides samples of likely candidate models of decision-making.

4.1 Beam Sampling for iAMM

The beam-sampling algorithm for iAMM is summarized in Algorithm 1. This algorithm is a variant of Gibbs sampling, in which state sequences are sampled as a single block; this improves mixing time, since correlations between successive states are accounted for. The sampling of entire state sequences at once is achieved via additional auxiliary variables and the use of dynamic programming. While the structure of the algorithm is similar to that designed by Van Gael et al. (2008), due to a different graphical model structure, conditional distributions differ, which we detail below.

Algorithm 1: Beam Sampling for iAMM

Data: $a^i_{0:N_i}$, $s^i_{0:N_i}$, $\quad i = 1, 2, \cdots, I$
Parameters: α, γ, ρ, N_m
Result: Samples for $\mathbf{x}^{1:I}$, π, T_x, b_x, β
Initialize $\mathbf{x}^{1:I}$, β randomly
while number of samples generated $< N_m$ **do**
  Sample initial distribution, $Pr(b_x|\mathbf{x}^{1:I}, \text{data}, \beta, \alpha)$
  Sample transition function, $Pr(T_x|\mathbf{x}^{1:I}, \text{data}, \beta, \alpha)$
  Sample policy, $Pr(\pi|\mathbf{x}^{1:I}, \text{data}, \rho)$
  Sample auxiliary variables for beam sampling, $Pr(\mathbf{u}^{1:I}|\mathbf{x}^{1:I}, \text{ data}, T_x, b_x)$
  Sample latent states, $Pr(\mathbf{x}^{1:I}|\text{data}, T_x, b_x, \pi, \beta)$
  Sample base distribution, $Pr(\beta|\mathbf{x}^{1:I}, \alpha, \gamma)$
  (optional) Sample hyper-parameters α and γ
end

Data and known parameters: The input to the algorithm is I execution traces from the decision maker of potentially different lengths N_i; the i-th trace is denoted as $a^i_{0:N_i}, s^i_{0:N_i}$ – or, equivalently, $\mathbf{a}^i, \mathbf{s}^i$. The superscripts denote the i-th execution, and subscripts denote the time indices within each time series. The algorithm also has access to known parameters: namely, the parameters for policy prior ρ, hyper-parameters, and number of desired model samples N_m.

Sampling auxiliary variables, u: Beam sampling is made possible by sampling additional auxiliary variables $\mathbf{u}^{1:I}$, with the following conditional distributions

$$u^i_0 \sim \textsc{Uniform}(0, b_x(x^i_0)) \tag{5a}$$

$$u^i_t \sim \textsc{Uniform}(0, T_x(x^i_{t+1}|x^i_t, s^i_t, a^i_t)) \tag{5b}$$

Sampling model variables: The symbol M is used to describe model variables T_x, b_x, π. Conditional distributions for model variables T_x, b_x and base distribution β follow from sampling of transition function and base distribution in beam sampling for HDP-HMM, but requires additional bookkeeping due to the presence of multiple sequences and an unknown initial latent state. For instance, for sampling T_x, we define n_{ijsa} as the count of transition from latent state i to latent state j for action a and observed state s. Both i and j range from 1 to K, where K is the number of distinct states in the sample of latent states $\mathbf{x}^{1:I}$. The conditional distribution of $T_x(\cdot|x, s, a)$ given remaining variables is as follows: $\textsc{Dirichlet}(n_{x1sa} + \alpha\beta_1, n_{x2sa} + \alpha\beta_2, \cdots, n_{xKsa} + \alpha\beta_K, \alpha(1 - \sum_{i=1}^K \beta_i))$. The sampling of initial and base distribution follows similarly.

Conditional distributions for π depend upon the policy prior. For the policy prior of Eq. 4, this distribution is given by $\pi(\cdot|x, s) \sim \textsc{Dirichlet}(\rho_{1:|A|} + \bar{n}_{ixs})$ — where the vector $\bar{n}_{ixs}$ is the count of action decisions i in latent state x and observed state s, and is computed using sampled latent state sequences $\mathbf{x}^{1:I}$ and the execution traces.

Sampling state sequences, $\mathbf{x}$: Latent state sequences are sampled using a variant of forward filtering-backward sampling (FFBS). However, since potentially countably infinite latent states exist, an approach to limit the trajectory space is necessary: auxiliary variables are used to create a finite partition of likely trajectories and to decide whether to add or remove latent states. In the forward pass of FFBS, the filtered estimates $Pr(\hat{x}^i_t) \equiv Pr(x^i_t|u^i_{0:t}, s^i_{0:t}, a^i_{0:t}, M)$ are computed iteratively for each

time index t of each trajectory i:

$$Pr(\hat{x}_0^i) \sim \pi(a_0^i|x_0^i, s_0^i)\ \mathbb{1}(u_0^i < b_x(x_0^i)) \tag{6a}$$

$$Pr(\hat{x}_t^i) \sim \pi(a_0^i|x_0^i, s_0^i) \cdot \sum_{x_{t-1}^i} \{Pr(\hat{x}_t^i)\mathbb{1}(u_t^i < T_x(x_t^i|x_{t-1}^i, s_{t-1}^i, a_{t-1}^i))\}. \tag{6b}$$

In the backward pass, first the state at the final time index $x_{N_i}^i$ is sampled using the $Pr(\hat{x}_{N_i}^i)$ computed in the forward pass. The remaining sequence is then sampled iteratively using the following,

$$x_t^i \sim Pr(x_t^i|x_{t+1}^i, u_{0:N_i}^i, M, \text{data}) \tag{7a}$$

$$= Pr(\hat{x}_t^i)\ \mathbb{1}(u_{t+1}^i < T_x(x_{t+1}^i|x_t^i, s_t^i, a_t^i)). \tag{7b}$$

Sampling hyper-parameters, α, γ: Lastly, the values of hyper-parameters α, γ may be set, or otherwise inferred using Gamma priors Escobar and West (1995).

4.2 Presence of Latent States

Algorithm 1 generates samples of models and latent state sequences. These inferred quantities essentially segment the observed trajectories and assign these segments to different clusters (latent states) corresponding to different unmodeled behaviors $\pi(a|x, s)$. In case only one cluster exists in the generated samples $\mathbf{x}^{1:I}$, our algorithm indicates that no unmodeled latent states exist – since, the decision maker's policy is invariant of the latent state x. In addition to segmentation and clustering, the beam sampler also provides candidate models for how the hidden states change during execution and impact action selection via samples of transition dynamics of hidden states and the state-dependent policy.

4.3 Challenges for Estimating Model Variables

As demonstrated in Section 6, the beam sampler-based approach is able to identify the presence of latent state features when they exist, and provides a solution for Problem 1 of Section 3.1. The sampler is also capable of generating samples of model variables $M \equiv (\pi, T_x, b_x)$. However, in our experiments, we observed that even though latent states were identified, the generated samples of M generally differed from the true model variables. Here, we discuss the cause behind this mismatch in true and sampled model variables, and then provide a solution for Problem 2.

Claim 1. *Two different models, M_1 and M_2, can result in identical execution traces.*

We demonstrate Claim 1 via an example. Consider two decision-maker agents, Alice and Bob, traversing the transit system depicted in Fig. 1. Alice has two policies corresponding to two latent preferences: traveling east for work (latent goal A) and traveling west for home (latent goal E). Bob also has two policies: traveling inbound toward the city center (C) and traveling outbound away from the center. If both the agents begin at point E, with Alice's plan being "travel to work" and Bob's plan being "travel inbound to C, and then outbound to A," the execution traces for each agent will be identical, while the hidden states (preferences) and the model variables (π, T_x) will be different. Hence, as a consequence of Claim 1, we can explain the mismatch between true models and those inferred using Algorithm 1. Claim 1 further implies that execution traces are insufficient to disambiguate two models; additional information is needed to identify the decision maker's true model. Next, we propose an approach in which the observer actively queries the decision maker to identify the decision maker's model from a set of candidate models. Since the labels for hidden states used by the decision maker might differ from those of the observer, any queries for the decision maker should be limited to observed state features s and actions a.

5 Identifying State Dynamics and Policy

We next explore the use of queries and provide an active querying approach to identify the model variables M, which include latent state dynamics T_x, of the decision maker.

Query for the decision maker: Queries for the decision maker can only depend on the observed variables – i.e., decisions a and known state features s – not on the specific labels of the latent state

x. Consequently, in our approach we consider queries of the type, "Can the latent state change when action a is chosen in the observable state feature s?" This query is motivated by the heuristic that the number of latent states (hidden preferences or goals) are typically fewer than the observed state features, s, and thus change less frequently over the course of a task execution.

Input: The available input for identifying the true model variables is a set of J likely candidate models M_j. These candidate models can be generated using Algorithm 1. In addition, N_q specifies the limited number of queries that the observer can make to the decision maker.

Objective: Select N_q queries to identify the most likely model from the set of J candidate models.

5.1 Bayesian Approach to Query Selection

We adopt a Bayesian approach to identify the most informative queries. The approach begins by defining a prior $Pr(M_j)$ over the available models, and quantifies the initial belief regarding which candidate model is most likely. The normalized likelihood of each model $Pr(\text{data}|M_j)/I$ is used as its initial score $\texttt{score}_j$, and used to arrive at the prior probability $Pr(M_j) \propto \texttt{score}_j$.

Next, for each model M_j, the approach determines the set of potential queries Q_j by identifying (s, a)-pairs for which transition between hidden states x is possible. To identify the possibility of hidden state transition for a given (s, a)-pair for each model, the approach first computes the KL divergence of the rows of the transition matrix T_x with respect to the rows of an identity matrix. The identity matrix corresponds to a transition matrix describing only self-transitions, and is chosen based on the previously mentioned heuristic. If the KL divergence score for a (s, a)-pair exceeds a pre-specified threshold ζ_{KL}, then the pair is added to Q_j. The set of all queries is then given as $Q = \cup_j Q_j$.

To compute the most informative query from Q, the approach selects the query q^* whose response $\texttt{r}(q_k)$ in expectation results in the least conditional entropy $H(M|q_k)$. The response to a query $\texttt{r}(q_k)$ can either be "yes" or "no," and has conditional distribution $Pr(\texttt{r}(q_k)|M_j) = \mathbb{1}[(q_k \in Q_j) = \texttt{r}(q_k)]$. The conditional entropy for a given query q_k is given as follows:

$$H(M|q_k) = -\sum_{\texttt{r}(q_k)} \sum_{M_j} Pr(M_j, \texttt{r}(q_k)) \ln Pr(M_j|\texttt{r}(q_k)). \tag{8}$$

The most informative query is then determined as follows: $q^* = \arg\min_{q_k} H(M|q_k)$.

Next, the approach updates the posterior belief over models $Pr(M_j|\texttt{r}(q_k))$ based on the query response. The above process, using the updated belief, is repeated until all N_q queries are generated. The most likely model is selected as the mode of the final posterior distribution over model samples. It is possible that the true model does not exist in the generated set of likely candidate models M_j. Such a situation can be identified if the response to a query results in empty support of the posterior. In this case, the approach reinitializes the beam sampler using the constraints determined by the querying system and generates an alternate set of candidate models for querying. While we focused on a specific query type, other types of queries could also be addressed using this approach, provided their impact on entropy $H(M|q)$ can be computed.

6 Experiments

We examined the performance of our approach using synthetic data, which allowed us to evaluate the discrepancy between the true and estimated models. For each of the evaluation scenarios, we first described the generator – the true model used to generate the execution traces. Next, using Algorithm 1, we identified the presence of latent state features, if any. Lastly, we determined the most-likely model using the active querying approach, and compared its performance to that of a baseline of randomly generated queries.

Simulation parameters: Identical Gamma priors were used to infer the hyper-parameters $\alpha \sim \text{Gamma}(4, 10)$ and $\gamma \sim \text{Gamma}(2, 1)$ for all scenarios. A weak prior was used for γ to model unknown number of latent states, and the α prior was chosen to emphasize unimodal distributions as the transition function T_x rows. To estimate the policy π, all actions were modelled equally likely via the prior, i.e., ρ_a=0.1 $\forall\ a \in A$. We used a burn-in of 2000 samples and selected every 400^{th} sample to generate a set of 20 candidate models using Algorithm 1. For querying, the sets Q_j were

computed using ζ_{KL}=$-\ln(0.9)$, and only for those (s, a)-pairs present in the data. We used difference in the query set of true and estimated models as the metric for model accuracy. The Wilcoxon signed-rank test was used to conduct the statistical tests.

6.1 Public Transit System

The public transit system example described in Fig. 1 served as the first evaluation scenario. We generated execution traces by simulating trajectories for an agent navigating the transit system, with a list of stations it needed to visit. For each execution trace, the number of stations (goals) ranged from two to three, and the goals included either stations A or E. The known feature s was the current location of the agent, while the behavior of the agent also depended upon the unmodeled feature x – its intended goal. The transition dynamics of known state s, given current location and action, were deterministic. We conducted ten simulation trials, with ten execution traces generated for each trial. The scenario was challenging since the data included different actions for the same location s, and cyclic trajectories.

The sampler generated candidate models with more than one latent state for all trials – confirming the presence of unmodeled state features. Next, to assess the performance of active querying separate from inference, we computed the number of queries required to arrive at the true model. For fair basis of comparison, we ensured the true model was included as one of the candidate models. Over a set of ten simulations, our active querying approach required on average 1.7 queries to identify the true model as compared to 3.0 queries required by the baseline approach. This difference is both statistically significant ($p < 0.05$) and meaningful for applications that require the judicious utilization of human input.

Lastly, we computed the most likely model using our integrated sampler and active querying system and employing a query budget of $N_q = 4$. A new set of candidate models was generated after each query response using the beam sampler and the constraints learnt based on the response. The difference between the true and estimated model using our interleaved querying and sampling approach was 1.6 (s, a)-pairs on average, and lower ($p < 0.01$) than 4.7 (s, a)-pairs, the difference to the most likely model generated by the sampler without querying – thereby demonstrating the benefit of our integrated querying approach.

6.2 Collaborative Robot Behavior

Next, we simulated a scenario motivated by a collaborative robot supporting a human in a factory, and aimed to recover a model for robot's behavior. The robot helps carry a heavy part for a human between two depots (d_A and d_B) in a factory. Robot's known states include its location, and its actions allow for moving between these locations and staying idle. The human states are modeled as "requesting robot support". and "working independently," and evolve stochastically independent of robot's actions. The robot's policy depends on the human and robot's known states as well as latent states that model whether the robot is presently busy and whether it has correctly identified the human's request. The scenario assumes the transitions between latent states are stochastic to model the uncertainty that the robot identifies the human's request – thereby ensuring that the inference of robot's latent states and associated model variables is a challenging task.

We conducted ten simulation trials in the same manner as the first scenario. The beam sampler generated models with more than one latent state – correctly indicating presence of latent features in the behavioral data. The difference between the true and estimated model using the interleaved sampling and active querying approach was 4.1 on average, and lower ($p < 0.01$) as compared to estimates generated without querying, which resulted in an error of 7.1-(s, a) pairs. Although we set a query budget of $N_q = 4$, we observed that after two queries using our integrated system the discrepancy in model reduced by $> 40\%$.

6.3 Discussion

Our experiment results validated that Algorithm 1 was capable of identifying presence of latent states. Our method was also able to generate potential (s, a)-pairs that induce change in the latent state x. This was made possible by modeling latent state transitions to depend on both known state s and actions a. While the sampler was capable of generating estimates of model variables, we

observed that execution traces alone were not sufficient to identify the most accurate model among the samples. Similar ambiguity in estimating true model parameters arises in related problems, e.g., estimating the true reward function in IRL. As our objective was to recover the true model, we employed an active querying approach in conjunction with inference.

Our evaluation results indicated that our query selection approach was capable of reducing model uncertainty, and required fewer queries to identify the true model as compared to the baseline. This result is especially promising when the observer is learning models for human-AI collaboration, and might want to minimize the number of queries to a human (or an oracle) for learning the sequential decision-making model. The form of our query is readily posed to humans, since it is based on only the "common ground" of observed features and actions and is independent of the labels of the latent states.

Our approach is limited in that it can only identify the presence of latent states and corresponding model variables, but not the semantic labels or interpretation of these latent states. Further, if two preferences or latent states correspond to identical policy, the inference approach will collapse them into a single latent state. Our approach currently utilizes a flat state representation for the latent state. Lastly, we note the query used in our evaluations is especially useful for systems where both s and a impact transitions of latent states; however, other query types also need to be explored. We aim in future work to apply our approach to human-in-the-loop model learning, and to address these limitations through use of function approximation to represent model variables and factored latent state representations.

7 Related Work

Behavioral models of human and artificial agents are required for successful human-AI collaboration. Recent research into algorithmic human-robot interaction has provided examples of employing (variants of) inverse reinforcement learning (IRL) to estimate humans' policy during sequential tasks Sadigh et al. (2016); Majumdar et al. (2017) – albeit with complete specification of state features available. However, it is often difficult to pre-specify latent causes of decision making (e.g., see research in Gershman et al. (2015)).

Teh et al. (2006) provided HDP-HMM, a Bayesian nonparametric Hidden Markov Model for scenarios in which the latent states are unknown. Several extensions of HDP-HMM have since been developed, including hierarchical and semi-Markovian variants Saeedi et al. (2016); Johnson and Willsky (2013); however, these approaches do not explicitly model agent policy or decision making.

Bayesian nonparametric extensions of decision-making models, both for planning Doshi-Velez et al. (2015); Liu et al. (2011) and IRL Michini and How (2012); Ranchod et al. (2015), have also been developed. The nonparametric IRL approaches use the same input data as our approach, and aim to recover the decision maker's latent state dependent reward/policy, resulting in better performance than parametric IRL approaches when complete state specification is unavailable. However, these approaches assume that transitions between latent states are independent of action or known state features, and may not recover the true transition function of the latent states. In contrast, our approach allows for latent state transitions to depend upon known state features and actions, and incorporates a querying approach to identify the true transition function of the latent states.

Lastly, while active learning approaches for learning MDP parameters exist, they assume complete knowledge of the state features and focus on a different problem setting: action selection to reduce uncertainty over MDP parameters Araya-López et al. (2011); Chen and Nielsen (2012). In contrast, for our problem, complete knowledge of the decision maker's state features is not assumed, and the actions of the decision maker are available as input data and cannot be modified by the observer.

8 Conclusion

We present a novel approach to learn models of sequential decision making from sequences of decision-maker agent's decisions and partial state features. We first developed a Bayesian nonparametric inference scheme to identify the presence of any unmodeled (latent) state features that influence the agent's decisions, and generate samples of candidate decision-making models. We also provided an active querying approach designed to identify the most accurate model from a set

of candidate models given a limited budget of queries. We then demonstrated, using synthetic data, that our inference technique, coupled with the active querying approach, is capable of estimating the decision maker's model.

References

Abbeel, P. and Ng, A. Y. (2004). Apprenticeship learning via inverse reinforcement learning. In *Intl. Conf. on Machine Learning (ICML)*. ACM.

Aleven, V., Sewall, J., et al. (2015). The beginning of a beautiful friendship? Intelligent tutoring systems and MOOCs. In *Intl. Conf. on Artificial Intelligence in Education*. Springer.

Araya-López, M., Buffet, O., Thomas, V., and Charpillet, F. (2011). Active learning of MDP models. In *European Workshop on Reinforcement Learning*, pages 42–53. Springer.

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

Chen, Y. and Nielsen, T. D. (2012). Active learning of Markov decision processes for system verification. In *Intl. Conf. on Machine Learning and Applications (ICMLA)*. IEEE.

Doshi-Velez, F., Pfau, D., Wood, F., and Roy, N. (2015). Bayesian nonparametric methods for partially-observable reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 37(2):394–407.

Escobar, M. D. and West, M. (1995). Bayesian density estimation and inference using mixtures. *Journal of the American Statistical Association*, 90(430):577–588.

Georgeff, M., Pell, B., Pollack, M., Tambe, M., and Wooldridge, M. (1998). The belief-desire-intention model of agency. In *Intl. Workshop on Agent Theories, Architectures, and Languages*, pages 1–10. Springer.

Gershman, S. J., Norman, K. A., and Niv, Y. (2015). Discovering latent causes in reinforcement learning. *Current Opinion in Behavioral Sciences*, 5:43–50.

Hayes, B. and Shah, J. A. (2017). Improving robot controller transparency through autonomous policy explanation. In *Intl. Conf. on Human-Robot Interaction (HRI)*, pages 303–312. ACM.

Johnson, M. J. and Willsky, A. S. (2013). Bayesian nonparametric hidden semi-Markov models. *Journal of Machine Learning Research*, 14(Feb):673–701.

Kahneman, D. (2003). Maps of bounded rationality: Psychology for behavioral economics. *The American economic review*, 93(5):1449–1475.

Liu, M., Liao, X., and Carin, L. (2011). The infinite regionalized policy representation. In *Intl. Conf. on Machine Learning (ICML)*. ACM.

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

Michini, B. and How, J. (2012). Bayesian nonparametric inverse reinforcement learning. In *Joint European Conf. on Machine Learning and Knowledge Discovery in Databases*. Springer.

Newell, A., Simon, H. A., et al. (1972). *Human problem solving*, volume 104. Prentice-Hall Englewood Cliffs, NJ.

Puterman, M. L. (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Ramachandran, A., Huang, C.-M., and Scassellati, B. (2017). Give me a break!: Personalized timing strategies to promote learning in robot-child tutoring. In *Intl. Conf. on Human-Robot Interaction (HRI)*, pages 146–155. ACM.

Ranchod, P., Rosman, B., and Konidaris, G. (2015). Nonparametric bayesian reward segmentation for skill discovery using inverse reinforcement learning. In *Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 471–477. IEEE.

Sadigh, D., Sastry, S., Seshia, S. A., and Dragan, A. D. (2016). Planning for autonomous cars that leverages effects on human actions. In *Robotics: Science and Systems (R:SS)*.

Saeedi, A., Hoffman, M., Johnson, M., and Adams, R. (2016). The segmented iHMM: a simple, efficient hierarchical infinite HMM. In *Intl. Conf. on Machine Learning (ICML)*.

Simon, H. A. (1982). *Models of bounded rationality: Empirically grounded economic reason*, volume 3. MIT press.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Thomaz, A., Hoffman, G., Cakmak, M., et al. (2016). Computational human-robot interaction. *Foundations and Trends in Robotics*, 4(2-3):105–223.

Van Gael, J., Saatci, Y., Teh, Y. W., and Ghahramani, Z. (2008). Beam sampling for the infinite hidden Markov model. In *Intl. Conf. on Machine Learning (ICML)*, pages 1088–1095. ACM.

Ziebart, B. D., Maas, A. L., Bagnell, J. A., and Dey, A. K. (2008). Maximum Entropy Inverse Reinforcement Learning. In *AAAI*, volume 8, pages 1433–1438. Chicago, IL, USA.