Interactive Robot Training for Non-Markov Tasks

Ankit Shah
CSAIL
Massachusetts Institute of Technology
Cambridge, Massachusetts, 02139
Email: ajshah@mit.edu

Julie Shah
CSAIL
Massachusetts Institute of Technology
Cambridge, Massachusetts, 02139
Email: julie_a_shah@csail.mit.edu

Abstract—Defining sound and complete specifications for robots using formal languages is challenging, while learning formal specifications directly from demonstrations can lead to over-constrained task policies. In this paper, we propose a Bayesian interactive robot training framework that allows the robot to learn from both demonstrations provided by a teacher, and that teacher’s assessments of the robot’s task executions. We also present an active learning approach – inspired by uncertainty sampling – to identify the task execution with the most uncertain degree of acceptability. We demonstrate that active learning within our framework identifies a teacher’s intended task specification to a greater degree of similarity when compared with an approach that learns purely from demonstrations. Finally, we also conduct a user-study that demonstrates the efficacy of our active learning framework in learning a table-setting task from a human teacher.

I. INTRODUCTION

Humans are adept at quickly learning to perform multi-step tasks like setting a dinner table, clearing a desk, or assembling furniture. Tasks such as these typically involve temporal elements like adherence to constraints, decomposition into and prioritization of sub-tasks. Linear temporal logic (LTL) provides an expressive grammar for modeling a range of such non-Markov temporal properties; however, formal languages like LTL are often unwieldy for an average user. In order to facilitate rapid deployment of capable robots to novel scenarios and tasks, it is desirable to allow users with task-specific expertise to directly program the robots.

There has been a considerable amount of research related to inferring formal specifications through intuitive interfaces such as demonstrations and preferences expressed as natural language instructions. To resolve the ambiguity associated with these teaching modalities, Shah et al. proposed planning with uncertain specifications (PUnS), a framework for generating task plans wherein specifications are expressed as a belief $P(\phi)$ over LTL formulas. However, policies computed to optimize the PUnS criteria generate task executions that attempt to satisfy a large number of candidate formulas, potentially over-constraining task execution. In this paper, we demonstrate that belief over LTL formulas can also serve to identify task executions with an uncertain degree of acceptability. These executions can then be demonstrated back to the user in order to elicit an assessment of their acceptability, which in turn can reduce the uncertainty of the distribution.

In this paper, we propose computational models for identifying and performing such ambiguous task executions. We also evaluate the performance of this active learning approach compared with learning purely from demonstrations (termed Batch), and another interactive approach wherein task executions are generated by performance of random actions (termed Random). We demonstrate that our proposed method yields posterior belief distributions with higher similarity to the ground truth specification Batch and Random approaches for a wide range of ground truth task specifications. We also conducted a human-participant study involving training a robot to set a dinner table using the three training protocols. Our results demonstrated the efficacy of our active learning approach in learning the task specifications that are well aligned with the ground truth specifications (average similarity: 0.86 95% CI [0.82, 0.92]); however, they also indicated that relative performance of the robot training protocols was dependent on the ground truth specifications of the task that the robot must learn.

II. RELATED WORK

The objective of allowing domain experts to directly program robots has driven research into methods for programming robots through intuitive modalities. Prior research has developed models to learn the teacher’s intended task by processing the teacher’s input provided through demonstrations or preferences or corrections or preferences. A key feature of our approach is the ability to model temporal tasks by using LTL as the specification language. Chanlette-Vazquez et al. proposed a maximum entropy estimator of observing the demonstrated task execution given the true specification. Kasenburg and Scheutz proposed optimization-based framework for modeling a decision-maker’s behavior as an LTL formula. Cama-cho et al. developed an exact method for mining LTL formulas based on sets of satisfying and non-satisfying traces for the shortest LTL-F(inite) formula. Shah et al. proposed a Bayesian approach to specification inference to model the uncertainty associated with inferring task specifications from a few demonstrations. While most of the previous work on learning non-Markov task specifications has focused on learning solely from teacher’s demonstrations, in this paper, we adopt an iterative Bayesian approach that unifies the teacher’s
input provided via demonstrations or as assessments of robot’s task executions.

There has also been considerable interest in developing algorithms to allow the learner to elicit the teacher’s feedback (An active learning paradigm). An expected benefit of active approaches is that the learner can guide the teacher’s feedback to be most impactful in modifying its own behavior. Çakmak et al. [18, 19] developed a taxonomy of queries that allow a learner to refine its understanding of the task specifications. Sadigh et al. [22] proposed an active learning framework for sequential decision-making problems that relies on pairwise preference between candidate trajectories, that were selected according to a maximum volume removal heuristic. Biyik et al. [14] extended this to generate queries using maximum information gain criterion. Biyik and Sadigh [13] proposed a batch active framework for preference based learning where multiple queries are generated simultaneously instead of a single query at a time. However, present research into active learning for robotics has largely focused on formulations that represent the underlying task as a Markov decision process (MDP) with the state space known a priori.

Admitting non-Markov task specifications represents an improvement in task complexity that the robot can handle. Therefore, prior research has lead to development of planning algorithms for hybrid controller synthesis [20], symbolic planning [21, 22], reinforcement learning [23, 24, 25, 26]. In this paper, we extend Planning with uncertain specification (PUnS) [6], a problem formulation that allows task specifications to be expressed as a belief over multiple LTL formulas. Policies computed to optimize the PUnS evaluation criteria satisfy the entire belief distribution rather than a single LTL formula. This allows the learner to reconcile the ambiguity inherent in the teacher’s demonstrations. Our proposed extension leverages the reward machine [26] representing the learner’s belief over LTL formula to identify a task execution suitable for active learning.

Our contribution in this paper is two-fold. First, we propose a novel interactive learning framework (Figure 1) for non-Markov tasks that unifies teacher inputs through demonstrations and assessments of the learner’s task execution. Second, we develop develop an active learning approach wherein we leverage the reward machine representation of an instance of a PUnS problem to identify task executions with the most uncertain degree of acceptability.

III. PRELIMINARIES

A. Linear Temporal Logic

Linear temporal logic (LTL), first proposed by Pnueli [11], provides a flexible grammar for defining temporal properties over Boolean propositions. A valid LTL formula is constructed using atomic propositions (discrete time sequences of Boolean values), and logical and temporal operators. The truth value of an LTL formula is evaluated for traces [α] for a set of atomic propositions α. The notation [α], t |= ϕ indicates that formula ϕ holds at time t. Trace [α] satisfies ϕ (denoted as [α] |= ϕ) iff [α], 0 |= ϕ. The minimal syntax of LTL is as follows:

\[
\varphi ::= p \mid \neg \varphi_1 \mid \varphi_1 \lor \varphi_2 \mid X \varphi_1 \mid \varphi_1 U \varphi_2
\]  

Here, p is an atomic proposition, and ϕ1 and ϕ2 represent valid LTL formulas. The operator X is read as “next”, and Xϕ1 evaluates as true at t if ϕ1 holds at t + 1. The operator U is read as “until” and ϕ1 U ϕ2 evaluates as true at t if ϕ2 holds at some time t ≥ t1, and ϕ1 holds for all t, where t1 ≤ t ≤ t2. In addition to the minimal syntax, we also incorporate the conjunction operator ∧, along with two other temporal operators; F (eventually), and G (globally). Fϕ1 holds at t1 if ϕ1 holds for some time t ≥ t1; similarly, Gϕ1 holds at t1 if ϕ1 holds for all t ≥ t1.

Finally, a progression Prog(ϕ, α) over an LTL formula with respect to the truth assignment, α, is defined such that ∀[α] : [α, [α], t] |= ϕ iff [α], t + 1 |= Prog(ϕ, α). A progression of an LTL formula with respect to a particular truth assignment is a formula that must hold at the next time step in order for the original formula to hold at the current time step. We use the syntactic progression rules defined by Bacchus and Kabanza [27] to compute formula progressions.

B. Belief over Specifications

In this paper, we adopt a Bayesian approach to inferring task specifications from demonstrations and user assessments of query executions. The robot maintains a belief over candidate LTL formulas; this belief represents the probability of a particular formula being the true specification. This distribution is defined as a mass function \( P : \varphi \rightarrow [0, 1] \). The support of \( P(\varphi) \) is restricted to a discrete set of LTL formulas \{ϕ\}, where each formula represents a property belonging to the “Obligations” class as defined by Manna and Pnueli [28].

C. Q-Learning

A Markov decision process (MDP) is a planning problem defined as a tuple \( M = (S, A, T, R) \), where \( S \) represents the set of all possible states; \( A \) is the set of actions available to the learner; \( T : P(s' \mid s, a) \) is a probability distribution over the next state \( s' \in S \) given current state \( s \in S \), and the action \( a \in A \) executed at the current time step; and \( R : S \rightarrow R \) is the reward function that returns a scalar value given the current state.

Tabular Q-learning [29] is an off-policy algorithm for computing and MDP’s optimal policy, given a discount factor γ. The action advantage function \( Q(s, a) \) is the expected discounted cumulative reward if action \( a \) were to be selected with initial state \( s \), and the optimal policy is followed from the subsequent time steps. In tabular Q-learning, the Q-value is updated via an arbitrary exploration policy; it is thus considered an “off-policy” algorithm. Given an initial estimate of the Q-value \( Q(s, a) \), if the agent performs action \( a \) from state \( s \), resulting in state \( s' \), the discount factor is \( \gamma \in [0, 1] \), and the learning rate is \( \lambda \), then the Q-value is updated as follows:

\[
Q(s, a) \leftarrow (1 - \lambda)Q(s, a) + \lambda(r + \gamma \max_{a' \in A} Q(s', a'))
\]
IV. INTERACTIVE TRAINING FOR NON-MARKOV TASKS

A. Problem Formulation

In this setting, the teacher intends to teach a task represented by an LTL formula $\varphi^*$ (unknown to the learner). In keeping with a Bayesian approach, the learner always maintains a belief over LTL formulas $P(\varphi)$ with support $\{\varphi\}$, represented as a probability distribution over candidate LTL formulas likely to be the teacher’s intended formula. The learner’s degree of success is determined using the intersection-over-union metric for LTL formulas proposed by Shah et al. [2].

The learner represents the task environment as a state, $x \in X$, and also has access to a set of actions $A$. The state of the system, $x$, maps to a set of finite known Boolean propositions, $\alpha \in \{0, 1\}^{|prop|}$, through a labeling function, $f : X \rightarrow \{0, 1\}^{|prop|}$. We assume that a trace of propositions, depicted by $[\alpha]$, is sufficient to determine the truth value of any formula within the support $\{\varphi\}$ of the learner’s belief; thus, any task execution, whether generated by the robot or demonstrated by the teacher, is represented as a trace, $[\alpha]$. We also define a Boolean label, $L([\alpha]) \in \{0, 1\}$ that indicates whether the given trace is acceptable. For the purposes of this paper, we assume that all task executions demonstrated by the teacher are labeled as acceptable, and the teacher’s assessment of the executions demonstrated by the learner is perfect.

B. Overview of the Interactive Learning Framework

Figure 1 depicts our proposed interactive framework for training a learner using demonstrations provided by a teacher and that teacher’s assessments of task executions generated as a query by the learner. The learner must carry out two processes: learning, wherein the robot updates its belief conditioned upon labeled task executions; and planning, where it must use that belief to generate task executions. We adopt an iterative version of Bayesian specification inference proposed by Shah et al. [2] as the inference formulation, and extend it to allow both positive and negative examples (as elaborated upon in Section IV-C). Formally, if the learner’s initial belief over formulas is $P(\varphi)$, and the learner receives a dataset of task executions and their labels, $D = \{(\alpha, L([\alpha]))\}$, then the learner computes an estimate of the posterior distribution $P(\varphi \mid D)$. The learner updates its belief to be the computed posterior as follows:

$$P^{i+1}(\varphi) \leftarrow P(\varphi \mid D) \quad (3)$$

The learner has the ability to compute two types of policies depending upon the availability of a teacher to assess its task executions. If a teacher’s assessment is unavailable, the learner computes a policy to satisfy its current belief, $P^i(\varphi)$. (This is an instance of planning with uncertain specifications (PUnS) [6], briefly described in IV-D.) The original non-Markov planning problem is compiled into an equivalent MDP representation with a reward function representing the minimum regret criterion proposed by Shah et al. [6].

If a teacher’s assessments are available, the learner computes a policy to generate a task execution with the most uncertain degree of acceptability as per the learners current belief $P^i(\varphi)$. The teacher’s assessment of this task execution would be most beneficial in reducing the learner’s uncertainty of the true specification. We describe our approach to generating such an informative query in Section IV-E.

C. Bayesian Specification Inference

Bayesian specification inference [2] is a probabilistic model for using demonstrations provided by a teacher in order to infer LTL formulas corresponding to the task specifications [2]. According to this model, the hypothesis space of candidate LTL formulas comprises the set of formulas corresponding to the following template, which includes conjunctions of temporal behaviors identified by Dwyer et al. [30].

$$\varphi = \varphi_{global} \land \varphi_{eventual} \land \varphi_{order} \quad (4)$$

Shah et al. [6] also proposed a domain-independent approximation of the likelihood function $P([\alpha] \mid \varphi)$ – depending upon the number of conjunctive clauses – that satisfied the
size principle \[31\]. A restrictive hypothesis has greater likelihood than a less-restrictive hypothesis in presence of data conforming to both. We extend this to a case wherein the acceptability label \(L([\alpha])\) is provided along with an execution trace. Consider two candidate formulas \(\phi_1\) and \(\phi_2\) with \(N_{conj,1}\) and \(N_{conj,2}\) conjunctive clauses, and \([\alpha] = \phi_1\). If the trace is labeled as acceptable (\(L([\alpha]) = 1\)), the approximate likelihood odds ratio is computed as follows:

\[
P(\langle [\alpha], L([\alpha]) = 1 \rangle | \phi_2) = \frac{\frac{N_{conj,2}}{2^{N_{conj,2}}} \cdot [\alpha] \models \phi_2}{\frac{N_{conj,1}}{2^{N_{conj,1}}} \cdot [\alpha] \models \phi_2} \tag{5}
\]

This is identical to the likelihood function proposed by Shah et al. \[2\]; however if the trace is labeled as unacceptable (\(L([\alpha]) = 0\)), and \([\alpha] \not\models \phi_1\), the likelihood odds ratio is computed as follows:

\[
P(\langle [\alpha], L([\alpha]) = 1 \rangle | \phi_2) = \frac{\frac{N_{conj,2}}{2^{N_{conj,2} - 1}} \cdot [\alpha] \models \phi_2}{\frac{N_{conj,1}}{2^{N_{conj,1} - 1}} \cdot [\alpha] \models \phi_2} \tag{6}
\]

We assume that each data point in a given dataset \(D = \{\langle [\alpha], L([\alpha]) \rangle\}\) is independent of the others; thus, the likelihood of the entire dataset is the product of the individual likelihoods as follows:

\[
P(D | \varphi) = \prod_{\langle [\alpha], L([\alpha]) \rangle \in D} P(\langle [\alpha], L([\alpha]) \rangle | \varphi) \tag{7}
\]

The probabilistic model is implemented in webppl \[32\], a universal probabilistic programming language. The posterior is approximated using webppl’s Markov chain Monte Carlo algorithm with the Metropolis-Hastings acceptance criterion.

D. Planning with Uncertain Specifications

Planning with uncertain specifications (PUnS) \[6\] is a formulation for planning problems wherein task specifications are known as beliefs over LTL formulas \(P(\varphi)\). An instance of a PUnS problem is defined by the planning environment, which is encoded as an MDP sans a reward function, \(\mathcal{M}_X = \langle X, A, T_X \rangle\); a task specification represented as a belief over LTL formulas, \(P(\varphi)\), with support over a finite set of formulas, \{\varphi\}; and one of the four evaluation criteria proposed by Shah et al. \[6\] for satisfying a belief over LTL formulas.

In order to compute the policy, the PUnS instance is first compiled into a reward machine (\[24\]) corresponding to a Markov representation for \(P(\varphi)\), represented as a deterministic MDP, \(\mathcal{M}(\varphi) = \{(\varphi'), \{0, 1\}^{prop}, T_{\varphi}, R_{\varphi}\}\). \{(\varphi')\} is the set of ordered tuples \(\varphi'\) that represent all progressions of the formulas contained in \{\varphi\}, and the actions represent the truth values of the propositions, \(\alpha\). Let \(\varphi^i\) represent the \(i^{th}\) formulas in the tuple \(\langle \varphi' \rangle\); the transition function \(T_{\varphi}(\varphi_i, \varphi_2, \alpha)\) is then defined as follows:

\[
T_{\varphi}(\langle \varphi_1 \rangle, \langle \varphi_2 \rangle, \alpha) = \begin{cases} 
1 & \text{if } \varphi_2^i = \text{Prog}(\varphi^i, \alpha) \forall i \\
0 & \text{otherwise}
\end{cases} \tag{8}
\]

Let \(\langle \varphi' \rangle_{term}\) be the set of terminal states, where each of the component formula has either been satisfied (\(\top\)), dissatisfied (\(\bot\)), or has progressed to a safe-LTL formula. The reward function depends upon the choice of the PUnS evaluation criterion. For the minimum regret criterion, the reward function is defined as follows:

\[
R_{\varphi}(\langle \varphi' \rangle) = \begin{cases} 
\sum P(\varphi') r(\varphi^i) & , \text{if } \langle \varphi' \rangle \in \langle \varphi' \rangle_{term} \\
0 & , \text{otherwise}
\end{cases} \tag{9}
\]

where \(r(\varphi^i)\) is defined as follows:

\[
r(\varphi^i) = \begin{cases} 
1 & , \varphi^i = \top \text{ or } \varphi^i \in \text{safe-LTL} \\
-1 & , \varphi^i = \bot
\end{cases} \tag{10}
\]

This compiled deterministic MDP, \(\mathcal{M}(\varphi)\) is then composed with \(\mathcal{M}_X\) to obtain an MDP equivalent of the original PUnS problem, defined as follows:

\[
\mathcal{M}_{spec} = \langle X \times \{\varphi'\}, A, T_{spec}, R_{\varphi} \rangle \tag{11}
\]

Here,

\[
T_{spec}(\langle \varphi_1 \rangle, \langle \varphi_2 \rangle, a) = \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \ Quad
A task execution that ends in the state \( (G-T_0, G-T_0) \) or \( \langle \perp, \perp \rangle \) is not informative, as it would be labeled either acceptable or unacceptable according to both formulas. An informative query would reach the state \( \langle \perp, G-T_0 \rangle \). If this task execution were acceptable, then \( \varphi_2 \) is more likely to be the ground truth specification, conversely, if the task execution were judged unacceptable, then \( \varphi_1 \) is more likely to be the ground truth specification.

In general, for binary labels, uncertainty sampling queries the task execution whose probability of being acceptable is closest to 0.5 as per the learner’s current belief. Given a belief \( P(\varphi) \), the learner’s certainty of its acceptability of a demonstration trace \( [\alpha] \) is computed as follows:

\[
P(\hat{\mathcal{L}}([\alpha]) = 1) = \sum_{\varphi \in \mathcal{L}} 1([\alpha] \models \varphi)P(\varphi) \tag{13}
\]

The reward value according to the minimum regret criterion is a linear function of \( P(\hat{\mathcal{L}}([\alpha]) = 1) \), and; and \( P(\hat{\mathcal{L}}([\alpha]) = 1) = 0.5 \) corresponds to a reward value of 0. Therefore, given a reward machine \( \mathcal{M}_\varphi \), the most informative query as per the uncertainty sampling approach should end in a state defined as follows:

\[
\langle \varphi \rangle_{\text{selected}} = \arg\min_{\langle \varphi' \rangle \in \langle \varphi \rangle_{\text{term}}} |R_{\varphi}(\langle \varphi' \rangle)| \tag{14}
\]

Here, \( \langle \varphi \rangle \) is the set of terminal states of \( \mathcal{M}_\varphi \). Finally, in order to compute a policy for performing a task execution that terminates in \( \langle \varphi \rangle_{\text{selected}} \), we reshape the reward values of \( \mathcal{M}_\varphi \); Let \( \langle \varphi \rangle_{\text{path}} \) be the set of states that lie along any path joining the initial state, \( \langle \varphi \rangle \), and \( \langle \varphi \rangle_{\text{selected}} \); the reshaped reward function would then be defined as follows:

\[
R_{\text{shaped}}(\langle \varphi' \rangle) = \begin{cases} 
1 & , \langle \varphi' \rangle = \langle \varphi \rangle_{\text{selected}} \\
0 & , \langle \varphi' \rangle \in \langle \varphi \rangle_{\text{path}} \\
-1 & , \text{otherwise}
\end{cases} \tag{15}
\]

The reshaped reward \( R_{\text{shaped}}(\langle \varphi' \rangle) \) is indicated in blue for the example problem described in Figure 2. Note that this reward is only maximized when an execution terminates in \( \langle \varphi \rangle_{\text{selected}} \), and the learner is equally uncertain with regards to the acceptability of any task execution that terminates in \( \langle \varphi \rangle_{\text{selected}} \). The policy to generate an informative query execution can be computed by solving the MDP \( \mathcal{M}_{\text{spec}} = (\mathcal{X} \times \langle \langle \varphi' \rangle \rangle, \mathcal{A}, \mathcal{T}_{\text{spec}, \text{reshaped}}). \) (Note this is identical to \( \mathcal{M}_{\text{spec}} \) apart from reward function.)

V. Evaluations

We evaluated our proposed framework using both simulated experiment and a human-participant study. The simulated experiment used the synthetic environment proposed by Shah et al. [2] to rapidly generate scenarios with varying temporal specifications. We assessed the ability of our proposed framework to infer the correct LTL specifications compared with baselines as described in Section V-A and found that an active learning protocol within our framework generated posterior beliefs that were better aligned with the ground truth specification compared to learning purely from demonstrations or an interactive framework with randomly sampled queries.

We also designed a human-participant study centered around a similar protocol for a dinner table setting task; in order to evaluate our framework’s efficacy, and to determine whether the subjective perceptions of the participants aligned with the objective metrics. Our results indicate that contrary to the simulation experiments, learning only from demonstrations might outperform active learning on certain temporal tasks. (We discuss this further in Section V-C6.)

A. Baselines

To our knowledge, our proposed framework is first to model robot learning for non-Markov tasks that unifies demonstrations and acceptability assessments by the teacher. A natural baseline for our framework is the classical learning-from-demonstrations (LfD) framework, where the learner learns solely from demonstrations provided by the teacher. Second, we also wanted to evaluate the effect of query selection on the learning performance, therefore as a second baseline we generated the query executions by selecting actions at each time step from a uniform random distribution. Based on these three paradigms, we used the following three protocols:

1) **Active**: The teacher initially provides two demonstrations, followed by the learner generating queries. The learner’s belief over LTL formulas is updated after an assessment is provided for each of the queries. Each query is generated to reach an informative terminal state as defined by Equation 14. One the teacher answers a pre-defined number of queries, the task policy is computed using the final belief and the minimum regret criterion.
2) **Random**: The teacher initially provides two demonstrations; the learner then generates queries eliciting the teacher’s assessment. After each query, the learner’s belief is updated accordingly. In contrast to the **Active** condition, the queries in the **Random** condition are generated by uniformly sampling available actions at each time step. Once a pre-defined number of queries were assessed by the teacher, the task policy is computed using the final belief along and the minimum regret PUnS criterion.

3) **Batch**: In this condition the teacher only provides demonstrations, and the learner can not elicit any assessment on its task performance. The final belief is the poster distribution computed using Bayesian specification inference [6]. The task policy is computed using the final belief and the minimum regret PUnS criterion.

The number of task executions provided to the learner (as either demonstrations or queries) was equal in all cases.

### B. Simulation Experiments

The task environment for all the simulations was based on the synthetic domain proposed by Shah et al. [2]. This domain allows a variable number of threats and waypoints, where the admissible orders for visiting the waypoints are encoded within the ground truth formula LTL formula. We allowed a maximum of five waypoints and five threats for any simulation run. The available action space allowed the learner to select any of these 10 targets to visit.

For all runs of the simulation, the procedure was as follows:

1) Select the number of queries $n_{query}$
2) A ground truth LTL formula $\varphi$ was sampled from the priors proposed by Shah et al. [2].
3) Two (2) demonstrations were generated which satisfied the ground truth formula and added to the dataset $\mathcal{D} = \{ ([\alpha]_1, 1), ([\alpha]_2, 1) \}$
4) $\mathcal{D}$ was used with the **Active** protocol with $n_{query}$ queries generated by the learner. The final belief $P_{active}(\varphi)$ was recorded.
5) $\mathcal{D}$ was used with the **Random** protocol with $n_{query}$ queries generated by the learner. The final belief $P_{random}(\varphi)$ was recorded.
6) An augmented dataset $\mathcal{D}_{batch} = \mathcal{D} \cup \{ ([\alpha]_{2+i}, 1) : i \in \{1, \ldots, n_{query} \} \}$ was created by generating three additional demonstrations that satisfied the ground truth formula. The dataset was then used with the **Batch** protocol, and the final belief $P_{batch}(\varphi)$ was recorded. (This ensured the total number of task executions provided to all baselines was equal.)

The experiment was conducted for values of $n_{query} = \{1, \ldots, 6\}$, with 200 runs for each value. A different ground truth formula was sampled for each run. For every individual run, the entropy of the final belief, and similarity to the ground truth formula (as per the intersection-over-union metric proposed by Shah et al. [2]) were recorded for each of the training protocols.

### C. Human-Participant Study

Guided by the results of our simulation experiment, we designed a study involving human participants in order to evaluate the following hypotheses:

- The **Active** protocol will return final beliefs with a greater degree of similarity to the ground truth formula compared with either the **Batch** (H1) or **Random** (H2) protocols.
- The participants will prefer the **Active** protocol to the **Batch** (H3) and **Random** (H4) protocols.
The participant’s label was only recorded once the entire execution assessment after observing the robot as it executed the task. A placed on Table B. Participants were instructed to provide an informed that the objects could not be picked up again once to move only a single object at a time. They were also demonstrated by the participant or performed by the robot) the five executions performed by the robot. The robot’s final policy was computed by the participant or performed by the robot) the five demonstrations, while for Active and Random conditions, it involved providing two demonstration followed by assessing three query executions performed by the robot. The robot’s final policy was computed using tabular Q-learning to learn the policy for the MDP equivalent of a PUnS problem compiled with the minimum value is zero, when any formula in the support of the ground truth formula. The largest possible value for this metric is unity, when all the probability mass is associated with a formula that is equivalent to the ground truth formula, while the minimum value is zero, when any formula in the support of the distribution has no subformula in common with respect to the ground truth formula.

1) Experiment design: We utilized a within-subjects design with a single independent variable: the training protocol. There were three treatment conditions, Active, Batch, and Random. The order in which the participants experienced the protocol was counterbalanced using a $3 \times 3$ Latin square. Each condition was divided into a training and a testing phase. During the training phase, the participants had to teach the robot to set the dinner table through demonstrations and assessments of query executions performed by the robot. The training phase for the Batch condition involved providing five demonstrations, while for Active and Random conditions, it involved providing two demonstration followed by assessing three query executions performed by the robot. The robot’s final policy was computed using tabular Q-learning to learn the policy for the MDP equivalent of a PUnS problem compiled with the minimum regret evaluation criterion.

During the test phase, the robot performed three task executions using a stochastic policy to demonstrate its learning to the participant. At the end of the testing phase, we instructed the participants to complete a task questionnaire assessing whether the robot correctly learned the task. The participants received a worksheet to record the order in which they placed the objects, and the order in which the robot placed the objects.

2) Implementation details: Figure 4a depicts the experiment setup. During each task execution (whether demonstrated by the participant or performed by the robot) the five objects were initially placed on Table A, and subsequently arranged on Table B in a configuration depicted in Figure 4b. While providing demonstrations, we instructed the participants to move only a single object at a time. They were also informed that the objects could not be picked up again once placed on Table B. Participants were instructed to provide an assessment after observing the robot as it executed the task. A participant’s label was only recorded once the entire execution had been completed.

The state space of the robot $X$ was identical to the set of propositions required for evaluating the task $\alpha$, and contained five Boolean propositions, each of which encoded whether a particular object was successfully placed on the table. The robot’s action space, $A$, comprised five actions (one for each object). Initiating an action, triggered a sequence of parameterized primitives programmed into the robot to locate, pick up, and place the object on Table B. Based on the constraints provided to the participants and the robot’s action space, the only way to successfully complete the task was to ensure that the dinner plate, small plate, and the bowl were placed in that particular order. (The fork and the knife could be placed at any instant.)

3) Metrics: The objective metrics under consideration are computed from the final belief distributions $P_{\text{active}}(\phi)$, $P_{\text{random}}(\phi)$, and $P_{\text{batch}}(\phi)$ resulting from the training protocols. To evaluate $H1$ and $H2$ we computed the similarity score as per the intersection-over-union metric proposed by Shah et al. [2]. The largest possible value for this metric is unity, when all the probability mass is associated with a formula that is equivalent to the ground truth formula, while the minimum value is zero, when any formula in the support of the distribution has no subformula in common with respect to the ground truth formula.

The participants answered a questionnaire on completion of each training protocol. The questionnaire was designed to assess three metrics. First, the correctness of the robot’s learned policy was assessed through the following prompts:

1) The robot was a good learner.
2) The robots perceived accurately what my goals were.
3) I trusted the robot to place the objects in an appropriate order.

Second, the participant’s perception of the flexibility of the robot’s policy was assessed through the following prompts:

1) The robot learned multiple valid orderings for completing the task.
2) The robot copied my demonstrations.

The participant’s responses to these prompts were provided as a five-level balanced Likert-item. Finally, the participants scored the robot’s overall learning performance on a 10-point scale. We assessed H3 and H4 based on this overall score.

4) Results: We performed this study with 18 participants. We had to terminate our experiment protocol with 3 participants due to robot hardware failure. We report the results for 15 participants (10 male, 5 female, median age: 26 years). Seven participants reported prior experience with robots or other automation systems.

The effect of the training protocol on the similarity of the final distribution with respect to the true formula was statistically significant as evaluated by a Friedman test ($p < 0.01$), which yielded a value of 17.7 for the test statistic. Next we conducted pairwise comparisons between the Active and the Batch and Random protocols, using the one-tailed Wilcoxon signed-rank test. The Active protocol showed a statistically significantly better performance than the Random protocol ($p < 0.01$), thus providing evidence in support of H2. Surprisingly, the Batch protocol showed a statistically significantly better performance than the Active protocol ($p < 0.01$), thus providing evidence against H2. The average similarity values across participants for all the training protocols along with a 95% confidence interval are depicted in Figure 4c (orange bars). For the subjective metrics, the Friedman test on the overall score indicated no statistically significant difference in participants’ overall ratings (average scores were: Active: 9.00, Random: 8.60, Batch: 9.06).

5) Post-hoc analysis: The effect of the training protocol on the entropy of the final distribution was also statistically significant ($p < 0.01$). Pairwise comparisons through one-sided Wilcoxon rank-sum test also indicated that the Random protocol resulted in a belief with statistically significantly higher entropy than the Active ($p < 0.01$) and Batch ($p < 0.01$) protocols. The Batch protocol also resulted in a statistically significantly ($p = 0.011$) lower entropy than the Active protocol, indicating that the final belief represented a more certain distribution.

We also evaluated the consistency of the prompts for assessing the correctness of the robot and its flexibility in task execution. The responses to the three prompts for assessing the correctness of robot’s policy yielded a Cronbach’s alpha statistic value of $\alpha = 0.81$, and those for assessing the flexibility of robot’s policy yielded $\alpha = 0.97$ indicating that these Likert items can be added to create Likert scales. However, no statistically significant effect of training protocol was observed in any of the subjective evaluations.

Finally, we repeated the simulation experiment with the same ground truth formula as the table-setting task in an analog simulated environment. The mean similarity values resulting from 15 trials are depicted in Figure 4c (blue bars).

6) Discussion: The results of simulation experiments described in Section V-B suggest that an active learning protocol outperformed learning purely from demonstrations, and our study with human participants confirms that the Active protocol yielded belief distributions that were well aligned with the ground truth specifications (average similarity: 0.86 95% CI [0.82, 0.92]).

An unexpected outcome of the user study was that the Batch training protocol (average similarity: 0.93 95% CI [0.88, 0.98]) outperformed the Active protocol (average similarity: 0.86 95% CI [0.82, 0.92]) for learning the table-setting task. To investigate further, we repeated the simulation experiment while trying to emulate the conditions of the user-study as closely as possible. We ran the simulation experiment with $n_{query} = 3$, with the same ground truth formula as the table-setting task for 15 runs. The results (depicted in Figure 4c) indicated that Batch protocol is not expected to perform worse than the Active protocol for the ground truth formula corresponding to the table setting task; however, as described in Section V-B the Active protocol will outperform the Batch protocol on a wider gamut of ground truth formulas. This motivates further analysis and experiments to assess the relative performance of the training protocols with the task itself being one of the independent variables.

The results of the user-study were enlightening in identifying future research avenues. For the simulation experiments we assumed that demonstrations provided by the teacher were generated independently; however, open-ended comments from participants indicate that they tried to deliberately provide varied examples (“... I attempted to show the robot 5 different permutations of the objects, in order to enforce the order of certain elements as critical (e.g., dinner plate must be placed before small plate), and to showcase that others were not (e.g., fork may come before knife, or vice versa...”). Further research into how how humans provide varied demonstrations would also be of value in guiding how the robot should communicate it’s learning to the teacher.

The participants also indicated that information in addition to just the query execution would be more helpful in assessing the robot’s learning (“... during the querying phase, perhaps my trust in the robot would have increased if it produced some sort of confidence score alongside each query...”; “... unable to understand why the robot took more time at some points than others...”; “... how does the robot decide the order?...”). This motivates investigation into algorithms to summarize the information that the robot intends to gain from a query.

VI. Conclusion

In conclusion, we proposed an interactive training framework capable of learning non-Markov tasks from both demonstrations provided by a teacher, and that teacher’s assessments of robot’s task executions. We further proposed a querying algorithm that allows the learner to identify and perform a task execution with the most uncertain degree of acceptability within an active learning framework based on the principle of uncertainty sampling. Through simulation experiments we demonstrated that our proposed active learning framework outperforms both learning purely from demonstration, and an interactive learning protocol with randomly generated query
executions for a range of ground truth specifications. Finally, we demonstrated our active learning frameworks efficacy at learning to set a dinner table through a user study; however, our results indicated that the relative performance of training protocols is dependent on the temporal structure of the task that a learner must learn to perform.

REFERENCES

[1] Amir Pnueli. The temporal logic of programs. In Foundations of Computer Science, 1977., 18th Annual Symposium on, pages 46–57. IEEE, 1977.

[2] Ankit Shah, Prithish Kamath, Julie A Shah, and Shen Li. Bayesian Inference of Temporal Task Specifications from Demonstrations. In Advances in Neural Information Processing Systems 31, pages 3804–3813, 2018.

[3] Joseph Kim, Christian Muise, Ankit Shah, Shubhham Agarwal, and Julie Shah. Bayesian inference of linear temporal logic specifications for contrastive explanations. In IJCAI, 2019.

[4] Yoonseon Oh, Roma Patel, Thao Nguyen, Baichuan Huang, Ellie Pavlick, and Stefanie Tellex. Planning with state abstractions for non-Markovian task specifications. In RSS, June 2019. doi: 10.15607/RSS.2019.XV.059.

[5] Nakul Gopalan, Dilip Arumugam, Lawson LS Wong, and Stefanie Tellex. Sequence-to-sequence language grounding of non-markovian task specifications. In Robotics: Science and Systems, 2018.

[6] Ankit Shah, Shen Li, and Julie Shah. Planning with uncertain specifications (PU2S). IEEE Robotics and Automation Letters, 2020.

[7] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. Robotics and autonomous systems, 57(5):469–483, 2009.

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

[9] Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob Fosser, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19, pages 6309–6317. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/980. URL https://doi.org/10.24963/ijcai.2019/980.

[10] Andrea Bajcsy, Dylan P Losey, Marcia K O’Malley, and Anca D Dragan. Learning robot objectives from physical human interaction. Proceedings of Machine Learning Research, 78:217–226, 2017.

[11] Andrea Bajcsy, Dylan P Losey, Marcia K. O’Malley, and Anca D. Dragan. Learning from physical human corrections, one feature at a time. In Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction, HRI ’18, page 141–149, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450349536. doi: 10.1145/3171221.3171267. URL https://doi.org/10.1145/3171221.3171267.

[12] Anca D. Dragan Dorsa Sadigh, Shankar Sastry, and Sanjit A Seshia. Active preference-based learning of reward functions. In Robotics: Science and Systems (RSS), 2017.

[13] Erdem Biyik and Dorsa Sadigh. Batch active preference-based learning of reward functions. In Conference on Robot Learning, pages 519–528, 2018.

[14] Erdem Biyik, Malayandi Palan, Nicholas C. Landolfi, Dylan P. Losey, and Dorsa Sadigh. Asking easy questions: A user-friendly approach to active reward learning. In 3rd Conference on Robot Learning (CoRL), October 2019.

[15] Marcell Vazquez-Chanalatte, Susmit Jha, Ashish Tiwari, Mark K Ho, and Sanjit Seshia. Learning task specifications from demonstrations. In Advances in Neural Information Processing Systems 31, pages 5368–5378, 2018.

[16] Daniel Kasenberg and Matthias Scheutz. Interpretable apprenticeship learning with temporal logic specifications. In IEEE Conference on Decision and Control, 2017.

[17] Alberto Camacho and Sheila A. McIlraith. Learning interpretable models in linear temporal logic. In Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling (ICAPS), pages 621–630, 2019.

[18] Maya Cakmak and Andrea L Thomaz. Designing robot learners that ask good questions. In Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction, pages 17–24. ACM, 2012.

[19] Maya Cakmak, Crystal Chao, and Andrea L Thomaz. Designing interactions for robot active learners. IEEE Transactions on Autonomous Mental Development, 2(2):108–118, 2010.

[20] Hadas Kress-Gazit, Georgios E Fainekos, and George J Pappas. Temporal-logic-based reactive mission and motion planning. IEEE transactions on robotics, 25(6):1370–1381, 2009.

[21] Alberto Camacho, Jorge A Baier, Christian Muise, and Sheila A McIlraith. Finite LTL synthesis as planning. In Twenty-Eighth International Conference on Automated Planning and Scheduling, 2018.

[22] Alberto Camacho and Sheila A. McIlraith. Strong fully observable non-deterministic planning with LTL and LTL-f goals. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI), pages 5523–5531, 2019.

[23] Michael L Littman, Ufuk Topcu, Jie Fu, Charles Isbell, Min Wen, and James MacGlashan. Environment-independent task specifications via GLTL. arXiv preprint arXiv:1704.04341, 2017.

[24] Rodrigo Toro Icarte, Torny Q Klassen, Richard Valenzano, and Sheila A McIlraith. Teaching multiple tasks to an RL agent using LTL. In Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, pages 452–461. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[25] Rodrigo Toro Icarte, Torny Klassen, Richard Valenzano, and Sheila McIlraith. Using reward machines for high-level task specification and decomposition in reinforcement learning. In International Conference on Machine Learning, pages 2112–2121, 2018.

[26] Alberto Camacho, R Toro Icarte, Torny Q Klassen, Richard Valenzano, and Sheila A McIlraith. LTL and beyond: Formal languages for reward function specification in reinforcement learning. In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI), pages 6065–6073, 2019.

[27] Fahiem Bacchus and Froduald Kabanza. Using temporal logics to express search control knowledge for planning. Artificial intelligence, 116(1-2):123–191, 2000.

[28] Zohar Manna and Amir Pnueli. A hierarchy of temporal properties. Department of Computer Science, 1987.

[29] Christopher JCH Watkins and Peter Dayan. Machine learning, 8(3-4):279–292, 1992.

[30] Matthew B Dwyer, George S Avrunin, and James C Corbett. Patterns in property specifications for finite-state verification. In Proceedings of the 21st international conference on Software engineering, pages 411–420. ACM, 1999.

[31] Joshua B Tenenbaum. Rules and similarity in concept learning. In Advances in neural information processing systems, pages 59–65, 2000.

[32] Noah D Goodman and Andreas Stuhlmüller. The Design and Implementation of Probabilistic Programming Languages. [http://dippl.org]. 2014. Accessed: 2018-4-9.
[33] Burr Settles. Active learning literature survey. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 2009.

[34] David D Lewis and Jason Catlett. Heterogeneous uncertainty sampling for supervised learning. In Machine learning proceedings 1994, pages 148–156. Elsevier, 1994.