Learning Constraints on Autonomous Behavior from Proactive Feedback

Connor Basich*, Sadduddin Mahmud*, and Shlomo Zilberstein

***Abstract*— Learning from feedback is a common paradigm to acquire information that is hard to specify a priori. In this work, we consider a planning agent with a known nominal reward model that captures their high-level task objective, but is subject to constraints that are unknown *a priori* and must be inferred from human interventions. Unlike existing methods, our approach does not rely on full or partial demonstration trajectories or assume a fully reactive human. Instead, we assume access only to *sparse interventions*, which may in fact be generated *proactively* by the human, and make only minimal assumptions about the human. We provide both theoretical bounds on performance, and empirical validations of our method. We show that our method enables an agent to learn a constraint set with high accuracy that generalizes well to new environments within a domain, whereas methods that only consider reactive feedback learn an incorrect constraint set that does not generalize well, making constraint violations more likely in new environments.**

I. Introduction

Although great efforts are being made towards deploying autonomous robotic systems in the open world, it is widely recognized that designing systems *a priori* with perfect models of the world is generally infeasible due to practical and theoretical limitations [1], [2]. These limitations can arise from a lack of sufficient training data [3], limited prior domain information [4], or divergence between the training environment and deployment setting [5]. To handle this challenge, many techniques have been developed to enable an agent to expand or refine their models online [6], [7].

Consider a planning agent with a known nominal model of the domain and a reward function that captures their nominal task description. Unknown to the agent *a priori* is a set of constraints that captures more nuanced aspects of the domain such as user preferences or safety concerns. We propose a constraint learning method for the agent that infers such constraints from *sparse human intervention* in a supervised setting. We consider this supervise-and-intervene setting as it is a natural teaching framework for the human since interventions are easy to provide and, assuming a reasonable level of understanding by the human, also provide useful information to the agent as an intervention is often an indicator of a potential constraint being violated [8].

We choose an intervention-based approach because learning from demonstrations limits the setting to those where the human can manually operate the system to generate the demonstrations, which even when possible can be costly and may require greater expertise than simply intervening. Likewise, a query-for-information framework may be inefficient and onerous to the human if the agent frequently queries for the existence of a non-existent constraint, or unsafe if the agent fails to query in the presence of possible constraint violations. Prior work in learning from interventions [8], [9], [10], [11], [12] has assumed that feedback is provided either reactively or retroactively, but has ignored the possibility of *proactive* feedback.

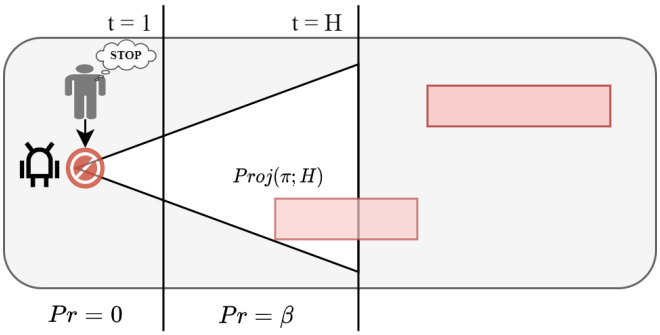

Fig. 1: Proactive feedback provided at time $t = 1$ is conditioned on the likelihood of a constraint violation (red) occurring during $t = 1 : H$. $Proj(\pi)$ indicates the H-step projection of policy π.

However, growing evidence suggests that humans' cognitive control processes are generally either reactive in response to rapid or unexpected changes in their external contexts, or proactive as a means of strategically optimizing behavior resulting from anticipated goal-relevant interference that they aim to ameliorate before occurring [13], [14]. Consequently, we assume that human interventions can either be reactive — intended for the current state and action of the agent — or proactive — conditioned on the possibility of inferred future constraint violations by the agent given the human's understanding of the agent's behavior (see Fig. 1).

Our approach makes minimal assumptions about the human, but enables the agent to iteratively learn a constraint model and intervention model from intervention data using a data aggregation approach [3]. Additionally, we use uncertainty-based incentives to balance exploration and exploitation during training time to improve convergence. We show that our method enables an agent in a domain with an accurate nominal task model to efficiently learn a set of unspecified constraints from sparse, proactive feedback that generalizes to novel environments in the domain. Furthermore, we show that failing to account for proactivity in human feedback can lead an agent to infer an incorrect constraint set and perform poorly in new environments.

*Authors contributed equally.

University of Massachusetts Amherst, College of Information and Computer Science, Amherst, MA, USA. {cbasich, smahmud, shlomo}@cs.umass.edu

II. Related Work

The general area of learning from human feedback is broad and extensive, including action selection guidance [15], [16], demonstrations [17], [5], labels [3], [18], [19], [20], [21], and implicit feedback [22], [23], [24]. Consequently, we focus our discussion of related work on the problem of learning constraints from feedback (our objective) and learning from intervention (our problem setting).

A. Constraint Learning

Learning or inferring constraints from feedback is often motivated by the belief that, for many problems, it is easier to infer a set of constraints on allowable behavior from feedback that dictates what the agent can't do, rather than attempting to learn an explicit reward function that best explains the observed human feedback or even an entire policy. Additionally, constraints are often shared across many tasks and environments within a domain, which is useful for generalization [25]. This problem has been studied rather extensively in the context of learning from demonstrations.

Scobee and Sastry [26] presented a method for maximum likelihood constraint inference in an inverse reinforcement learning setting that iteratively infers the constraints that best explain behavior observed in human demonstrations. Their work focused on purely deterministic systems, but was later extended to non-deterministic systems [27] and continuous, model-free settings [28]. Papadimitriou et al. [29] introduced a Bayesian constraint inference method based on human demonstrations that, unlike maximum likelihood methods, is able to work with both partial trajectories and sets of disjoint state-action pairs, in addition to the full demonstration trajectories used in maximum likelihood inference.

To the best of our knowledge, our work is the first to explicitly study both the problem of learning constraints from interventions specifically, as well as learning from proactive feedback. Spencer et al. [8] considered a similar problem in their *expert intervention learning* framework, which leveraged demonstrations, generated during *interventions* gated by a human supervisor's decision, to learn a portion of the state-action space that is deemed "good enough" to operate within. The agent's objective is consequently to minimize the time spent outside of the "good enough" region while simultaneously minimizing misclassification of intervention actions. However, their approach still relies on partial trajectories of human behavior and assumes non-proactive interventions.

B. Learning from Interventions

The problem of learning from interventions has been notably less studied in the inverse reinforcement learning literature. Bajcsy et al. [9] introduced a methodology in which a robot can learn an objective parameterized by a set of features via physical corrections made by a human supervisor, focusing on learning one feature at a time in order to reduce unintended learning from the human's interventions. Li et al. [12] and Liu et al. [11] both consider human-in-the-loop IRL settings in which an agent learns via reward signals generated by the environment to act in its domain while being constantly supervised by human operators who are ready and capable of intervening and taking control when the agent attempts something risky, providing a subsequent demonstration trajectory. As with the work done by Spencer et al. [8], these approaches differ from what is considered in this paper primarily in that they rely on human demonstration trajectories, only consider reactive human interventions, and also assume that the agent does not have an *a priori* well-specified nominal reward function.

In the model-based planning literature, Basich et al. [30] proposed a learning from intervention framework for an agent with a specified nominal model to learn their *competence* in the form of the optimal level of autonomy (LoA) to utilize. As in this paper, they do not consider demonstration availability (although they do allow for human control of the system) for their learning problem, but, unlike in this work, they restrict their interventions to be purely reactive. Additionally, the model they learn is purely a predictive model of the human's feedback, and not a constraint set as is considered here. Saisubramanian et al. [2] proposed a lexicographic planning problem [31] for an agent that is aware of its primary objective function, but lacks knowledge of a secondary cost function that it seeks to learn online. They consider several forms of human feedback, including interventions from an oracle who upon intervention specified a new action for the agent to take instead. As above, their work does not consider proactive feedback, and makes strong assumptions about the information that the supervisor has with respect to the agent's model.

III. Preliminaries

A Markov decision process (MDP) is represented by the tuple $\langle S, A, T, R\rangle$ where S is a finite set of states, A is a finite set of actions, $T : S \times A \times S \to [0, 1]$ is a transition function representing the probability of arriving in state s' having taken the action a in state s, and $R : S \times A \to \mathbb{R}$ is a reward function representing the immediate expected reward of taking action a in state s.

A solution to an MDP is a *policy*, denoted $\pi : S \to A$, which maps states to actions. A policy π induces the state-value function $V^\pi : S \to \mathbb{R}$, defined as $V^\pi(s) = R(s, \pi(s)) + \sum_{s' \in S} T(s, \pi(s), s')V^\pi(s')$, which represents the expected cumulative reward when starting in the state s and following the policy π. The objective is to find an optimal policy, i.e. $\pi^* \in \text{argmax}_{\pi \in \Pi} V^\pi(s)$.

In multi-objective sequential decision making, there are multiple competing objectives that must be considered when producing an optimal policy. There are two primary ways of modeling multi-objective problems that we utilize in this paper, and we define them below.

A multi-objective MDP (MOMDP) [32] is an MDP with multiple objectives that are optimized up to a weighted sum over the objective reward functions. Formally, a MOMDP is represented by the tuple $\langle S, A, T, \mathbf{R}, f_\mathbf{w}\rangle$ where S, A, and T are the same as in an MDP, $\mathbf{R} = [R_1, ..., R_k]^T$ is a vector of reward functions $R_i : S \times A \to \mathbb{R}$, and $f_\mathbf{w} : \mathbb{R}^k \to \mathbb{R}$ is a weighting function parameterized by

a vector $\mathbf{w}$. In a MOMDP, the weighting function produces a single value function of the form, $V^\pi_{\mathbf{w}}(s) = f_{\mathbf{w}}(\mathbf{V}^\pi(s))$ where $\mathbf{V}^\pi(s) = [V^\pi_1(s), ..., V^\pi_k(s)]$ is a vector of value functions computed as above under the policy π for each of the k objectives. We consider a *linear scalarization* method in which $f_{\mathbf{w}}(\mathbf{V}^\pi(s)) = \mathbf{w}^T\mathbf{V}^\pi(s)$ where each element $w_i \in \mathbf{w}$ is a positive real number, and $\mathbf{w}$ sums to 1.

A lexicographic MDP [31] is an MDP with competing objectives defined in a lexicographic order where each objective is optimized in order of the lexicographic ordering up to some slack value. Formally, an LMDP is represented by the tuple $\langle S, A, T, \mathbf{R}, \mathbf{\Delta}, o\rangle$ where S, A, T, and $\mathbf{R}$ are the same as above, $\mathbf{\Delta} = \langle \delta_1, ..., \delta_{k-1}\rangle$ is a tuple of *slack* variables with $\delta_i \in \mathbb{R}^+$ for every $i \in [k]$, and o is a strict ordering over the k objectives. Here, δ_i denotes the maximum allowable deviation from the optimal expected value for objective o_i when optimizing for objective o_{i+1}. Solving the LMDP involves sequentially optimizing for each objective in the lexicographic order given the slack by constraining the available actions as $A_{i+1}(s) = \{a \in A | \max_{a' \in A_i} Q_i(s, a') - Q_i(s, a) \leq \gamma_i\}$.

IV. Problem Formulation

Let $M = \langle S, A, T, R\rangle$ be an MDP that represents the nominal domain of the agent, which, for reward R, induces what we refer to as the *nominal objective* (o_R) that the agent aims to maximize. In other words, the nominal objective aims to find the policy, π that maximizes the cumulative reward under R,

$$\underset{\pi \in \Pi}{\operatorname{argmax}} \mathbb{E}\Big[\sum_{s \sim d_\pi} R(s, \pi(s))\Big]. \tag{1}$$

However, there exists a set, $C \subset S \times A$, of *constraints* that is unknown to the agent *a priori* that describes the state-action pairs that the agent is disallowed from performing (or, more generally, are undesirable). The constraint set induces an additional objective (o_C), which aims to find the policy that minimizes the expected number of constraint violations,

$$\underset{\pi \in \Pi}{\operatorname{argmin}} \mathbb{E}\Big[\sum_{s \sim d_\pi} C(s, \pi(s))\Big], \tag{2}$$

where, for clarity, we let $C : (s, a) \to \mathbb{I}[(s, a) \in C]$.

Because the constraint set is unknown *a priori*, the agent must learn the constraint set from proactive feedback provided by a human expert. The human is modeled by the tuple, $\mathcal{H} = \langle \epsilon, h, \tau\rangle$ where $\epsilon \in (0, 1]$ models the degree of the human's understanding of the agent's behavior, $h \in \mathbb{Z}^+$ is a temporal horizon, and β is an intervention temperature parameter.

Under ϵ, we can compute a policy-like function that we call the *corrupted observer policy*, $\hat{\pi}$, which models the human's lookahead belief over the agent's behavior $i \geq 1$ steps into the future from state s_t, as

$$\hat{\pi}(s_{t+i}) = \begin{cases} \pi(s_{t+i}) & \text{w.p. } 1 - \epsilon \\ \sim U(A \setminus \{\pi(s_{t+1})\}) & \text{w.p. } \epsilon \end{cases} \tag{3}$$

For the current time step, where $i = 0$, we assume that the human knows exactly the action that the agent is performing, ensuring that they can intervene in any constraint violation while supervising the agent. Intuitively, this represents the human's error in their model of the agent's behavior.

Given $\mathcal{H}$ and $\hat{\pi}$, we define the human intervention function $I(s_t | h, \beta, \hat{\pi}, C) \to \{0, 1\}$, which represents the binary decision of the human to override the system, or not, in state s_t given horizon h, temperature parameter β, corrupted observer policy $\hat{\pi}$ and the true constraint set C. In general, it is expected that it will be proportional to the expected number of constraint violations in the h-term future horizon starting at state s_t given policy $\hat{\pi}$:

$$I(s_t | h, \beta, \hat{\pi}, C) \propto \mathbb{E}_{s \sim \tau(s_t, \hat{\pi}, h)}\Big[C(s, \hat{\pi}(s))\Big] \tag{4}$$

where $\tau(s_t, \hat{\pi}, h)$ represents the full set of h-step trajectories under $\hat{\pi}$ starting from state s_t.

More generally, we may only have access to some distribution over β, D_β, and distribution over h, D_h; in which case, we get that

$$I(s_t | D_h, D_\beta, \hat{\pi}, C) = \int_{\beta \sim D_\beta} \int_{h \sim D_h} I(s_t | h, \beta, \hat{\pi}, C). \tag{5}$$

While the agent is still learning C, it maintains an estimate, $\hat{C} : S \times A \to [0, 1]$, from which we can compute an approximate intervention function by replacing C with $\hat{C}$ in Eqn. 4.

A. Constraint Learning

In this paper, we consider a *train-then-deploy* setting where the agent has a fixed amount of training time available during which it must learn the constraint set to the best of its ability. Once the training time is over, the agent is deployed into an unsupervised setting where it must optimize its nominal task while adhering to its learned constraint set. Extending the proactive feedback model to the learning-on-the-go setting introduces challenging complexities that are beyond the scope of this paper, but a discussion of them can be found in Section VII.

1) Training: During training, we add a secondary, information-theoretic reward for interventions, E, to provide an exploration utility to the agent for learning more of its constraint set. Consequently, during training we model the problem as a MOMDP where the primary objective is the agent's nominal task objective, and the secondary objective is to learn its constraint set. In general, one can also include a penalty, I, for interventions as well; however, we associate no cost to an intervention during training as it is the job of the human to supervise and train the agent. Formally, we define $M^T = \langle S, A, T, \begin{bmatrix} R & I & E \end{bmatrix}^T, \begin{bmatrix} w_R & w_I & w_E \end{bmatrix}\rangle$.

2) Deployment: During deployment, the agent does not have a human that it can rely on to provide it safety. Hence, we model the problem as an LMDP where the primary objective is to minimize the probability of violating a constraint, and the secondary objective is to maximize the nominal task reward. Formally, define $M^D = \langle S, A, T, \begin{bmatrix} -C & R \end{bmatrix}^T, \langle \delta \rangle, [o_C, o_R]\rangle$ where $\delta \in \mathbb{R}^+$ is the maximum slack allowed from the minimal probability of constraint violation.

Algorithm 1: MB-CLPF

Input: Domain Model D, Horizon h, Sample K, Exploration Bonus Function E, Reward Weight $\mathbf{w}$
Result: Constraint Model C_{θ_1}

1 Initialize Constraint Model C_{θ_1}
2 Initialize Intervention Model I_{θ_2}
3 $\pi_0 \leftarrow ComputePolicy(D/R, R)$
4 $\mathcal{D} \leftarrow \emptyset$
5 **for** $i = 0 : \infty$ **do**
6 $\quad \mathcal{F}_i \leftarrow CollectInterventionFeedback(\pi_i, K)$
7 $\quad \mathcal{D} \leftarrow D \cup (\mathcal{F}_i, \pi_i)$
8 $\quad Train(C_{\theta_1}, I_{\theta_2}; \mathcal{D}, h)$
9 $\quad R^T \leftarrow GetTrainingReward(D, R, \hat{I}, E, \mathbf{w})$
10 $\quad \pi_{i+1} \leftarrow CalculatePolicy(D/R, R^T)$
11 $\quad$ **if** Termination Condition is Met
12 $\quad\quad$ BREAK
13 **return** C_{θ_1}

V. Methodology

In this section, we describe our proposed method for learning constraints from spare binary feedback signals in the form of proactive interventions. We start by describing the agent's training loop using a fixed known horizon **h** for ease of understanding. Later, we relax this requirement and provide a more general solution.

A. Training Loop

The algorithm starts by initializing the constraint model, C_{θ_1}, and the intervention model, I_{θ_2}, randomly (line 1-2). Here, models are represented by two feedforward RELU neural networks parameterized by θ_1 and θ_2 respectively. Next, the agent's policy, π_0, is initialized using the policy under the nominal reward model R (line 3). Lastly, the dataset $\mathcal{D}$ that is used for training C_{θ_1} and I_{θ_2} is initialized to $\emptyset$. After initialization, C_{θ_1} and I_{θ_2} are iteratively updated through an interactive learning process (line 5-11), as detailed next.

The i-th iteration of the training loop starts by collecting a set of feedback, $\mathcal{F}_i$, using human supervision and the agent's current policy π (line 6). Feedback is provided as sparse, binary signals, which come in the form $\langle s, f \rangle$, where the state s is labeled by $f \in \{not\ intervened(0), intervened(1)\}$. The set of feedback signals, $\mathcal{F}_i$, along with current policy, π_i, are added to the existing dataset $\mathcal{D}$ (line 7). D is then used to jointly train C_{θ_1} and I_{θ_2} using gradient-based optimization (line 8). Based on the updated C_{θ_1} and I_{θ_2}, a new training reward R^T is calculated using the nominal reward R, inferred intervention set $\hat{I}$, the information-theoretic exploration bonus E, and the linearization weight $\mathbf{w}$ (line 9). Finally, the policy is updated using the new training reward R^T (line 10). The training loop is continued until terminating conditions are met. There are many possible choices for terminating conditions including using a fixed number of iterations, the total number of interventions during feedback collection, or the amount of change in constraint set prediction. Finally, the learned constraint model, C_{θ_1}, is returned.

B. Constraint Learning

Our core assumption is that the probability of a human intervention in a state s_j is a function of the expected number of constraint violations in the future from the current state s_j. We define this value as

$$CV^h_{\pi_i}(s_j) = \mathbb{E}_{\tau \sim \pi_i(s_j)} \left[\sum_{t=0,(s_t,a_t)\in\tau}^{t=h} C(s_t, a_t) \right]. \quad (6)$$

To estimate this value, we learn a constraint model, C_{θ_1}, parameterized by θ_1, under which we estimate the probability of a state-action pair (s_j, a_j) being constrained as

$$C^{a_j}_{s_j} = P((s_j, a_j) \in C) \approx \sigma(C_{\theta_1}(s_j, a_j)) \quad (7)$$

where $\sigma(\cdot)$ is n sigmoid function. Hence, $CV^h_{\pi_i}(\cdot)$ can be estimated as follows:

$$CV^h_{\pi_i}(s_j) \approx \mathbb{E}_{\tau \sim \pi_i(s)} \left[\sum_{t=0,(s_t,a_t)\in\tau}^{t=h} C^{a_t}_{s_t} \right]. \quad (8)$$

Next, we use this value to estimate the probability of intervention at state s_j using a learned function, I_{θ_2}, parameterized by θ_2; specifically:

$$\hat{f}_j = P(f_j = 1 | s_j) \approx \sigma(I_{\theta_2}(CV^h_{\pi_i}(s_j))). \quad (9)$$

To optimize both θ_1 and θ_2 we use two different loss functions. First, we use a binary cross-entropy loss to train both $C_{\theta 1}$ and $I_{\theta 2}$ using the intervention feedback:

$$\mathcal{L}_f(\theta_1, \theta_2) = -\frac{1}{|D|} \sum_{\langle s_j, f_j \rangle \in D} f_j \log(\hat{f}_j) + (1 - f_j) \log(1 - \hat{f}_j). \quad (10)$$

However, as we do not have a ground truth label for constraint classification, in order to ensure that C_θ learns the correct representation, we want to enforce the following constraint:

$$\mathop{\mathbb{E}}_{\langle s_j, f_j \rangle \in D_1} [CV^h_{\pi_i}(s_j)] \quad > \mathop{\mathbb{E}}_{\langle s_j, f_j \rangle \in D_0} [CV^h_{\pi_i}(s_j)]. \quad (11)$$

Here, $D_1 = \{\langle s_j, f_j \rangle \in D | f_j = 1\}$ and $D_0 = \{\langle s_j, f_j \rangle \in D | f_j = 0\}$. Generally, we expect to see more constraint violations after an intervention state. This constraint can naturally be enforced by a negative log_softmax loss. Specifically,

$$\mathcal{L}_c(\theta_1) = -\log\left(\frac{exp(J_1)}{exp(J_1) + exp(J_0)}\right) \quad (12)$$

where

$$J_1 = \mathop{\mathbb{E}}_{\langle s_j, f_j \rangle \in D_1} [CV^h_{\pi_i}(s_j)] \quad (13)$$

and

$$J_0 = \mathop{\mathbb{E}}_{\langle s_j, f_j \rangle \in D_0} [CV^h_{\pi_i}(s_j)]. \quad (14)$$

Intuitively, this loss goes down as J_1 gets larger than J_0.

C. Ensemble Learning

As it has been shown that sigmoid-based classification often under or overestimates uncertainty [33], in order to get a better estimate of the uncertainty of our model, we use the ensemble method [34]. To train each model in the ensemble, in addition to sampling different subsets of the dataset, we also use a different horizon, h, sampled from $[1, h_{max}]$ according to distribution $\mathcal{D}_h$. We use a weighted voting scheme to estimate the ensemble. Specifically, we calculate the probability of a constraint violation, $C_{s_j}^{a_j}$, as

$$C_{s_j}^{a_j} = \frac{\sum_k w_k \sigma(C_{\theta_1^k}(s_j, a_j))}{\sum_k w_k}, \tag{15}$$

where $w_k \propto Accuracy(I_{\theta_2^k})$. Intuitively this puts more weight on models in the ensembles that better explain the observed data, and also reduces the reliance on knowing the exact horizon h.

Finally, we use an uncertainty measurement to improve constraint learning during the training phase. Specifically, we define an information-theoretic exploration bonus, $E : S \times A \to \mathbb{R}$, as

$$E(s_j, a_j) = \mathbf{H}(P(constraint|(s_j, a_j))). \tag{16}$$

where $\mathbf{H}$ is the Shannon entropy. The purpose of this exploration bonus is to encourage the agent to explore parts of the state-action space where the agent is uncertain about its constraint prediction. Based on this, we construct a multi-objective reward, $R^T = \begin{bmatrix} R & \hat{I} & E \end{bmatrix}^T$, where $\hat{I}(s,a) = -\mathbb{I}[I_{\theta_2}(s,a) > \mu]$ for some classification threshold $\mu \in [0,1]$ (usually set to 0.5), and solve the problem as a MOMDP with linear weights, $\mathbf{w} = \begin{bmatrix} w_R & w_I & w_E \end{bmatrix}$. Using the weight parameters w_R and w_E we can balance between exploitation of the nominal task reward and exploration (of constraints) during training, particularly in settings where constraint learning is not the only reason for training. The weight w_I, which is assumed to be 0 in our work, can also be tuned to balance the agent's risk-aversion during training.

D. Deployment Phase

After the training phase is completed, we utilize our learned models in a deployment phase wherein we assume that there is no human that can intervene and protect the robot. We construct a multi-objective reward, $R^D = \begin{bmatrix} -\hat{C} & R \end{bmatrix}$, where $\hat{C}$ and R are as defined above (as the true constraint set is not known); note that we remove the entropy-based exploration bonus during the deployment phase as there is no longer a human who can provide intervention signals. We consider the lexicographic ordering, $\mathbf{o} = [o_C, o_R]$, aiming to optimize first the constraint-based objective, and the nominal task objective second given some slack $\delta \in \mathbb{R}^+$ on objective o_C.

Additionally, we can modify the classification threshold μ to control how conservative the agent should be during deployment about predicting possible constraints. For example, setting μ to be very small will cause the agent to avoid a state-action pair if there is even a small probability that the state-action pair is constrained.

Finally, we would like to bound the worst-case performance of our model in deployment given what is learned by the learning algorithm. Below, we show that if our learned constraint set is at most α inaccurate, then the expected performance of the optimal policy computed for the learned constraint set will be at most a constant factor worse than the optimal policy computed for the true constraint set, assuming a finite horizon, T, for the problem.

Theorem 1. Let $\hat{\pi}^*$ be the optimal policy given the learned constraint set $\hat{C}$ and π^* be the optimal policy given the true constraint set C, and let $T \in \mathbb{N}$ be a maximum horizon for the problem. Denote by V_c^π and $V_{\hat{c}}^\pi$ the value functions induced by the policy π under constraint sets C and $\hat{C}$ respectively for the primary objective (see Eq. 2) within the horizon T. If $\hat{C}$ is at least $(1-\alpha)$-accurate, i.e.,

$$\frac{\sum_{(s,a)\in S\times A}[\hat{C}(s,a) == C(s,a)]}{|S||A|} \geq 1 - \alpha,$$

then $\max_{s\in S} V_c^{\hat{\pi}^*}(s) \leq \frac{1+\alpha T}{1-\alpha T} V_c^{\pi^*}(s)$.

Proof. For notational clarity, we write, e.g., V_c^π in stead of $V_c^\pi(s)$, understanding that our proof is with respect to the maximal difference in value functions over all states. First, observe that if we fix a policy π, then given that $\hat{C}$ is at most α incorrect, it follows that $|V_c^\pi - V_{\hat{c}}^\pi| \leq \alpha T V_c^\pi$ simply by the definition under Eqn. 2. Consequently, we have that

$$V_c^\pi - V_{\hat{c}}^\pi \leq \alpha T V_c^\pi \tag{17}$$

and

$$V_{\hat{c}}^\pi - V_c^\pi \leq \alpha T V_c^\pi \tag{18}$$

for any fixed policy π. We can apply Eqn. 18 to π^*, which gives us that $V_{\hat{c}}^{\pi^*} \leq (1+\alpha T) V_c^{\pi^*}$. Additionally, by the definition of optimality (as this is a minimization problem), we know that $V_{\hat{c}}^{\hat{\pi}^*} \leq V_{\hat{c}}^{\pi^*}$. Consequently, we get that $V_{\hat{c}}^{\hat{\pi}^*} \leq (1+\alpha T) V_c^{\pi^*}$. By applying $\hat{\pi}^*$ to Eqn. 17, we also have that $V_{\hat{c}}^{\hat{\pi}^*} \geq V_c^{\hat{\pi}^*} - \alpha T V_c^{\hat{\pi}^*}$. Finally, by connecting the inequalities, we get that $V_c^{\hat{\pi}^*} - \alpha T V_c^{\hat{\pi}^*} \leq (1+\alpha T) V_c^{\pi^*} \implies V_c^{\hat{\pi}^*} \leq \frac{1+\alpha T}{1-\alpha T} V_c^{\pi^*}$ which gives us the claim. □

VI. Empirical Evaluations

We test our approach in two different simulated domains as described below. In both domains, we begin by training the agent in one environment using human intervention, before testing them in three different environments within the domain where there is no human to intervene.

A. Domain Descriptions

1) Box-pushing: This domain is inspired by the Box-pushing domain presented in [2] where a robot is asked with pushing a box from a starting position to a goal position. The nominal reward is 0 in a goal state and negative elsewhere, encouraging the agent to reach the goal as efficiently as possible. However, there are rugs on the floor that the human does not want the robot to move over while pushing boxes. Consequently, when the human believes the robot is about to move onto a rug, they intervene and relocate the robot

Robot Box-Pushing									
		$\mathbf{E_1}$	$\mathbf{E_2}$	$\mathbf{E_5}$	$\mathbf{E_8}$	$\mathbf{E_{10}}$	**Test 1**	**Test 2**	**Test 3**
$\mathbf{H = 1}$	**I Acc.**	55.3	75.2	77.4	78.1	78.99	—	—	—
	C Acc.	78.4	88.1	83.7	82.2	83.5	85.6	86.5	82.7
$\mathbf{H \sim U(1, 4)}$	**I Acc.**	63.0	79.8	87.8	88.7	90.06	—	—	—
	C Acc.	92.01	95.7	96.4	96.7	97.2	97.8	97.6	97.1
$\mathbf{H \sim U(1, 5)}$	**I Acc.**	55.1	73.2	82.8	86.1	86.2	—	—	—
	C Acc.	94.2	96.1	95.4	95.4	97.07	97.9	98.0	97.4
$\mathbf{H \sim U(1, 6)}$	**I Acc.**	55.4	70.5	81.7	83.8	84.83	—	—	—
	C Acc.	89.2	97.1	95.1	96.5	98.0	98.5	98.0	98.3
$\mathbf{H \sim tN(3, 1.0)}$	**I Acc.**	51.3	73.5	81.3	84.1	84.35	—	—	—
	C Acc.	99.6	98.6	98.7	98.6	99.7	99.8	99.7	99.7
$\mathbf{H \sim tN(3, 0.25)}$	**I Acc.**	53.4	72.6	82.4	84.7	85.3	—	—	—
	C Acc.	99.5	99.1	100.0	99.5	99.2	100.0	99.6	99.5

Autonomous Vehicle Navigation									
		$\mathbf{E_1}$	$\mathbf{E_5}$	$\mathbf{E_{10}}$	$\mathbf{E_{15}}$	$\mathbf{E_{20}}$	**Test 1**	**Test 2**	**Test 3**
$\mathbf{H = 1}$	**I Acc.**	95.0	93.4	94.2	95.0	95.5	—	—	—
	C Acc.	55.0	63.0	63.0	62.0	63.0	65.0	64.0	63.0
$\mathbf{H \sim U(1, 4)}$	**I Acc.**	52.5	80.9	80.8	84.0	80.6	—	—	—
	C Acc.	61.0	75.0	77.0	76.0	76.0	75.0	76.0	76.0
$\mathbf{H \sim U(1, 5)}$	**I Acc.**	50.5	83.8	85.3	90.7	83.4	—	—	—
	C Acc.	60.0	78.0	77.0	81.0	78.0	79.0	79.0	79.0
$\mathbf{H \sim U(1, 6)}$	**I Acc.**	63.0	83.5	83.3	83.9	85.9	—	—	—
	C Acc.	61.0	77.0	77.0	80.0	81.0	81.0	81.0	81.0
$\mathbf{H \sim tN(3.5, 2.5)}$	**I Acc.**	60.0	82.6	84.8	85.5	81.5	—	—	—
	C Acc.	65.0	80.0	80.0	83.0	80.0	80.0	80.0	80.0
$\mathbf{H \sim tN(4, 0.5)}$	**I Acc.**	52.0	87.6	90.2	90.0	91.2	—	—	—
	C Acc.	66.0	79.0	81.0	82.0	82.0	83.0	82.0	82.0

TABLE I: I Acc. and C Acc. denote the accuracies of the intervention and constraint models respectively after each of 5 epochs of training, $\mathbf{E_i}$, in a fixed training environment, and in each of 3 test environments. $\mathbf{U}$ denotes a uniform distribution, and $\mathbf{tN}$ denotes a truncated normal distribution, truncated from 1 to 6.

to a location from which there is no chance of a constraint violation within their horizon. States $s \in S$ are represented by the tuple $\langle x_i, y_i, x_g, y_g, \phi \rangle$, where x_i and y_i denotes the current location of the robot, x_g and y_g denotes the robot's goal location, and ϕ denotes if there is a rug in the robot's current location. Actions $a \in A$ represent 8 directional movements. Each of the train and test environments differs in the number of rugs and their positions.

2) Autonomous Vehicle Navigation: In this domain, an autonomous vehicle (AV) is supervised by a human and must traverse a graph from a start node (intersection) to a goal node (intersection), modulating its speed as necessary. States $s \in S$ are represented by the tuple $\langle id, o, pr, s, x, \theta \rangle$ where id is the intersection ID, o denotes whether the intersection appears obstructed, pr denotes whether the intersection is protected for the AV, s denotes the AV's speed, x denotes the AV's position relative to the intersection, and θ denotes their current heading. Actions $a \in A$ are represented by a direction of travel and a speed modulation (acceleration, soft brake, hard brake, and nothing). The AV is initially only aware of hard legal constraints, such as stopping at an unprotected intersection with a stop sign, but not that it needs to slow down sufficiently early as it approaches an intersection (even if it is protected) if there is a potential obstruction, and cannot hard-brake right as it reaches an intersection for the comfort of the human. The human intervenes by braking when they expect the AV to not slow down, or slow down too late.

B. Results

Tables I and II show the results from our experiment where we compare the performance against a reactive agent that assumes that the human always intervenes due to the current state-action pair, and five different proactive agents, to illustrate how the agent's prior knowledge of the human's temporal model affects its learning ability. We present both the classification accuracy on interventions (i.e., predicting whether the human will intervene for a given (s, a)) and constraints (i.e., predicting whether a given (s, a) belongs to C) after five epochs of training in table I. In table II we present the average objective value for each objective – constraint minimization and domain reward maximization – over 10,000 trials for 4 environments: the environment where the agent was trained, and three different test environments. We tested a proactive agent with three uniform distributions and two truncated normal distributions, wherein all experiments the human had a *fixed* horizon of 3 and 4 for the box-pushing and autonomous vehicle domains respectively.

In table I, we see that the reactive agent in the autonomous vehicle domain achieves high intervention accuracy (although, notably not in the box-pushing domain), but is never able to break 90% and 70% in the two domains respectively in terms of accuracy on the constraint set in either the train or test environments. This illustrates that the agent is learning the wrong measure to apply to future environments where predicting an intervention does not directly predict a constraint. On the other hand, each proactive agent is able to achieve high ($\approx 85 - 90\%$) accuracy on the intervention set, and nearly 100% accuracy in the box-pushing domain, and at least 80% accuracy in the autonomous vehicle domain, for the constraint set.

In table II, we included the performance of an agent, referred to as `Ignorant`, which did not model the constraint set at all, as a baseline for the nominal objective. Notably, the reactive agent did not perform the best in either objective across all environments tested. In the box-pushing domain, there was not a significant difference in performance between the proactive agents, but all five outperformed the reactive agent in both objectives, indicating that simply accounting for proactivity is enough to properly learn the constraint set. In the autonomous vehicle domain, the proactive agent with distribution $\mathbf{U(1, 6)}$ performed the best over all uniform-distribution agents, and better than the normally distributed agent, $\mathbf{tN(3.5, 2.5)}$. This indicates that appropriate coverage over the horizon h, so that sufficient *weight* is placed on or around the human's true horizon, can be more important than the distribution itself when lacking a well-informed prior.

However, an agent with almost perfect knowledge of the human's horizon, using distributions $\mathbf{tN(3, 0.25)}$ and $\mathbf{tN(4, 0.5)}$ for each domain respectively, led to the best results overall. This is particularly notable in the autonomous vehicle domain where the agent incurred a 0.0, or nearly 0.0, sample likelihood of a constraint violation in the three test environments, outperforming the other proactive agents by at least an order of magnitude. These results strongly indicate that a well-informed prior on the human's temporal model can significantly improve the quality of the learned constraint set by the agent in domains where the constraints are sparse, which is particularly important in safety-critical domains.

Robot Box-Pushing				
	Train Eval	**Test Eval 1**	**Test Eval 2**	**Test Eval 3**
`Ignorant`	$[2.67, \mathbf{-2.09}]$	$[4.42, \mathbf{-2.07}]$	$[5.96, \mathbf{-2.05}]$	$[3.92, \mathbf{-2.05}]$
$\mathbf{H = 1}$	$[0.33, -2.90]$	$[0.62, -3.04]$	$[0.34, -3.15]$	$[0.17, -3.00]$
$\mathbf{H \sim U(1, 4)}$	$[0.13, -2.88]$	$[0.26, -2.92]$	$[0.05, -3.08]$	$[0.02, -2.93]$
$\mathbf{H \sim U(1, 5)}$	$[0.13, -2.84]$	$[0.26, -2.94]$	$[0.04, -3.08]$	$[0.02, -2.94]$
$\mathbf{H \sim U(1, 6)}$	$[0.12, -2.82]$	$[0.26, -2.93]$	$[0.03, -3.07]$	$[0.02, -2.93]$
$\mathbf{H \sim tN(3, 1.0)}$	$[0.10, -2.84]$	$[0.25, -2.92]$	$[0.02, -3.06]$	$[0.01, -2.91]$
$\mathbf{H \sim tN(3, 0.25)}$	$[\mathbf{0.09}, -2.84]$	$[\mathbf{0.24}, -2.92]$	$[\mathbf{0.01}, -3.05]$	$[\mathbf{0.01}, -2.91]$

Autonomous Vehicle Navigation				
	Train Eval	**Test Eval 1**	**Test Eval 2**	**Test Eval 3**
`Ignorant`	$[1.04, \mathbf{-14.07}]$	$[0.66, \mathbf{-9.43}]$	$[0.87, \mathbf{-10.37}]$	$[0.89, \mathbf{-17.13}]$
$\mathbf{H = 1}$	$[0.75, -15.19]$	$[0.68, -9.52]$	$[0.33, -12.83]$	$[0.76, -17.57]$
$\mathbf{H \sim U(1, 4)}$	$[0.39, -16.03]$	$[0.34, -10.31]$	$[0.17, -13.20]$	$[0.36, -18.71]$
$\mathbf{H \sim U(1, 5)}$	$[0.39, -15.77]$	$[0.34, -10.32]$	$[0.27, -12.56]$	$[0.38, -18.63]$
$\mathbf{H \sim U(1, 6)}$	$[0.07, -16.59]$	$[0.10, -10.83]$	$[0.07, -13.04]$	$[0.05, -19.21]$
$\mathbf{H \sim tN(3.5, 2.5)}$	$[0.35, -16.07]$	$[0.33, -10.29]$	$[0.17, -13.19]$	$[0.36, -18.58]$
$\mathbf{H \sim tN(4, 0.5)}$	$[\mathbf{0.02}, -16.62]$	$[\mathbf{0.00}, -10.96]$	$[\mathbf{0.00}, -12.92]$	$[\mathbf{.003}, -19.27]$

TABLE II: Each $[x, y]$ denotes the sample average constraint violations per randomized episode (x) and sample average nominal reward value per randomized episode (y), over $10,000$ simulations.

In figure 2, we plot the heat-map of constraints occurring in each of the four environments used in the box-pushing domain; note that the white squares indicate areas where the agent is constrained from moving on top of. Each heat-map is created by averaging the results given 24 different goal locations. The maps labeled *True* illustrate the ground truth constraint maps; we can see that the proactive agent, $\mathbf{tN(3, 0.25)}$, achieves nearly identical results; this is due to near-perfect constraint prediction as seen in the table I. On the other hand, we can see that the reactive agent hallucinated constraints in many unconstrained states. These are states where the human intervened in the agent's behavior, and the agent incorrectly inferred that these states were in the constraint set. This over-constraining leads to the goals being unreachable, wherein the agent breaks many *actual* constraints to reach the goal at all, resulting in poor performance.

Finally, we conducted an ablation study on the exploration bonus E (see Table III). Here we can see that including E led to a significant decrease in constraint violation sample likelihood in many of the scenarios, up to 100%.

VII. Discussion

A. Learning Setting

In this paper, we consider a train-then-deploy learning setting in which an agent has a fixed amount of time or resources to learn its objective, an unknown constraint set. During training, the agent has a human teacher who can provide a safety net for the agent before it is deployed in its operational environment without such a safety net, but where the constraints will continue to hold. We focus on learning only from sparse interventions, however the addition of other types of feedback such as demonstrations, agent-driven queries, or action guidance, only benefits our approach if available, and is a primary direction of future work. Another natural extension is the online learning setting where the agent's training occurs throughout its operation.

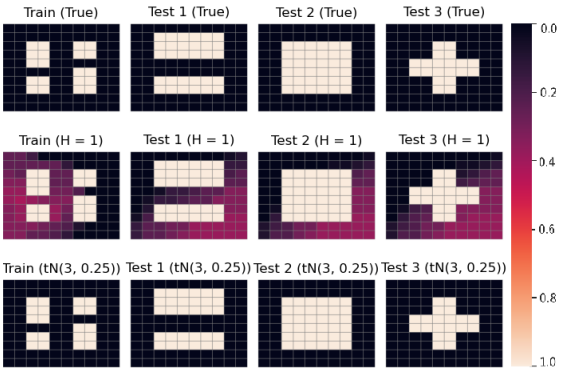

Fig. 2: Constraint inference in the Box-Pushing domain.

	Robot Box-Pushing				**Autonomous Vehicle**			
	Tr	**Te 1**	**Te 2**	**Te 3**	**Tr**	**Te 1**	**Te 2**	**Te 3**
$\mathbf{H = 1}$	42.1	39.1	50.1	57.5	-0.04	-0.01	0.00	0.01
$\mathbf{H \sim U(1, 4)}$	40.2	48.4	66.6	71.1	17.2	0.00	0.00	-0.06
$\mathbf{H \sim U(1, 5)}$	55.9	50.7	84.2	0.11	18.8	0.00	-0.04	0.00
$\mathbf{H \sim U(1, 6)}$	42.3	22.2	72.5	50.0	78.8	37.5	77.0	58.3
$\mathbf{H \sim tN(i)}$	28.9	13.7	71.1	66.6	16.7	0.00	32.6	-6.00
$\mathbf{H \sim tN(j)}$	55.0	29.5	91.5	66.6	93.2	100.0	100.0	94.2

TABLE III: Impact of adding exploration bonus E as *percent decrease* in sample likelihood of constraint violation, where i denotes $(3, 1)$ and $(3.5, 2.5)$, and j denotes $(3, 0.25)$ and $(4, 0.5)$, for the two domains respectively.

The challenge of such a setting is that, generally, human interventions are not free, as we have considered in this work; as a consequence, a non-fully cooperative game is induced as the agent's objective is now impacted by the likelihood of an intervention, impacting its own policy, which is known by the human and determines the human's likelihood of intervening.

B. Constraint Optimization

There are several ways of modeling constraint optimization problems; in this paper, we consider one that is naturally applicable to the types of open-world, safety-critical domains that motivate this work. That is, domains where we aim to minimize the expected number of constraint violations as the primary objective in a lexicographic optimization setting. However, our approach naturally extends to other settings. First, one may consider a *soft-constraint* setting, where constraints model non-critical elements like preferences or negative side effects [2]; this approach can be naturally formulated as an LMDP where the constraint violations represent the *secondary* objective. Second, one may consider a *budgeted* violation setting in which the number of constraint violations must simply be kept below a budget, B. Such an approach can be naturally formulated as a linear program, which is a known approach to compute an optimal policy for a Markov decision process [35].

VIII. Conclusion

In this paper, we consider the challenge of learning from proactive feedback, i.e., feedback generated by the human operator that is conditioned on their inferred near-term future behavior of the agent under an ϵ-noisy model of the agent's

policy. Our approach relies on minimal assumptions about the human's temporal model, yet still enables an agent to learn a high-quality model of the constraint set from proactively generated feedback. We prove that if the learned constraint model is at least $(1-\alpha)$ accurate, then the expected number of constraint violations under the optimal policy for the learned constraint model will be at most a factor of $\frac{1+\alpha T}{1-\alpha T}$ of the optimal expected number of constraint violations, when operating in a finite horizon setting with horizon $T \in \mathbb{N}$. We empirically validate our approach against a reactive learning model that assumes that all interventions are conditioned on the current state-action pair. We show that our approach achieves higher accuracy on constraint prediction and lower sample likelihood of a constraint violation across multiple environments in two simulated domains, without sacrificing quality in the nominal objective, and sometimes improving it. As a result, we suggest that in real world settings where the human's feedback may be proactive, it is critical for models that learn from such feedback to consider this proactivity to ensure that they learn the correct model. Furthermore, our results indicate that a high quality prior over the human's temporal model can significantly improve performance, leading to almost no constraint violations across all environments tested. Future work will examine how calibration tasks, such as those performed by Schrum et al. [23], may be used to produce high quality priors on the human's temporal model.

REFERENCES

[1] T. G. Dietterich, "Steps toward robust artificial intelligence," *AI Magazine*, vol. 38, no. 3, 2017.

[2] S. Saisubramanian, E. Kamar, and S. Zilberstein, "Avoiding negative side effects of autonomous systems in the open world," *Journal of Artificial Intelligence Research*, vol. 74, 2022.

[3] S. Ross and D. Bagnell, "Efficient reductions for imitation learning," in *International Conference on Artificial Intelligence and Statistics (AIStat)*, 2010.

[4] D. Wang, J. A. Russino, C. Basich, and S. Chien, "Analyzing the efficacy of flexible execution, replanning, and plan optimization for a planetary lander," in *International Conference on Automated Planning and Scheduling (ICAPS)*, 2020.

[5] R. Ramakrishnan, E. Kamar, B. Nushi, D. Dey, J. Shah, and E. Horvitz, "Overcoming blind spots in the real world: Leveraging complementary abilities for joint execution," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2019.

[6] S. Arora and P. Doshi, "A survey of inverse reinforcement learning: Challenges, methods and progress," *Artificial Intelligence*, vol. 297, 2021.

[7] S. Adams, T. Cody, and P. A. Beling, "A survey of inverse reinforcement learning," *Artificial Intelligence Review*, 2022.

[8] J. Spencer, S. Choudhury, M. Barnes, M. Schmittle, M. Chiang, P. Ramadge, and S. Srinivasa, "Expert intervention learning," *Autonomous Robots*, vol. 46, no. 1, 2022.

[9] A. Bajcsy, D. P. Losey, M. K. O'Malley, and A. D. Dragan, "Learning from physical human corrections, one feature at a time," in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2018.

[10] C. Basich, J. Svegliato, K. H. Wray, S. Witwicki, J. Biswas, and S. Zilberstein, "Learning to optimize autonomy in competence-aware systems," in *International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 2020.

[11] H. Liu, S. Nasiriany, L. Zhang, Z. Bao, and Y. Zhu, "Robot learning on the job: Human-in-the-loop autonomy and learning during deployment," *Computing Research Repository*, vol. abs/2211.08416, 2022.

[12] Q. Li, Z. Peng, and B. Zhou, "Efficient learning of safe driving policy via human-ai copilot optimization," in *International Conference on Learning Representations, (ICLR)*, 2022.

[13] T. S. Braver, "The variable nature of cognitive control: a dual mechanisms framework," *Trends in Cognitive Sciences*, vol. 16, no. 2, 2012.

[14] L. G. Appelbaum, C. N. Boehler, L. A. Davis, R. J. Won, and M. G. Woldorff, "The dynamics of proactive and reactive cognitive control processes in the human brain," *Journal of Cognitive Neuroscience*, vol. 26, no. 5, 2014.

[15] S. Chernova and M. Veloso, "Interactive policy learning through confidence-based autonomy," *Journal of Artificial Intelligence Research*, vol. 34, 2009.

[16] M. T. Rosenstein and A. G. Barto, "Supervised actor-critic reinforcement learning," in *Handbook of Learning and Approximate Dynamic Programming*. IEEE Press, 2004.

[17] D. Brown, W. Goo, P. Nagarajan, and S. Niekum, "Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations," in *International Conference on Machine Learning (ICML)*, 2019.

[18] W. B. Knox, P. Stone, and C. Breazeal, "Training a robot via human feedback: A case study," in *International Conference on Social Robotics (ICSR)*, 2013.

[19] S. Griffith, K. Subramanian, J. Scholz, C. L. Isbell, and A. L. Thomaz, "Policy shaping: Integrating human feedback with reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 26, 2013.

[20] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer, "HG-DAgger: Interactive imitation learning with human experts," in *IEEE/RSJ International Conference on Robotics and Automation (ICRA)*, 2019.

[21] C. Basich, J. Svegliato, A. Beach, K. H. Wray, S. Witwicki, and S. Zilberstein, "Improving competence via iterative state space refinement," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[22] Y. Cui and S. Niekum, "Active reward learning from critiques," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[23] M. L. Schrum, E. Hedlund-Botti, N. Moorman, and M. C. Gombolay, "MIND MELD: Personalized meta-learning for robot-centric imitation learning," in *Human Robot Interactions*, 2022.

[24] M. L. Schrum, E. Hedlund-Botti, and M. Gombolay, "Reciprocal MIND MELD: improving learning from demonstration via personalized, reciprocal teaching," in *Conference on Robot Learning (CoRL)*, 2022.

[25] G. Chou, D. Berenson, and N. Ozay, "Learning constraints from demonstrations with grid and parametric representations," *The International Journal of Robotics Research*, vol. 40, no. 10-11, 2021.

[26] D. R. R. Scobee and S. S. Sastry, "Maximum likelihood constraint inference for inverse reinforcement learning," in *International Conference on Learning Representations (ICLR)*, 2020.

[27] D. L. McPherson, K. C. Stocking, and S. S. Sastry, "Maximum likelihood constraint inference from stochastic demonstrations," in *Conference on Control Technology and Applications (CCTA)*, 2021.

[28] S. Malik, U. Anwar, A. Aghasi, and A. Ahmed, "Inverse constrained reinforcement learning," in *International Conference on Machine Learning (ICML)*, 2021.

[29] D. Papadimitriou, U. Anwar, and D. S. Brown, "Bayesian methods for constraint inference in reinforcement learning," *Transactions on Machine Learning Research*, 2022.

[30] C. Basich, J. Svegliato, K. H. Wray, S. Witwicki, J. Biswas, and S. Zilberstein, "Competence-aware systems," *Artificial Intelligence*, vol. 316, 2023.

[31] K. H. Wray, S. Zilberstein, and A.-I. Mouaddib, "Multi-objective MDPs with conditional lexicographic reward preferences," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

[32] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, 2013.

[33] J. Gawlikowski, C. R. N. Tassi, M. Ali, J. Lee, M. Humt, J. Feng, A. M. Kruspe, R. Triebel, P. Jung, R. Roscher, M. Shahzad, W. Yang, R. Bamler, and X. X. Zhu, "A survey of uncertainty in deep neural networks," *Computing Research Repository*, vol. abs/2107.03342, 2021.

[34] O. Sagi and L. Rokach, "Ensemble learning: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, 2018.

[35] A. S. Manne, "Linear programming and sequential decisions," *Management Science*, vol. 6, no. 3, 1960.