# Semi-Supervised Learning of Decision-Making Models for Human-Robot Collaboration

**Vaibhav V. Unhelkar*** **Shen Li*** **Julie A. Shah**
Massachusetts Institute of Technology
Cambridge, MA, United States
{unhelkar,shenli,julie_a_shah}@csail.mit.edu

**Abstract:** We consider human-robot collaboration in sequential tasks with known task objectives. For interaction planning in this setting, the utility of models for *decision-making under uncertainty* has been demonstrated across domains. However, in practice, specifying the model parameters remains challenging, requiring significant effort from the robot developer. To alleviate this challenge, we present AdaCoRL, a framework to specify decision-making models and generate robot behavior for interaction. Central to our approach are a factored task model and a semi-supervised algorithm to learn models of human behavior. We demonstrate that our specification approach, despite significantly fewer labels, generates models (and policies) that perform equally well or better than models learned with supervised data. By leveraging pre-computed performance bounds and an online planner, AdaCoRL can generate robot behavior for collaborative tasks with large state spaces ($> 1$ million states) and short planning times ($< 0.5$ s).

**Keywords:** Human-in-the-Loop Learning, Planning under Uncertainty

## 1 Introduction

Planning robot actions for interacting with humans, or *interaction planning*, is essential to effective human-robot collaboration. The utility of interaction planning for improving both fluency and efficiency of human-robot collaboration has been demonstrated across a variety of collaborative tasks, including shared workspace, shared manipulation, and handover tasks [1, 2, 3, 4, 5]. Here, we consider the problem of interaction planning for a subclass of human-robot collaborative tasks, namely, sequential tasks with known task objectives.

Instantiations of this subclass of collaborative tasks are found across domains, including homes, offices, hospitals, and outer space. For instance, consider a robotic assistant tasked to support a human in the assembly of parts in a factory [6]. Assembly tasks typically require multiple steps and are, thus, sequential. Further, the human, the robot, and the robot developer (i.e., the person/s programming the robot) know the objective of the assembly task a priori. Other examples include, a robotic scrub nurse supporting a human surgeon [7] and a robot supporting astronauts [8].

Despite the knowledge of the task objective, manually hand-crafting robot behavior (i.e., policy) is difficult; since, among other factors, it requires the robot developer to specify a robot action for all possible interaction scenarios. To address this difficulty, researchers have developed algorithmic approaches to generate robot behavior [3, 6, 9, 10]. Broadly, these algorithmic approaches to interaction planning involve two steps, namely, specification of a robot's decision-making model and generation of the robot's policy through algorithms for decision-making.

Algorithmic computation of robot's policy speeds up the specification of robot behavior; however, in practice, specification of the decision-making models remains time-intensive. In the process of specifying the robot's decision-making model, the developer needs to specify models for both the collaborative task and human behavior. The task model corresponds to the specification of the task objective and dynamics (i.e., a model specifying the outcomes of human and robot actions). The model of human behavior allows for the prediction of human states and actions.

*These authors contributed equally to this work.

3rd Conference on Robot Learning (CoRL 2019), Osaka, Japan.

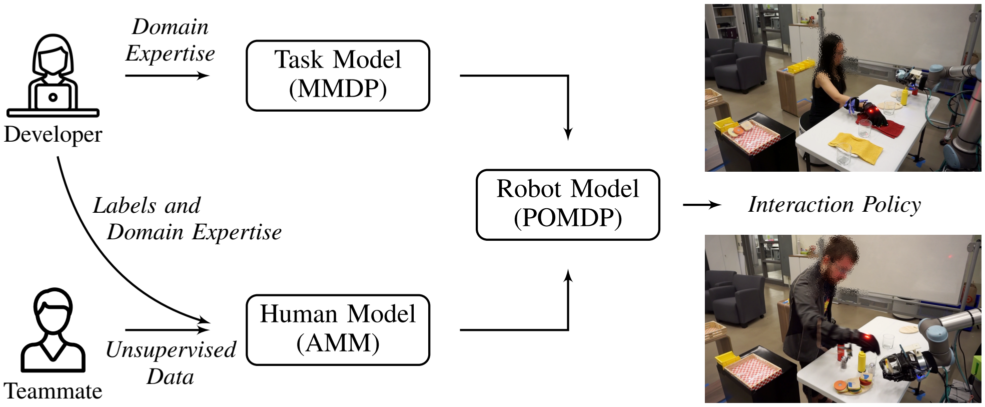

Figure 1: AdaCoRL: a model specification and interaction planning framework for human-robot collaboration. The developer specifies the collaborative task and provides partial specifications of human behavior. By combining data and domain expertise, our approach reduces the labeling effort required to specify human models. Given the specifications, both the robot's model and interaction policy are generated algorithmically. (right) Stills from demonstrations of our approach in two human-robot collaborative tasks: (top) a shared workspace task, (bottom) handover task.

To enable human-robot collaboration at scale, approaches that can accelerate this model specification process will prove to be critical. Here, we present such an approach that reduces the robot developer's effort while specifying models for interaction planning. As described next, our framework – titled Adaptive Collaboration with Reduced Labeling (AdaCoRL) – enables both semi-supervised model specification and execution-time planning for interaction (see Fig. 1).

**Specifying Task Model** Due to our focus on known tasks, the developer has the necessary domain expertise to specify the task model. However, the choice of model representation is critical to facilitating model specification. Thus, in AdaCoRL, we model the task as a multi-agent Markov decision process (MMDP) [11] and utilize a factored state representation, wherein the developer can specify the known dynamics of the robot, the human, and the environment in a modular fashion.

**Specifying Human Model** While necessary domain expertise is available for the task model, robot developers only have partial knowledge of human behavior. Further, human behavior depends on latent states (such as goal, trust, attention) that are difficult to sense. Hence, in the prior art, learning approaches have been used to specify models of human behavior, which require labeled data. To reduce the effort required for labeling data, in AdaCoRL, we leverage constrained variational inference (CVI) to generate models of human behavior [12]. CVI, a hybrid learning approach, enables the use of both semi-supervised data and domain expertise in the learning process. We demonstrate through numerical experiments that, despite significantly fewer labels, models learned through CVI are equally or more accurate as compared to models learned through a supervised approach.

**Generating Robot Behavior** Given the task and human models, our approach allows for the automatic generation of the robot's decision-making model as a partially observable Markov decision process (POMDP). We generate robot policies, given the decision-making model, using a variant of the R-DESPOT algorithm, a state-of-the-art POMDP solver [13]. We demonstrate AdaCoRL in human-robot collaborative tasks with state spaces significantly larger than prior art ($>1$ million states), short planning time ($< 0.5$ s), and temporally-extended actions (shown in Fig. 1). Through experiments, we confirm that the benefit of our model specification approach, when coupled with the planning algorithm, also translates to metrics of human-robot collaboration.

## 2 Related Work

In recent years, multiple models and algorithms have been developed to enable robots to work with or alongside humans [14]. Here, we discuss approaches related to interaction planning for human-robot collaborative tasks. We focus on sequential tasks with two agents (i.e., a single human and a single robot) and a known objective. Based on the domain and the application, this task objective is typically composed of metrics of safety, task success, efficiency, and fluency of interaction [4].

**Modeling Human Behavior** Despite a shared and known task objective, the human might have multiple ways to accomplish the collaborative task. The human's behavior may also depend on their preference and mental states. Thus, approaches to interaction planning utilize a predictive model of human behavior.[1] Due to difficulty in hand-crafting a human model for all possible interaction scenarios, learning algorithms that utilize data of human behavior are used in prior art [16, 17]. For instance, Javdani et al. [3] utilizes inverse reinforcement learning to learn goal-directed human policies. Similarly, variants of inverse reinforcement learning and supervised learning have been used to model human behavior for scenarios of close-proximity interaction, shared teleoperation, and simulated autonomous driving [6, 10, 18, 19, 20, 21, 22, 23]. Along with data of human behavior, these learning approaches require labels for human's latent states (such as goal, preference, or trust).

In practice, obtaining labels for these latent states is time-intensive, as the latent states cannot be readily measured and evolve during task execution. Our approach aims to reduce this labeling effort. Nikolaidis et al. [2] provides an unsupervised approach to recover human type but assumes that the latent state (human type) does not change during task execution. In contrast, by utilizing semi-supervised data, AdaCoRL jointly learns human's policy and latent state dynamics.

**Generating Robot Behavior** Given the knowledge of task objective and a model of human behavior, decision-making algorithms are required to generate robot behavior. Different modeling and algorithmic frameworks have been proposed, building upon foundations from planning as inference [24], timed-Petri nets [9], graph search [6], model predictive control [21, 22], robust control [25], and (partially observable) Markov decision processes [2, 3, 10, 19, 26, 27], among others [14, 28]. Motivated by the presence of latent states in human task execution and their heritage in human-robot collaborative research, we focus on and utilize POMDPs as the decision-theoretic model to generate robot behavior. However, to facilitate the specification of the decision-making model, AdaCoRL requires only the specification of the task model from robot developers. The robot's decision-making model is algorithmically generated using the task model and the learned model of human behavior.

## 3 Task Specification

We utilize a factored MMDP to describe the human-robot collaborative task, described as follows:

- The state space $S$ denotes the finite set of states $s$. We use a factored representation where $s \doteq (s_H, s_R, s_E)$, in which $s_H$ and $s_R$ correspond to human- and robot-specific features, respectively, and $s_E$ denotes additional features (e.g., based on task structure and the environment);
- The action space $A = A_H \times A_R$ denotes the finite set of joint actions $a \doteq (a_H, a_R)$. $A_H$ and $A_R$ denote the sets of humans action $a_H$ and robot action $a_R$, respectively.
- The state dynamics are assumed to be Markovian, and are governed by the transition function $T(s'|s,a) : S \times A \times S \to [0,1]$. The transition function is assumed to have the following structure,

$$T(s|s,a) = T_H(s'_H|s,a) \cdot T_R(s'_R|s,a) \cdot T_E(s'_E|s,a), \tag{1}$$

  where $T_H, T_R, T_E$ are transition functions for factors of the MMDP state;
- The human-robot team receives a shared reward at each step, $R(s,a) : S \times A \to \mathbb{R}$; and
- $\gamma \in [0,1]$ denotes the discount factor.

The human-robot team's objective is to maximize the expected cumulative discounted reward.

**Scope** In this paper, we consider problems where the parameters of the MMDP, i.e., $(S, A, T, R, \gamma)$, are common knowledge, and the state $s$ is observable to both the agents. Note that while actions of both the agents impact the reward and transition function, the task model does not capture human behavior or mental states (such as intents or subgoals). Thus, given knowledge of the task, the task model can be specified by the robot developer without reasoning about human behavior.

**Example Task** We describe the specification of the MMDP model via an example human-robot handover task (see Fig. 1, bottom-right). Consider a robot assisting a person preparing sandwiches in a kitchen. The kitchen includes two cooking areas and three cabinets. To prepare one sandwich, the human needs to fetch ingredients from the cabinets and assemble them in the cooking areas.

[1] While approaches that do not explicitly model human can be used to generate robot behavior, their sample complexity for human-robot collaboration is often prohibitively large [15].

However, one of the ingredients is missing from the cabinets. The robot has access to the missing ingredient. To complete the sandwich, the robot needs to handover the missing ingredient to the human at one of the cooking areas. Thus, for making multiple sandwiches, the task requires the agents to visit multiple subgoals of interest (i.e., cooking areas, cabinets) and perform temporally-extended actions (i.e., fetch, cook, handover). Further, the robot needs to predict when and where the human will be to successfully perform the handover and accomplish the collaborative task.

**Example of MMDP Specification** While specifying the MMDP for the sequential handover task, $s_H$ corresponds to the human's position, and $s_R$ corresponds to the robot's configuration (joint angles and progress of temporally-extended actions). The task progress is encoded via $s_E$. The human actions correspond to her motion primitives. The robot actions include both motion primitives and temporally-extended actions (macro actions) for performing the handover. The transition models $T_H$ and $T_R$ correspond to the motion dynamics of the human and the robot, respectively. The transition model $T_E$ corresponds to the task recipe. The reward specifies that the task should be completed as soon as possible while maintaining a safe distance between the human and the robot.

## 4 Model for Robot Decision-Making

For successful collaboration, the robot needs to make decisions to maximize the team's objective. While the task performance depends on both the agents, the robot has autonomy only over its own actions. Thus, from the robot's perspective, the problem of interaction planning is viewed as a single-agent decision-making problem (cf. [3, 10, 21, 29]). Here, we discuss the approach to arrive at this model. Following prior research [3, 10, 19], we model the robot's decision-making using a single-agent POMDP [30]. We denote the model parameters using the subscript $c$ (for **collaboration**).

**State and Action Space** The state of the decision-making model is denoted as $s_c \doteq (s, x_H)$, which is obtained by augmenting the MMDP state $s$ with human's latent decision factors (denoted by $x_H \in X_H$). The latent decision factors cannot be sensed during interactions; further, manual labeling and extensive effort are required to collect $x_H$-data for training. We reiterate that the MMDP state includes human's observable features (such as position) but not latent states (such as subgoal). Only the robot's actions are considered for the single-agent model of decision-making, thus, $A_c \doteq A_R$.

**Transition Model** $T_c(s'_c|s_c, a_R) : S_c \times A_R \times S_c \to [0, 1]$ denotes the transition dynamics. The transitions only depend on the robot action and model the effect of robot action on the collaborative task and human's task execution (via their effect on human's latent state). The transition model is derived from the task specification as follows,

$$T_c(s'_c|s_c, a_R) = T(s', x'_H|s, x_H, a_R) = T(s'|s, x_H, a_R)T(x'_H|s, x_H, a_R) \tag{2}$$

$$\begin{aligned} T(s'|s, x_H, a_R) &= \Sigma_{a_H \in A_H} T(s', a_H|s, x_H, a_R) \\ &= \Sigma_{a_H \in A_H} T(s'|s, a_R, a_H) Pr(a_H|s, x_H, a_R) \end{aligned} \tag{3}$$

$$\begin{aligned} T(x'_H|s, x_H, a_R) &= \Sigma_{a_H \in A_H} T(x'_H, a_H|s, x_H, a_R) \\ &= \Sigma_{a_H \in A_H} Pr(x'_H|s, x_H, a_R, a_H) Pr(a_H|s, x_H, a_R) \end{aligned} \tag{4}$$

Our approach assumes that the transition model is factored (Eq. 2), and the human's latent state impacts the task only through human's actions, i.e., $T(s'|s, x_H, a_R, a_H) = T(s'|s, a_R, a_H)$. The term $T(s'|s, a_R, a_H)$ is known based on the task specification (transition model of MMDP). In addition, the probability distributions corresponding to human's decision-making policy $Pr(a_H|s, x_H, a_R)$ and latent state dynamics $Pr(x'_H|s, x_H, a_R, a_H)$ are required to define the POMDP transition model. We defer the discussion of learning the human decision-making model to Sec. 5.

**Reward** Similar to the transition model, the reward function $R_c(s_c, a_R)$ for the POMDP is obtained by combining the task specification and a model of human behavior,

$$R_c(s_c, a_R) = R_c(s, x_H, a_R) = \Sigma_{a_H \in A_H} R(s, a_H, a_R) Pr(a_H|s, x_H, a_R) \tag{5}$$

where, $R(s, a_H, a_R)$ is known based on the MMDP.

**Observation Space** The task state $s$ is modeled as observable, while $x_H$ is unobservable.

The discount factor is identical to that of the MMDP. The analysis describes the generation of robot's decision-making model from the task specification and a model of human behavior. AdaCoRL is modular in that the same human model can be utilized for different tasks, and vice-versa. This feature facilitates the personalization of robot behavior and reprogramming of robots for new tasks.

## 5 Model for Human Decision-Making

To complete the specification of the robot's decision-making model, the robot developer needs to specify human's policy $Pr(a_H|s, x_H, a_R)$ and latent state dynamics $Pr(x'_H|s, x_H, a_R, a_H)$. In our framework, with the aim of reducing the developer's specification effort, we provide a hybrid semi-supervised approach to jointly learn the human's policy and latent state dynamics.

**Domain Expertise for Model Specification** In addition to labeled data, partial specification of behavior is often available from the robot developer. Minimally, this includes the specification of features (decision factors) that impact the human's behavior (such as $s_H, x_H$). However, often partial knowledge of latent state dynamics or policies is also available. For instance, in an assembly task, not only are the human's subgoals ($x_H$) known, often, the (complete or partial) sequence of subgoals is also known. Similarly, for modeling behavior of a human driver, policy in some states (such as at a traffic signal) may be reliably known but not at other parts of the road. We posit that leveraging such domain expertise can reduce the labeling effort required from the developer.

**Agent Markov Model** To pose the learning problem, we represent the human's decision-making as an Agent Markov Model (AMM) [12]. The AMM models the sequential decision-making behavior of an agent with both observable $s_A$ and latent $x_A$ decision factors. The agent's action selection is quantified by the agent's policy $\pi_A(a_A|x_A, a_A)$. The transition dynamics of the decision factors are modeled as Markovian with the following factored structure

$$T(s'_A, x'_A|s_A, x_A, a_A) = T(s'_A|s_A, a_A) \cdot T_x(x'_A|s_A, x_A, a_A) \tag{6}$$

i.e., the latent factors model mental states and impact the observable states only via actions. For modeling the human behavior as an AMM, we choose $s_A \subseteq (s, a_R)$, $x_A \equiv x_H$, and $a_A \equiv a_H$. This enables us to recover the terms required for specifying the robot's decision-making model. Specifically, the human's policy $Pr(a_H|s, x_H, a_R) \equiv \pi_A(a_A|x_A, a_A)$, and latent state dynamics $Pr(x'_H|s, x_H, a_R, a_H) \equiv T_x(x'_A|s_A, x_A, a_A)$.

**Inputs for AMM Learning** AdaCoRL views the set of human's decision factors/states $(s_A, x_A)$ and the transition model of observable states $T(s'_A|s_A, a_A)$ as known parameters for learning. Since $s_A$ and its transition model is available from the task specification, the developer additionally only needs to specify the set of latent decision factors $x_A \in X_A$. For instance, in applications where the latent state represents the human's subgoal, this input corresponds to the specification of the number of subgoals. Similarly, if the latent state represents trust, this input corresponds to the number of discrete levels of trust. The training data includes unsupervised data of human behavior (i.e., $N$ sequences of $s_A, a_A$-tuples), and labels of the latent state for a subset of the unsupervised data (i.e., $M$ labels of $x_A$). The robot developer, optionally, can provide partial specification (i.e., some elements) of the human's policy and latent state dynamics.

**Bayesian AMM Learning** Following Unhelkar and Shah [12], we pose the problem of recovering AMM parameters as one of Bayesian learning. We first specify the priors for unknown model parameters $(\pi_A, T_x, b_x)$ and infer their posterior $Pr(\pi_A, T_x, b_x|\cdot)$ given the inputs for learning. In our approach, as the set of latent state is a known parameter, we utilize the following parametric priors for the latent state distributions, $b_x(\cdot) \sim \text{Dirichlet}(\alpha_b)$ and $T_x(\cdot|x_H, s_H, a_H) \sim \text{Dirichlet}(\alpha_t)$, where $\alpha_b$ and $\alpha_t$ are hyper-parameters. In the absence of additional domain knowledge, we use Dirichlet distribution as policy priors with hyper-parameter $\rho$. To perform posterior inference, we utilize a combination of sampling-based and variational inference algorithm.

**Blocked Gibbs Sampling** We first perform blocked Gibbs sampling, described in Appendix A. Intuitively, the Gibbs sampler begins with an initial guess for the unknown latent state $(x_A)$ sequences. Next, using this initial guess and semi-supervised data of behavior, the sampler updates the unknown AMM parameters. This procedure is repeated iteratively to generate successive samples. The Gibbs sampler provides a Bayesian approach to learn the AMM given semi-supervised data; however, it cannot incorporate the partial specifications available from the robot developer.

**Constrained Variational Inference** To perform hybrid learning and incorporate partial knowledge of $T_x$ or $\pi$, we utilize the constrained variational inference algorithm [12]. Similar to variational inference [31], CVI approximates the posterior using a known distribution, parametrized by $\lambda$. The variational parameter $\lambda$ (and, consequently, the approximate posterior) is learned through constrained optimization. The optimization objective is the same as that of variational inference (evidence lower bound, [31]), and constraints are obtained from partial specifications. For initializing CVI while learning the human model, we use the result of the Gibbs sampler.

We consider the following types of partial specifications about latent state dynamics,

$PS_1$) transitions from $x_A = i$ to $x_A \in X_n$ are not possible, where the developer specifies $i$, $X_n$;

$PS_2$) transitions from $x_A = i$ to $x_A \in \{j, k\}$ are equally likely, where the developer specifies $j, k$;

$PS_3$) minimum time $t_i$ spent in the state $x_A = i$, where the developer specifies $i$ and $t_i$; and

$PS_4$) the subset of features from the set $\{s_A, a_A, x_A\}$ that impact the latent state transition.

These specifications enable the developer to specify domain expertise regarding the human's behavior dynamics. For instance, consider the case where the robot developer models $x_A$ as a human's subgoal and has knowledge of the possible sequences of the human's subgoals (i.e., the subgoal sequence). If subgoal $j$ is not possible after the subgoal $i$, this information can be specified via the first type of partial specification $PS_1$. Similarly, if the information is available about the duration of activity at the subgoal $i$, it can be specified via $PS_3$. $PS_{1-3}$ incorporates domain expertise specific to a single state. $PS_4$ enables the robot developer to perform feature selection while modeling the dynamics of human behavior. To utilize CVI, the partial specifications need to be converted to constraints over the variational parameters. We utilize the moments of Dirichlet distribution to obtain these constraints [12]. In summary, by utilizing CVI, ADACORL provides a hybrid semi-supervised approach to specifying decision-making models for human-robot collaboration.

## 6 Algorithm for Robot Decision-Making

Having discussed the model specification process, we briefly describe our approach to interaction planning. Given a robot's decision-making model, existing POMDP solvers can be used to arrive at the robot's interaction policy, $\pi_R$, which maps the robot's belief about the state $s_c$ to robot's actions $a_R$. Planning time available during interaction is limited; thus, we leverage the R-DESPOT algorithm for interaction planning [13].

**R-DESPOT** The Regularized-DESPOT (or R-DESPOT) is an anytime execution-time algorithm for solving POMDPs. By reasoning at execution-time and in anytime fashion, it can generate policies for problems with large state spaces, for which offline solvers may have memory bottlenecks. To identify the best action given the robot's belief, R-DESPOT performs forward simulations using the POMDP model and creates a sparse approximation of the belief tree through heuristic search. To perform this search, the R-DESPOT requires a default policy and bounds on the value of a belief.

**Default Policy and Value Bounds** To pre-compute these inputs, we first arrive at an MDP corresponding to the robot's POMDP by assuming that the states are fully observable. We compute the optimal policy of this MDP (which maps states to actions) using value and policy iteration. Following Ye et al. [13], we use the mode-MDP policy to obtain the default POMDP policy (which maps beliefs to actions). The value of the MDP policy is used as the upper bound value for the state. The upper bound value for a belief is computed as the expectation of the MDP value under the belief, while the lower bound value is computed at planning time using forward simulations.

**Interleaving Planning and Execution** During interactions, the outcome of a robot's action and the corresponding observation are received a timestep after the robot selects its action. To interleave planning and execution, we modify the belief tree construction and action selection of R-DESPOT. While creating the sparse belief tree, R-DESPOT uses all actions to expand the root belief node. In contrast, we only include the previously selected action to expand the root node, as the action at the root node is known. At the end of the planning time, when a new observation is received, we perform action selection using the constructed tree. We first identify the child of the root node that has support for the new observation, and use the best action of this child node as the robot's action.

## 7 Experiments

We evaluate the performance of ADACORL using two collaborative tasks: the handover task and a shared workspace task. We conduct experiments both in simulation and with human participants. Video demonstrations, implemented on a UR-10 robot [32], are included as supplementary material. Further, additional results are included in Appendices B-C. We hypothesize that our hybrid semi-supervised approach, despite fewer number of inputs, can generate robot behavior that accrues equal or higher reward as compared to a supervised approach during human-robot collaboration.

| | Shared Workspace Task | | | | Handover Task | | | |
|---|---|---|---|---|---|---|---|---|
| Model | $w\text{KL}(T_x)$ | $w\text{KL}(\pi_H)$ | Reward | Stops | $w\text{KL}(T_x)$ | $w\text{KL}(\pi_H)$ | Reward | Handovers |
| Hand-crafted | 0 | 0 | -213.8 ± 22.2 | 0.03 | 0 | 0 | -163.8 ± 3.0 | 2.0 |
| Supervised | 0.011 | 0.041 | -223.6 ± 27.4 | 0.94 | 0.066 | 0.051 | -174.1 ± 5.4 | 1.4 |
| Semi-A | 0.014 | 0.042 | **-184.5 ± 23.1** | 0.09 | 0.035 | 0.055 | -174.4 ± 2.6 | 1.5 |
| Semi-B | 0.015 | 0.042 | -208.9 ± 25.7 | **0.03** | 0.030 | 0.052 | **-168.4 ± 3.0** | **1.7** |

Table 1: Model specification and interaction planning performance. The model specification performance ($w$KL) is reported for the best model learned after ten learning trials. The metrics of planning performance (reward, safety stop time, and number of handovers) are averaged over thirty-two trials.

**Collaboration Scenarios** Both the tasks occur in a shared kitchen environment, depicted in Fig. 1, where the human is making sandwiches. As described in Sec. 3, in the handover task, the robot needs to handover missing ingredients to the human. In our experiments, the robot needs to perform three handovers during an episode. In the shared workspace task, the robot is tasked with pouring drinks in four cups on the table, while the human is making sandwiches. The robot needs to finish pouring drinks as soon as possible while maintaining a safe distance with the human.

**Task Specification** While specifying both tasks as MMDPs, $s_H$ represents the position of human's hand, $s_R$ represents the robot's joint angles and the progress of macro action, and $s_E$ encodes task progress. A two-dimensional grid is used to represent the human's position; the size of the grid is 12 for the shared workspace task and 16 for the handover task. The robot's action space consists of motion primitives in the configuration space and macro actions (for pouring and handover); the size of $s_R$ state space is $\approx 4,000$. The task progress, $s_E$, is modeled with $\approx$ 30 states. In total, the shared workspace and handover tasks have $\approx 1.8 \times 10^6$ and $\approx 2 \times 10^6$ states, respectively. The reward function penalizes unsafe executions (i.e., collisions) and emphasizes faster task completion. To prevent collisions during execution, a safety stop is implemented, due to which the robot stops if it is within $0.1$ m of the human. The timestep of the MMDP (and the planning time) is $0.3$ s.

**Ground Truth Human Behavior** For simulation experiments, we define a ground truth model of human behavior with the observable decision factor $s_A$ as the human's position $s_H$, and the latent decision factor $x_A$ as the human's subgoal. Both the tasks include five subgoals, corresponding to the cabinets and cooking areas. The human exhibits subgoal-directed motion. Upon reaching a subgoal, the human finishes the activity (e.g., cook, fetch) at the subgoal within a pre-specified time interval. Thus, the simulated human exhibits non-Markovian behavior. The task structure is used to define the subgoal transitions, e.g., after collecting ingredients from the cabinet, the human goes to the cooking area for making the sandwich. The human can finish the task with different types of subgoal sequences. Datasets for the simulation experiments are generated using the true model.

**Baselines** We use a hand-crafted AMM human model, and a model learned using supervised learning as the baselines. The ground truth human behavior is used to specify the hand-crafted AMM model; however, the AMM is Markovian. Supervised learning is performed using the Gibbs sampler and variational inference described in Sec. 5, however, without sampling the latent states (as $x_H$ labels are available). The supervised approach serves as a proxy for prior approaches, which require labeled data of human's latent states and cannot utilize partial specifications.

A training dataset of sixteen $(s_H, a_H)$-sequences and identical hyper-parameters are used for all learning algorithms. Duration of each sequence is $\approx$ 200 timesteps. For supervised learning, in addition, labels of $x_H$ are provided for all sequences (i.e., $\approx$ 3200 labels). We evaluate two variants of our approach: Semi-A and Semi-B. For all the variants, labels of $x_H$ are provided for only four sequences (i.e., $\approx$ 800 labels). For Semi-A, additionally, $\approx$ 10 high-level inputs are provided that encode domain expertise about subgoal sequences (specified as $PS_{1-2}$) and minimum time to complete the activity at each subgoal (specified as $PS_3$). For Semi-B, we further specify that the subgoal dynamics $T_x$ only depends on the previous subgoal (i.e., feature selection specified as $PS_4$).

For each pair of task and learning algorithm, ten AMM models are learned. The AMM model with the lowest test error and the task specification are then used to arrive at the robot's POMDP. Thirty-two simulations of human-robot collaboration are conducted to evaluate the performance of the learned model for interaction planning. In practice, dataset of human behavior may not include all possible types of behavior. To test this case, for the handover task, half of the test set consists of behaviors (specifically, subgoal sequences) that are absent in the training set.

**Simulation Results** The results of the simulation evaluations are summarized in Table 1. Modeling performance is evaluated using the weighted KL divergence ($w$KL) between the learned and the true models [33]. In both the tasks, the models learned through our semi-supervised approach (Semi-A and Semi-B) have model alignment comparable to that of the supervised approach. As the hand-crafted model is specified using the ground truth model, it has zero KL divergence. Near identical POMDP models can have vastly different policies. Thus, while the modeling performance of our approach is encouraging, evaluating its planning performance is critical. We evaluate interaction planning performance using the total reward and task-specific metrics – namely, the timesteps for which robot's safety stop was engaged (denoted as Stops) and the number of successful handovers.

By reasoning about the human's latent state, all approaches are able to complete the shared workspace task with minimal safety stops; the highest average safety stop time is $1$ timestep for the supervised model. However, the semi-supervised model (Semi-A), on average, obtains higher reward than both the hand-crafted model and the supervised model. As the ground truth behavior is not Markovian, the robot behavior with the hand-crafted model is not necessarily optimal.

For the multi-step handover task, the hand-crafted model performs best. In their failure modes, the models either incorrectly identify the human's subgoal or identify too late to complete the handover. Among the learned models, we again observe that the semi-supervised approach (Semi-B), on average, accrues higher reward and completes more handovers as compared to the model learned via supervised learning. The supervised approach relies on data alone and cannot utilize high-level inputs, which can result in models that overfit to the training data. The generalization to behavior absent in the training set is critical for human-robot collaboration, as the training data will seldom have all possible types of human behavior. In contrast, our approach can successfully utilize both partially labeled data and domain expertise, thereby improving performance in collaborative tasks.

**Performance with Human Participants** We next evaluated our approach via a human subject study ($N = 9$; 5 female, 4 male; median age $= 29$ years). Two participants indicated prior experience with robots. The human-robot team performed the multi-step handover task, where $s_H$ was sensed using the PhaseSpace motion capture system. Each participant performed the task six times, twice each with the three models: hand-crafted, supervised, and semi-supervised (Semi-B). The order of treatments was randomized. To train the learned models, data was collected from three humans (different from study participants) and labeled by the authors. Similar to the simulations, the training data included a subset of possible human behaviors (subgoal sequences). Further, latent state (subgoal) labels of only $25\%$ of training sequences were provided to the semi-supervised approach.

We compare the task-specific metric, number of handovers, for the three models. A non-parametric Friedman test of differences was conducted, which rendered a Chi-square value of $11.6$ indicating statistical significance ($p < 0.01$). On average, the hand-crafted model resulted in $2.2$ handovers, the supervised model in $1.4$ handovers, and the Semi-B model in $2.9$ handovers. Similar to simulation experiments, the supervised model completes the fewest handovers. With human participants, the hand-crafted model (obtained from the true model of the simulated human) results in fewer handovers than the semi-supervised model. These results further demonstrate that in practice, it is difficult to rely only on either manual specification or data alone to generate models of decision-making. In summary, our evaluations confirm that our hybrid semi-supervised approach, despite $75\%$ fewer labels, performs equally well or better than a model learned with fully-labeled behavioral data.

## 8 Conclusion

We present a novel framework, ADACORL, to specify decision-making models and generate robot behavior for human-robot collaboration. In the prior art, labeled datasets of human's latent states have been a prerequisite to generate fluent robot behavior. Our approach to model specification relaxes this requirement by learning decision-making models with partially labeled data and domain expertise. We demonstrate our approach in two human-robot collaborative tasks, with state spaces significantly larger than those considered in the prior art. Our approach to interaction planning enables the robot to make decisions in these tasks, despite their large state spaces and short planning times. In future work, we aim to further expand the vocabulary of our semi-supervised approach by considering additional types of partial specifications and evaluating their relative utility for developers. Another avenue is to leverage approaches that infer task objectives [34, 35, 36, 37] and further reduce the developer's effort for specifying the task model and creating collaborative agents.

**Acknowledgments**

We thank the anonymous reviewers for their constructive comments and suggestions.

## References

[1] G. Hoffman and C. Breazeal. Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team. In *Intl. Conf. on Human-Robot Interaction (HRI)*, pages 1–8. ACM, 2007.

[2] S. Nikolaidis, R. Ramakrishnan, K. Gu, and J. Shah. Efficient model learning from joint-action demonstrations for human-robot collaborative tasks. In *Intl. Conf. on Human-Robot Interaction (HRI)*, pages 189–196. ACM, 2015.

[3] S. Javdani, H. Admoni, S. Pellegrinelli, S. S. Srinivasa, and J. A. Bagnell. Shared autonomy via hindsight optimization for teleoperation and teaming. *The International Journal of Robotics Research*, 37(7):717–742, 2018.

[4] G. Hoffman. Evaluating fluency in human–robot collaboration. *IEEE Transactions on Human-Machine Systems*, 49(3):209–218, 2019.

[5] T. Iqbal and L. D. Riek. Human-robot teaming: Approaches from joint action and dynamical systems. *Humanoid Robotics: A Reference*, pages 2293–2312, 2019.

[6] V. V. Unhelkar, P. A. Lasota, Q. Tyroller, R.-D. Buhai, L. Marceau, B. Deml, and J. A. Shah. Human-aware robotic assistant for collaborative assembly: Integrating human motion prediction with planning in time. *IEEE Robotics and Automation Letters*, 3(3):2394–2401, 2018.

[7] M. G. Jacob, Y.-T. Li, G. A. Akingba, and J. P. Wachs. Collaboration with a robotic scrub nurse. *Commun. ACM*, 56(5):68–75, 2013.

[8] M. A. Diftler, J. Mehling, M. E. Abdallah, N. A. Radford, L. B. Bridgwater, A. M. Sanders, R. S. Askew, D. M. Linn, J. D. Yamokoski, F. Permenter, et al. Robonaut 2-the first humanoid robot in space. In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 2178–2183. IEEE, 2011.

[9] C. Chao and A. Thomaz. Timed Petri nets for fluent turn-taking over multimodal interaction resources in human-robot collaboration. *The International Journal of Robotics Research*, 35(11):1330–1353, 2016.

[10] S. Nikolaidis, D. Hsu, and S. Srinivasa. Human-robot mutual adaptation in collaborative tasks: Models and experiments. *The International Journal of Robotics Research*, 36(5-7):618–634, 2017.

[11] F. A. Oliehoek, C. Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

[12] V. V. Unhelkar and J. A. Shah. Learning models of sequential decision-making with partial specification of agent behavior. In *AAAI Conf. on Artificial Intelligence*, 2019.

[13] N. Ye, A. Somani, D. Hsu, and W. S. Lee. DESPOT: Online POMDP planning with regularization. *Journal of Artificial Intelligence Research*, 58:231–266, 2017.

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

[15] R. Choudhury, G. Swamy, D. Hadfield-Menell, and A. D. Dragan. On the utility of model learning in HRI. In *Intl. Conf. on Human-Robot Interaction (HRI)*, pages 317–325. IEEE, 2019.

[16] P. A. Lasota and J. A. Shah. A multiple-predictor approach to human motion prediction. In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 2300–2307. IEEE, 2017.

[17] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI Conf. on Artificial Intelligence*, volume 8, pages 1433–1438, 2008.

[18] H. S. Koppula and A. Saxena. Anticipating human activities for reactive robotic response. In *Intl. Conf. on Intelligent Robots and Systems (IROS)*, page 2071. Tokyo, 2013.

[19] M. Chen, S. Nikolaidis, H. Soh, D. Hsu, and S. Srinivasa. Planning with trust for human-robot collaboration. In *Intl. Conf. on Human-Robot Interaction (HRI)*, pages 307–315. ACM, 2018.

[20] A. D. Dragan and S. S. Srinivasa. A policy-blending formalism for shared control. *The International Journal of Robotics Research*, 32(7):790–805, 2013.

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

[22] E. Schmerling, K. Leung, W. Vollprecht, and M. Pavone. Multimodal probabilistic model-based planning for human-robot interaction. In *Intl. Conf. on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018.

[23] S. Nikolaidis, M. Kwon, J. Forlizzi, and S. Srinivasa. Planning with verbal communication for human-robot collaboration. *ACM Transactions on Human-Robot Interaction (THRI)*, 7(3):22, 2018.

[24] P. Trautman, J. Ma, R. M. Murray, and A. Krause. Robot navigation in dense human crowds: Statistical models and experimental studies of human–robot cooperation. *The International Journal of Robotics Research*, 34(3):335–356, 2015.

[25] J. F. Fisac, A. Bajcsy, S. L. Herbert, D. Fridovich-Keil, S. Wang, C. J. Tomlin, and A. D. Dragan. Probabilistically safe robot planning with confidence-based human predictions. In *Robotics: Science and Systems (R:SS)*, 2018.

[26] F. Broz, I. Nourbakhsh, and R. Simmons. Planning for human–robot interaction in socially situated tasks. *International Journal of Social Robotics*, 5(2):193–214, 2013.

[27] D. Whitney, E. Rosen, J. MacGlashan, L. L. Wong, and S. Tellex. Reducing errors in object-fetching interactions through social feedback. In *Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2017.

[28] T. Kruse, A. K. Pandey, R. Alami, and A. Kirsch. Human-aware robot navigation: A survey. *Robotics and Autonomous Systems*, 61(12):1726–1743, 2013.

[29] F. S. Melo, M. T. Spaan, and S. J. Witwicki. QueryPOMDP: POMDP-based communication in multiagent systems. In *European Workshop on Multi-Agent Systems*, pages 189–204. Springer, 2011.

[30] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Partially observable markov decision processes for artificial intelligence. In *Annual Conference on Artificial Intelligence (AAAI)*, pages 1–17. Springer, 1995.

[31] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley. Stochastic variational inference. *The Journal of Machine Learning Research*, 14(1):1303–1347, 2013.

[32] Universal Robots. *UR 10 Collaborative Industrial Robotic Arm*, Accessed July 1, 2019. URL `https://www.universal-robots.com/products/ur10-robot/`.

[33] A. Panella and P. Gmytrasiewicz. Interactive POMDPs with finite-state models of other agents. *Journal of Autonomous Agents and Multi-Agent Systems*, 31(4):861–904, 2017.

[34] M. Cakmak and A. L. Thomaz. Designing robot learners that ask good questions. In *Intl. Conf. on Human-Robot Interaction (HRI)*, pages 17–24. ACM, 2012.

[35] S. Chernova and A. L. Thomaz. Robot learning from human teachers. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(3):1–121, 2014.

[36] D. Hadfield-Menell, S. Milli, P. Abbeel, S. J. Russell, and A. Dragan. Inverse reward design. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6765–6774, 2017.

[37] A. Shah, P. Kamath, S. Li, and J. A. Shah. Bayesian inference of temporal task specifications from demonstrations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3804–3813, 2018.

# A Blocked Gibbs Sampler for the Agent Markov Model

**Algorithm 1:** Gibbs Sampler to Learn AMM

```
Data: $N$ seqs. of $(s_A, a_A)$-tuples, $M$ labels of $x_A$
Result: Samples for $\pi_A$, $T_x$, $b_x$, and $N$ seqs. of $x_A$
Initialize $\pi_A$, $T_x$, $b_x$, and unknown $x_A$ randomly
while number of samples generated $< N_s$ do
    Sample initial distribution, $Pr(b_x|\mathbf{x_A}, \text{data}; \alpha_b)$
    Sample transition model, $Pr(T_x|\mathbf{x_A}, \text{data}; \alpha_v)$
    Sample policy, $Pr(\pi_H|\mathbf{x_A}, \text{data}; \rho)$
    Sample latent states, $Pr(\mathbf{x_A}|\text{data}, T_x, b_x, \pi_H)$
end
```

Algorithm 1 describes the blocked Gibbs sampler for the Agent Markov Model. Using data and (optionally) labels of human behavior, the blocked Gibbs sampler enables learning of the latent parameters $(\pi_A, T_x, b_x)$ of the AMM describing human behavior.

The Gibbs sampler begins with an initial guess for the unknown latent state sequences $\mathbf{x_A}$. Samples of model parameters $(\pi_A, T_x, b_x)$ are then generated by utilizing the counts of latent state sequences $\mathbf{x_A}$. For sampling $T_x$, we define $n_{isaj}$ as the count of transition from latent state $x_A = i$ to latent state $x'_A = j$ for action $a_A$ and observed state $s_A$. Both $i$ and $j$ range from 1 to $|X_A|$. The conditional distribution of $T_x$ is given as: $T_x(\cdot|x_A, s_A, a_A) \sim \text{Dirichlet}(n_{xsa\cdot} + \alpha_t)$. The sampling of initial distribution follows similarly, and depends on the initial counts of latent states in the data. Conditional distributions for $\pi_H$ depend upon the policy prior. Latent state sequences are sampled using a variant of forward filtering-backward sampling (FFBS) algorithm derived for the AMM. This procedure is repeated iteratively to generate successive samples.

# B Utility of Incorporating Domain Knowledge

The ability to incorporate domain knowledge, via partial specifications of latent AMM parameters $(\pi, T_x)$, is one of the key contributions of our model specification approach. In many human-robot collaboration tasks, such domain knowledge is available. However, it is difficult to incorporate this domain knowledge during model learning. To highlight the benefit of incorporating this domain knowledge, we include additional simulation evaluations.

Specifically, we conduct a set of simulations for a variant of our approach (Semi-C). Similar to Semi-A and Semi-B, labels of $x_H$ were provided for four sequences (i.e., $\approx 800$ labels). However, no high-level inputs were provided. Thus, Semi-C still performs semi-supervised learning but does not incorporate the developer's domain expertise. The remaining parameters were identical to that of the simulation experiments presented in Sec. 7.

| | Shared Workspace Task | | | | Handover Task | | | |
|---|---|---|---|---|---|---|---|---|
| Model | $w$KL($T_x$) | $w$KL($\pi_H$) | Reward | Stops | $w$KL($T_x$) | $w$KL($\pi_H$) | Reward | Handovers |
| Semi-C | 0.012 | 0.041 | -252.4 ± 32.4 | 3.5 | 0.069 | 0.056 | -173.9 ± 5.4 | 1.4 |

Table 2: Model specification and interaction planning performance for the variant Semi-C.

The performance of Semi-C is summarized in Table 2. We observe that for both the tasks, Semi-C accrues lower reward than both Semi-A and Semi-B. Thus, this set of evaluations further highlights the importance of the ability to utilize partial domain knowledge.

The semi-supervised approach Semi-C has access to significantly fewer labels than the supervised approach. Further, both approaches do not utilize partial domain knowledge. Thus, we expect the performance of the supervised approach to be equal or better than that of Semi-C. This is indeed the case for the shared workspace task. However, in the handover task, Semi-C accrues reward similar to that of the supervised approach. We posit that this trend occurs since the supervised approach may overfit to the training data, which is detrimental in tasks where the test set consists of behaviors that are absent in the training set (e.g., the handover task in our evaluations).

Thus, in summary, these additional evaluations confirm that, despite fewer labels, semi-supervised learning with partial domain knowledge (i.e., AdaCoRL) results in performance better than that of both supervised learning and vanilla semi-supervised learning. Further, the importance of domain knowledge is heightened for collaboration scenarios where the training data set of human behavior does not include human behavior encountered during the interaction.

## C Performance with Human Participants

In Sec. 7, we discuss the evaluations of our approach via a human subject study. To further describe the results from these experiments, we include figures summarizing the performance of the different approaches. Since we do not have access to the true model of human participants, the metrics of model alignment cannot be computed for experiments with human subjects. Thus, we provide figures for the metrics of collaborative performance – namely, cumulative reward and the number of handovers.

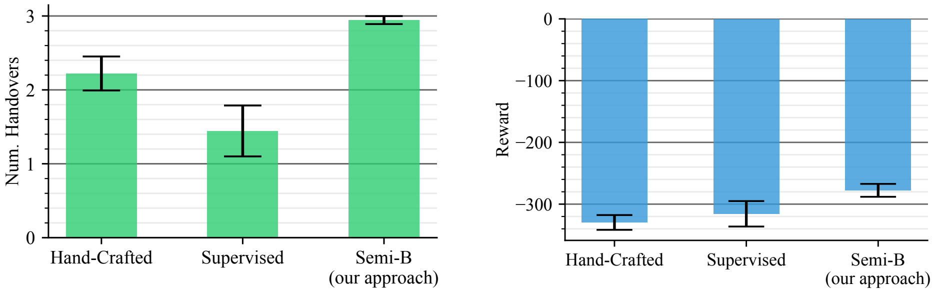

Figure 2: Metrics of collaborative performance from the human subject study.

We reiterate that successful handovers were essential for the handover task. Only by successfully completing the handover, the robot could assist human in preparation of the sandwich. As shown in Fig. 2, the robot using our approach performs more handovers than both the baselines. Further, our approach accrues a higher reward than both the hand-crafted and supervised models.