Norm Learning with Reward Models from Instructive and Evaluative Feedback

Eric Rosen¹, Eric Hsiung¹, Vivienne Bihe Chi¹ and Bertram F. Malle¹

Abstract—People are increasingly interacting with artificial agents in social settings, and as these agents become more sophisticated, people will have to teach them social norms. Two prominent teaching methods include instructing the learner how to act, and giving evaluative feedback on the learner’s actions. Our empirical findings indicate that people naturally adopt both methods when teaching norms to a simulated robot, and they use the methods selectively as a function of the robot’s perceived expertise and learning progress. In our algorithmic work, we conceptualized a set of context-specific norms as a reward function and integrate learning from the two teaching methods under a single likelihood-based algorithm, which estimates a reward function that induces policies maximally likely to satisfy the teacher’s intended norms. We compare robot learning under various teacher models and demonstrate that a robot responsive to both teaching methods can learn to reach its goal and minimize norm violations in a navigation task for a grid world. We improve the robot’s learning speed and performance by enabling teachers to give feedback at an abstract level (which rooms are acceptable to navigate) rather than at a low level (how to navigate any particular room).

I. INTRODUCTION AND BACKGROUND

Interactions between people and intelligent agents are becoming more prevalent in society. For example, mobile cobots are being deployed in hospitals to autonomously transport materials by traversing public areas where people may be present [29]. Human-robot interactions are bound to arise, and people will expect these intelligent agents to act in ways that conform to social norms. For example, in the hospital domain, some norms may prescribe to let another person pass in a narrow hallway and to greet the person, while other norms may prohibit entering certain rooms (like a doctor’s office), depending on the time of day.

In our work, we rely on a working definition of norms, compiled from several converging proposals [3], [5], [8], [21]: A norm is a directive, within a given community, to (not) perform an action in a specific context, provided a sufficient number of community members demand of each other to follow the directive and do in fact follow it.

An agent that can take context-specific norm-conforming actions must have an internal norm representation. Context-dependent reward models share several properties with the kinds of representations that are captured by the above definition. Reward models provide direction on how to act; these directions are context-specific; they are based on a compact, transferable representation of both context and action; and context-dependent policies can be derived that maximize some quantity analogous to norm conformity.

But representing norms is not enough. Robots could find themselves in a nearly infinite number of possible contexts—some familiar, some unfamiliar—so they must be able to learn new norms and update previously acquired norms. In this paper, we develop algorithms that both represent and learn norm-conforming reward models. The models are context-dependent, feature-based, represented and learned at an abstract level, and they generate context-dependent policies that conform to predefined norms. However, these algorithms do not learn from observation, as previous work has often focused on, but from human teachers. People tend to teach norms to others who enter a new community, who are uncertain about the local norms, or who violate norms, perhaps inadvertently. A robot that learns norms from people in a community must therefore be sensitive to how people actually tend to teach norms.

A. Two kinds of teaching

In the educational and developmental literature, two broad teaching strategies have been contrasted: (1) the teacher instructs the learner or (2) the teacher lets the learner explore and discover [10], guided by varying amounts of feedback [31]. Formal pedagogy (e.g., in schools) has generally favored teaching by instruction, but both methods are used, and often in mixed forms. Even children use both strategies to teach others, and they systematically vary them as a function of task difficulty, learner’s competence, and more [28], [6]. In the human-robot interaction literature, an analogous pair of teaching methods exists, in which robots are either directed to act a certain way (by prior programming, demonstration, or verbal commands [9], [25]) or are left to explore possible actions and given evaluative feedback that encourages or discourages those actions [30]. In the domain of norm learning, in particular, robots have been taught appropriate actions by instruction [9], [26] or exploration and feedback [19]. However, the teachers were researchers, so we do not yet know how ordinary people would approach the task of teaching norms to robots.

B. Overview

Our overall approach is to use behavioral research insights into how humans teach robots norms and harness these insights to then develop new learning algorithms that are sensitive to, and take advantage of, the natural teaching methods people bring to norm-learning robots. We previously tested empirically whether people would teach robots norms by using one or both of the two major teaching methods commonly used in HRI and human pedagogy: instruction

¹Affiliated with Brown University
and evaluative feedback [7]. We summarize the results of this experiment in more detail below. Our empirical results suggested that people use instruction and evaluative feedback in almost equal amounts but systematically select one or the other method as a function of the robot’s perceived abilities and its learning progress.

Given that humans appear to teach robots norms using both methods—instruction and evaluative feedback—and given that both teaching methods are so common in human pedagogy and human-robot interaction, it is critically important that algorithms learning norms should accommodate both modes of teaching equally well. We therefore developed several automated teacher models, some of which were inspired by our empirical human experiments data, and demonstrate that an agent running a likelihood-based learning algorithm capable of handling combinations of instruction and evaluative feedback can successfully learn context-dependent reward models. These reward-models induce norm-conforming policies, as evaluated by stochastic metrics of goal success and policy violations. Our contributions over previous work are in developing automated teacher policies based on realistic assumptions of human behavior, proposing a balanced method for handling different feedback types, and the use of abstractions for reducing the number of teacher interactions required for learning.

II. RELATED WORK

Prior work on norm learning for intelligent agents can be divided into two kinds of methods: learning from instruction (i.e., from nonverbal demonstration or verbal communication about how to act) and learning from evaluative feedback (i.e., from a signal indicating the appropriateness of performed actions). As examples of the first method, robots in [25] were able to learn cooking norms from written instructions in the form of abstracted probabilistic action-specific knowledge. In other work [9], a robot succeeded at one-shot learning of norms about what actions to take on an object, where norms and context were communicated through natural language. Others studied norm inference using inverse reinforcement learning (IRL), with norms explicitly represented by linear temporal logic (LTL) [15], and [26] used deep adversarial IRL to learn norm-conforming policies from demonstrations and a multi-objective reward function. Our framework also operates under the IRL framework, modeling the learning of norms as inferring rewards functions from variable feedback types from teachers, which enables our agents to leverage data-driven techniques from the reinforcement learning community. Here we review related work on algorithms that learn policies based on either action data or evaluative data, and discuss newer methods that leverage both.

In the reinforcement learning literature, several methods have been proposed to ultimately learn optimal policies, either from pre-collected datasets or via interactions and feedback from the environment or other agents. One class of algorithms falls under inverse reinforcement learning (IRL) [24], [1], where the agent infers a reward function from a dataset of pre-collected demonstrations such that the agent’s resulting policy can closely reproduce the trajectories in the original dataset. Learning these reward functions incrementally and interactively has also been explored: [2] introduced an incremental IRL framework, and [13] investigated agents interactively learning a reward function from two types of teachers—those with full observations of the agent’s current policy, and those with only access to noisy samples of the agent’s actions—each of which provide demonstration-only feedback.

A different learning approach, interactive RL, focuses on scalar feedback signals given by people, rather than the environment. In TAMER [16], agents use feedback from a human, who is assumed to provide scalar reward-based feedback that is time-delayed to learn a policy. In COACH [19], [20], agents interactively learn a policy from scalar policy-dependent human feedback, which assumes that teachers feedback is shaped based on how the teacher expects the agent to behave and how the agent actually performs. In [19], [20], expert researchers provide feedback to properly guide learning based on expert knowledge of the agent’s policy, whereas our work focuses on studying how non-expert humans teach robots norms.

Other researchers have considered how agents can learn from combinations of demonstration and scalar-based feedback [23]. However, a common strategy is to first teach with demonstrations and then merely fine-tune the algorithm with scalar evaluative feedback, as is done under both IRL-TAMER [18] and a method proposed by [22]. In both cases, the first phase of learning from demonstration is done in a supervised fashion. However, our preliminary work [7] suggests that this is not a realistic model of human teaching strategies, and that humans prefer to give varied feedback throughout the norm teaching process.

Our work differs from previous norm-learning methods in multiple respects. First, we model teaching behavior after how people actually teach agents [7], whereas the previous literature has relied on unrealistic assumptions about human teachers (e.g., Boltzman rationality [4], [12]). Second, we test an algorithm that combines in equal importance both information from instruction and information from evaluative feedback to interactively learn context-dependent reward models, hence norms. Third, going beyond our preliminary results using this combined approach [11], we take the critical step to introduce abstraction from individual states to sets of states that enable the agent to more efficiently learn from teacher feedback while still being able to plan tasks and abide by norms effectively. Indeed, the fundamental nature of “contexts” is that they are abstractions of specific situations, so we try to capture this property by abstracting away low-level state features across which the agent generalizes for planning and norm learning.

III. PRELIMINARIES

A. Human Teaching of Robots: Empirical Study

In an empirical study [7], we investigated people’s strategies when teaching a robot to behave appropriately in a healthcare setting—choosing to either instruct the robot how...
to act or to provide evaluative feedback on the robot’s proposed action. We designed a platform that allowed participants to engage in an interactive training session with a (simulated) robot agent through a smartphone-based interface resembling a messaging app. The robot was introduced to participants as “a competent nurse’s assistant robot who is ready to learn in situations where the best action is unclear.”

The main experiment consisted of 18 subtasks (teaching trials). For each subtask, the robot initiated the interaction by describing its current environment (e.g., standing in front of a patient’s room), the specific context (e.g., the door is cracked open), and its goal (e.g., enter politely). Following the robot’s description, participants were prompted to choose from one of the two teaching strategy options—instruct or evaluate.

When choosing to instruct, the participant selected the most norm-compliant action of three possible actions (prepared by the experimenter). When choosing to evaluate, the participant requested the robot’s planned action for the current subtask, and then the participant evaluated the proposed action on an ordinal 5-point scale. The robot’s action proposals were hard-coded in a way that demonstrated overall learning progress over the course of the training session.

Participants made a teaching strategy choice (instruct vs. evaluate) in each of the 18 teaching trials. Of the 206 participants recruited on Prolific (http://prolific.co), 89% used both strategies at various teaching phases. More importantly, they used them in selective ways.

Teaching by evaluation was favored when a teacher had gathered immediate or accumulated evidence about the robot’s learning progress and also for reoccurring tasks. Specifically, an appropriate action proposal by the robot in the most recent trial significantly increased people’s likelihood to evaluate the robot in the next trial, \( z = 3.85, p < .001 \). Further, we computed an accumulated performance score that captures the quality and amount of learning evidence the participant had observed in previous evaluation trials. A more evident learning progress indicated by a higher accumulated learning score predicted a significantly higher probability of evaluation, \( z = 8.4, p < .001 \). In addition, when the same subtask reoccurred later in training, probability of evaluation significantly increased from the initial instance to the reoccurring instance, \( z = 0.8, p = 0.001 \). It appears that people choose evaluation teaching when they have some evidence for the learner’s emerging competence.

By contrast, teaching by instruction was favored when people were just starting out in their teaching session and when they observed a surprising setback in learning progress. Specifically, the more a participant had accumulated evidence of previous learning success, the more they were sensitive to an inappropriate action proposal, and offered to instruct the robot in the immediately following trial, \( z = 5.5, p < .001 \).

The results suggest that human teachers naturally vary their teaching strategies both in direct response to the robot’s performance and their own accumulated impressions of the robot. Therefore, algorithms that allow social robots to learn human norms must be able to infer reward models from any combination of action- and scalar-based evaluative feedback.

### B. Technical Background

#### a) Markov Decision Processes.

A Markov Decision Process (MDP) is a model of an agent’s decision-making process in an environment that is represented by a tuple of \((\mathcal{S}, \mathcal{A}, T, \gamma, R)\), where \(\mathcal{S}\) is the set of states, \(\mathcal{A}\) is the set of actions, \(T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1] \) represents the transition dynamics of the environment, \(\gamma \in [0, 1] \) is a scalar discount factor, and \(R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow \mathbb{R} \) is a reward model representing reward received for acting in the environment. An agent in an MDP typically seeks to maximize its expected sum of discounted rewards received from the environment, known as expected return, by acting according to a policy \(\pi\). The value function \(V^\pi(s) = \mathbb{E}_{\pi} \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s \right] \) represents the expected return when starting in state \(s\) and acting under policy \(\pi\). Similarly, the state-action value function \(Q^\pi(s, a) = \mathbb{E}_{\pi} \left[ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a \right] \) is analogously defined to be the expected return starting in state \(s\), taking action \(a\), and acting forever according to policy \(\pi\). The optimal policy \(\pi^*\) is the policy that maximizes \(V^\pi(s)\) for all \(s \in \mathcal{S}\). When \(Q^\pi\) and \(V^\pi\) are known, the advantage function \(A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)\) measures the relative value of actions \(a \in \mathcal{A}\) that can be taken from state \(s\) when acting under policy \(\pi\). If \(\pi\) is an optimal policy, then taking the optimal action \(a^*\) from \(s\) has advantage 0.

#### b) Reward Models.

In many cases, the MDP may lack a known reward model \(R\), and can be written as \(MDP/R\). In this case, the reward model may be learned from a variety of signals in the environment. A common representation of learned reward models is to assume states are feature-based so that the reward model can be expressed as \(R(s, a, s') = R(s) = w^T \phi(s)\), where \(\phi : \mathcal{S} \rightarrow \mathbb{R}^m\) maps from state space to feature space, and \(w \in \mathbb{R}^m\) are learned feature weights, where \(m\) corresponds with the dimensions of feature space. Such a representation is convenient when computing expected return over trajectories, and it also has advantages of dimensionality reduction when attempting to learn reward over a reasonably large state space. However, a disadvantage of this representation is that aliasing may occur with distinct transition tuples \((s, a, s')\) that map to the same point in feature space.

#### c) Markov Chains.

In the course of evaluating how well a stochastic policy conforms to norms, it can be helpful to simulate a Markov Chain. A discrete-time Markov Chain consists of a sequence of random variables \(\{X_t\}_{t \in \mathbb{N}}\), an initial state distribution \(\mathcal{D}\) from which the initial state is sampled, \(X_0 \sim \mathcal{D}\), and a set of transition probabilities \(P(X_{t+1} = x_{t+1} | X_t = x_t)\) that satisfy the Markov property, in which the probability of transitioning to the next state \(x_t\) depends only on the current state \(x_t\). At every timestep, progression through the Markov Chain depends on the transition probabilities. When simulating a MDP as a Markov Chain, the transition probabilities can be computed from the transition dynamics \(T\) and the current stochastic policy \(\pi\).

#### d) Absorbing States.

When evaluating a stochastic policy from an MDP, it is helpful to set the goal state to...
be an absorbing state if modeling the MDP as a Markov Chain. A state $x_a$ is absorbing if $P(X_{t+1} = x_j | X_t = x_a) = \delta_{aj}$, where $\delta_{aj}$ is 1 if $a = j$ and 0 if $a \neq j$. By setting the goal states of a discrete MDP to be absorbing, a Markov Chain can be used to simulate the exact probability that an agent ends up in a goal state after a certain number of actions (i.e., the time horizon), based on the initial state distribution and the agent’s policy.

C. Feedback

In an MDP, the consequence of an agent’s action $a$ in state $s$ resulting in a transition to state $s'$ is a scalar-reward from $R$. However, when reward is not explicitly available (e.g.: an MDP/$R$), the agent may receive other forms of feedback as a consequence to its actions. In many cases, this feedback comes from another agent—often a human teacher who gives instructive or evaluative feedback. Since instruction feedback is policy-independent, the agent can store all received instruction feedback. Evaluative feedback is similar to reward, in that it is scalar. However, unlike reward, evaluative feedback is not necessarily fixed—the returned values for equivalent state-action pairs can change over time if the evaluative feedback is policy dependent, such as if evaluative feedback is based on policy-dependent advantage, $A^\pi(s,a)$.

IV. METHODS AND TECHNICAL APPROACH

We use the general approach of learning reward models from feedback instruction and evaluative feedback proposed by [11] and apply it to context-dependent norm-learning, while significantly enhancing the learned reward models, policies, and automated teachers. Importantly, we conceptually treat instruction feedback and demonstration feedback as equivalent, in that they both represent state-action sequences at varying levels of abstraction.

A. Norm-Learning from Feedback

Our objective is to teach an agent norms under a certain context by having the agent learn a context-dependent reward model that induces an appropriate policy, based on feedback received from an automated teacher. To do so, the agent models the environment as a discrete MDP and maintains a current estimate of a feature-based reward model $R(s) = w^T \phi(s)$, where the weights $w$ are learned by the agent based on the received feedback. Every time the agent receives feedback, the feedback can be used to update the weights $w$ and therefore the reward model. We initialize $w = 0$, because all policies are optimal under 0 reward.

a) Differentiable Planning.: In order for the agent to receive feedback, it must perform (or propose) an action in the environment. The agent selects an action by solving the MDP under the agent’s current reward model estimate. The agent performs dynamic programming with the softmax Bellman operator to a finite number of iterations to receive approximations of $V^\pi$ and $Q^\pi$, where $\pi$ is the softmax of the $Q$-values, as outlined in Algorithm 1 (Softmax was chosen for its smoothness properties compared to a greedy max). All outputs of the algorithm are differentiable with respect to $w$. Once $\pi$ is received, the agent picks the optimal action $a^* = \text{argmax}_a \pi(a|s) = \text{argmax}_a Q^\pi(s,a)$ to take in its current state $s$. This method of differentiable planning allows a computation graph to be constructed and utilized for updating the weights via gradient-based updates.

b) Learning Reward from Feedback.: In response to each action the agent takes, the teacher provides feedback, which can be either instruction-based or scalar-based evaluative feedback. Since instruction feedback is policy-independent, the agent can store all received instruction feedback into a feedback buffer $B$. Whenever instruction feedback is received by the agent, each $(s,a)$ in the sequence is added to the feedback buffer $B$ and the agent seeks to maximize the likelihood that its policy takes the preferred actions in $B$ using a gradient update $w \leftarrow w + \alpha \nabla_w \log L$ with learning rate $\alpha$, where $L(\pi; w) = \prod_s \pi(a|s; w)$, and $\nabla_w \log L = \sum_s \sum_a \nabla_w \log \pi(a|s; w)$, with the product and sums taken over all $(s,a) \in B$. In contrast, scalar-based policy-dependent evaluative feedback $f$ cannot be saved for reuse, because the received feedback value is only valid for the agent’s current policy, and it is used to update the probability of the action $a$ just taken by the agent from state $s$ according to a pseudoloss $L_p$ whose gradient with respect to $w$ is $\nabla_w L_p = \frac{\nabla_w \pi(a|s; w)}{\pi(a|s; w)} f$. Positive scalar values of $f$ serve to increase the likelihood of taking action $a$ in state $s$, whereas negative values of $f$ serve to decrease the likelihood. When feedback is received by the agent, it uses the corresponding gradient update to update $w$. In practice, for instruction feedback, the gradient is averaged over a batch, and multiple updates are applied.

B. Contexts and Automated Teaching

We model automated teachers that provide demonstration and evaluative feedback using different feedback strategies. Each teacher is provided with access to a copy of the MDP/$R$.

---

**Algorithm 1 Planning with Softmaxed Value Iteration**

**Input:** $R(s) = w^T \phi(s)$

**Parameters:** $\beta, \gamma, T(s'|a,s)$

**Output:** $Q^\pi(s,a), V^\pi(s), \pi(a|s)$

1: Initialize $V(s) = R(s)$.
2: while not converged or under max iterations do
3: $Q(s,a) = \sum_s T(s'|a,s) V(s')$
4: $\pi(a|s) = \sum_a e^{-\beta Q(s,a)}$
5: $V(s) = R(s) + \gamma \sum \pi(a|s) Q(s,a)$
6: end while
7: $\pi(a|s) = \sum_a e^{-\beta Q(s,a)}$
8: return $Q(s,a), V(s), \pi(a|s)$
the agent is acting in, the agent’s current policy $\pi$, and the state-action $(s,a)$ pair the agent has just acted on. Each teacher is also provided with the context under which the agent must be taught, in the form of a specific violation set $\mathcal{V} \subset \mathcal{S}$, which indirectly encodes the notion of prohibited actions (i.e., those actions that would transition into $\mathcal{V}$). Each automated teacher is also assigned a feedback strategy to act under: provide demonstration and evaluative feedback according to a probability distribution $\mathcal{P}$ that stays constant during training.

a) Demonstration Feedback.: Our automated teachers provide single-step-length demonstrations, representing the preferred action $a$ the teacher believes the agent should take from state $s$ under context $\mathcal{V}$. In simulation, the automated teachers pre-compute their preferred actions in every state under $\mathcal{V}$ according to the least path cost for reaching the goal state in the MDP/$R$. Passing through or occupying states $s \in \mathcal{V}$ incurs high cost compared to other states not in $\mathcal{V}$.

b) Evaluative Feedback.: In the case of evaluative feedback, teachers provide policy-dependent advantage-based feedback by evaluating the agent’s policy under the teacher’s internal ground truth reward model $R^*$, based on context $\mathcal{V}$ to arrive at $A^*(s,a)$, which is returned as $f$ to the agent.

C. Evaluation Metrics

In order to evaluate how well the agent learns norms from its interactions with teachers that use different feedback strategies, we measure stochastic goal success and stochastic policy violations. Intuitively, stochastic goal success measures whether the agent’s internal reward model estimate induces a policy allowing it to reach the goal state, whereas stochastic policy violations measure how likely the agent commits norm violations in context $\mathcal{V}$.

a) Stochastic Goal Success.: To compute stochastic goal success, an $H$-step walk of a Markov Chain is simulated starting from an initial uniform distribution over all non-goal states, and returning the total probability of being in an absorbing goal state after $H$ steps. The transition probabilities matrix $P^S$, with elements given by $P^S(s'|s) = \sum_a \pi(a|s)T(s'|s,a)$ for $s \notin \mathcal{S}_g$, is constructed from the agent’s current stochastic policy $\pi$ and the environmental transition matrix $T$. The set of goal states $\mathcal{S}_g \subset \mathcal{S}$ in the environment are set as absorbing, with $P^S(s'|s) = \delta_{s'}$ if $s \in \mathcal{S}_g$. Starting from an initial uniform state distribution represented by vector $b_0$ over all non-goal states $s \in \mathcal{S} - \mathcal{S}_g$, allows for capturing the probability $b_H(s)$ of ending up in a particular state $s \in \mathcal{S}$ after $H$-step rollouts given by $b_H = b_0(P^S)^H$.

Stochastic goal success is defined as $\sum_{s \in \mathcal{S}_g} b_H(s)$.

b) Stochastic Policy Violations.: For stochastic violations $\frac{1}{|\mathcal{S}|} \sum_{s} \pi(a|s)T(s'|s,a)B(s,a,s')$, the probability mass that is associated with violating actions in the policy $\pi$ is normalized over all states. We define the violation matrix elements $B(s,a,s')$ to be binary—either 0 if a transition from $s$ to $s'$ under action $a$ is non-violating (if $s' \notin \mathcal{V}$), or 1 if the transition is violating, where $s \in \mathcal{V}$. The metric can be interpreted as the expected probability that the agent takes a violating action. The hyperparameters and all code used in our simulation experiments are available at [https://github.com/hsbwnpcpodoet/ijcai].

V. RESULTS

We ran automated teaching experiments in GridWorlds of varying sizes (6x6 and 3x3), shown in Figure 1 under varying contexts $\mathcal{V}$ and at varying levels of state abstraction to see how well the agent learned from automated teacher feedback. Teacher feedback was modeled as one of seven mixtures of demonstration and scalar evaluation—from a teacher that uses 100% demonstration to a teacher that uses 100% evaluation, and mixtures of 90:10, 70:30, 50:50, 30:70, and 10:90 in between. Our evaluations for stochastic goal success and stochastic policy violations are reported as a function of steps taken, so as to illustrate how long it takes the agent to reach the goal and limit violations. We also provide representative examples of the learned reward models for each method displayed in Figure 2.

Our results show that an agent using this likelihood-based algorithm can learn context-dependent norms from varying combinations of demonstration and evaluative feedback. However, while norm violations decrease substantially under most conditions, goal success is achieved far better under abstraction (the 3x3 grids; second row of Figure 2). Pure evaluative feedback achieved high goal success even in the more fine-grained 6x6 grids, but at the expense of committing the highest number of violations among all the teaching mixtures. As the proportion of demonstration feedback increased, the violations decreased significantly, with pure demonstrations resulting in the fewest violations. However, pure demonstration feedback results in a period of low goal success immediately prior to high goal success, due to overfitting to the limited amount of optimal action data available early on in the training process. Moreover, in our human experiments we had found that very few participants used a pure demonstration teaching strategy, which would, in the real world, often be too time-consuming. The preferred strategy in the human experiment was a mixed instruction-evaluation one, and our teaching models with mixed strategies indeed showed the fewest violations and the fastest goal success (in the abstracted condition).

VI. DISCUSSION AND CONCLUSION

In this project, we used empirical evidence on how humans teach robots norms to inspire an algorithm capable of taking advantage of the two major teaching methods people
employ when teaching robots. The algorithm learned context-dependent reward models that induce context-dependent norm-conforming policies. We evaluated the learned policies with respect to stochastic goal success and stochastic policy violations, while varying the feedback strategies employed by automated teachers.

Our results indicate that agents using our algorithm can successfully learn context-dependent norms from varying combinations of demonstration and evaluative feedback; however, some amount of direct instruction is especially important for minimizing violations. Our results also suggest that interactively learning norms at an abstract level is beneficial for training time, which is important when considering the time and effort that a person would need to spend interactively teaching a robot. For example, in the 5x10 GridWorld presented by [11], many mixtures of demonstration and evaluative feedback were unable to reach 100% goal success within 10000 steps. Without abstraction to reduce the state space, premature over-fitting can result if too few demonstration are provided. This is detrimental to both the agent and the teacher, as the agent learns violation-avoiding behavior at the expense of goal success.

The central role of instruction in context-dependent norm learning concurs with human experimental findings in developmental psychology and our own recent empirical work, where teachers favor instruction when the learner is inexperienced, encounters more difficult tasks, or commits a surprising violation. Instruction is critical for norm teaching because it typically presents the optimal, norm-compliant action in the specific context. At the same time, it has the drawback that it often requires greater effort from the teacher [27], [17] and treats the learning agent as not fully capable. By contrast, evaluative feedback can, in real life, be done quickly and leaves the learner with more freedom, but it never explicitly conveys the pertinent norm. For example, our automated teacher’s evaluative feedback specifies how quickly and leaves the learner with more freedom, but it

much better the agent’s current action is than the average action the agent could take under its current policy and the teacher’s internal reward model. Using such policy-dependent advantage-based feedback $A^\pi(s, a)$ alone may not be sufficient to ensure the agent learns the desired norms: $A^\pi$ evaluates how well the agent’s current policy is at reaching the goal, but it does not account for any notion of violations in the world, except as represented by a teacher’s internal reward model.

A combination of instruction and evaluative feedback would therefore seem optimal, especially when the choice of teaching strategy is sensitive to the learner’s progress: At the beginning, and when the learner faces problems, instruction would be favored; over time, with increasing competence, evaluative feedback would be favored [14]. In our next project steps, we will equip automated teachers with more systematic, learner-sensitive choices of teaching method and, if successful, we will expose the algorithm to ordinary people who are told nothing more than to teach the robot how to act appropriately. This, after all, is the way people will likely encounter robots in future social worlds.

There are several limitations to our existing work that point towards open questions for future work to address. Regarding scalability, the abstract feature representations were hand-constructed in our experiments, so autonomously learning relevant features for a given context would enable robots to automatically improve their ability to learn social norms with fewer number of teacher interactions. Our framework was also only evaluated with simulated robots, and so experimental validation on robot hardware would be valuable for identifying novel issues in norm learning for robots (such as errors in perception, noisy control, etc.). Real-world teachers may also prefer to leverage a more diverse set of teaching strategies beyond instruction and evaluation, such as ranking behaviors as less or more appropriate. Lastly, our framework does not address the tension between generaliza-
tion and differentiation. For example, when encountering a new situation that has considerable similarity with a familiar context, should the robot generalize the previously learned norms to the new context or assume that it must differentiate the situation from previous ones and learn new norms? We view our existing work as providing a foundational learning framework for answering these questions in the future.

REFERENCES


