Language and Policy Learning from Human-delivered Feedback

Bei Peng[1], Robert Loftin[2], James MacGlashan[3], Michael L. Littman[3], Matthew E. Taylor[1], and David L. Roberts[2]

***Abstract*— Using rewards and punishments is a common and familiar paradigm for humans to train intelligent agents. Most existing learning algorithms in this paradigm follow a framework in which human feedback is treated as a numerical signal to be maximized by the agent. However, treating feedback as a numeric signal fails to capitalize on implied information the human trainer conveys with a lack of explicit feedback. For example, a trainer may withhold reward to signal to the agent a failure, or they may withhold punishment to signal that the agent is behaving correctly.**

We review our progress to date with *Strategy-aware Bayesian Learning*, which is able to learn from experience the ways trainers use feedback, and can exploit that knowledge to accelerate learning. Our work covers contextual bandits, goal-directed sequential decision-making tasks, and natural language command learning. We present a user study design to identify how users' feedback strategies are affected by properties of the environment and agent competency for natural language command learning in sequential decision making tasks, which will inform the development of more adaptive models of human feedback in the future.

I. INTRODUCTION

We address the development of techniques for teaching computers using intuitive and natural interactions inspired by dog training. Conveying a target behavior to a learning system is not new; efforts focusing on *imitation learning* are well documented in the literature [1], [2], [3], [4]. In this setting, a human trainer demonstrates a desired behavior to the learner, which copies it, either at the level of individual actions or in terms of its higher level intentions. In cases where the target behavior can be recognized but demonstrations are impossible, *reinforcement-learning* (RL) algorithms can be used [5], [6]—the learner seeks a behavior that maximizes a programmer-constructed reward function. Both approaches have been used effectively, but are limited.

We hypothesize that algorithms designed with inspiration from the ways humans train dogs will allow human trainers to convey target behaviors quickly and intuitively to computational learners. The novelty of our approach to learning in this human-delivered feedback paradigm lies in developing algorithms that decode the trainer's *intent* in their selection of training situations and types of feedback. The learning techniques discussed in this paper are initial steps towards algorithms that explicitly model and leverage the ways in which humans communicate with learners via a limited set of discrete feedback signals.

RL methods are most effective in environments where the numerically-valued reward function contains all the information needed to learn the policy. Learning from human feedback is substantively different. Trainers generally do not have numerical rewards to give to learners, using only a small set of discrete feedback signals, and they may give those signals in a number of different ways to implicitly communicate the target behavior. Thus, while standard RL algorithms can be used in this setting, they are not designed to leverage all of the available information.

> *Our overarching hypothesis is that models of the implied communication in training will enable machines to learn more, from fewer episodes.*

Below, we present a Bayesian inference learner that leverages trainer intent to significantly decrease the number of training episodes required to learn.

When learning from human trainers and the feedback they do (or do not) provide, implicatures can become significant. In dog training, for example, the trainer can reward the dog by giving a treat, or punish it with a stern "no." In these cases, the implicature of the reward and punishment are straightforward. However, there is a third feedback the trainer can use: a lack of reward or punishment. If the trainer neither rewards the dog nor says "no," it could mean "that is not what I want, try something else" or "okay, keep going and you will get rewarded eventually." If machine-learning algorithms are to be successful at learning from human-delivered feedback, they too must be able to decode the implicature of feedback from human trainers.

There are a number of ways training feedback can be provided. These so-called *operant conditioning paradigms* can be grouped into four categories [7]: positive reward (R+), negative reward (R-), positive punishment (P+), and negative punishment (P-). It is important to note that *rewards* (positive or negative) should always increase the frequency of the behavior they are associated with while *punishments* should decrease the frequency. In this context, *Positive* refers to adding a stimulus and *negative* refers to removing a stimulus. An example of R+ is giving a dog a treat (rewarding by adding a positive stimulus). An example of P- is withholding a treat (punishing by removing a positive stimulus). Often times R+ is paired with P- while R- is paired with P+. In contrast, RL algorithms consider inputs as numerical values; a reward of zero is treated exactly as any other reward value, and may increase or decrease the frequency of an action depending on the current expected reward for that action.

This work was supported in part by NSF IIS-1319412.

[1] School of Electrical Engineering and Computer Science, Pullman, WA; {bpeng,taylorm}@eecs.wsu.edu

[2] Department of Computer Science, North Carolina State University; {rtloftin,robertsd}@csc.ncsu.edu

[3] Brown University Department of Computer Science; {james_macglashan,michael_littman}@brown.edu

In prior work, we found strong evidence that people (possibly with only a tacit understanding of their approach) use these different operant paradigms when training a learning agent [8], [9]. As in that work, this work focuses on one aspect of feedback, that is, the use of implicit feedback. While there are cases where the interpretation of feedback may be more complex, such interpretations may be highly user or context dependent, and we do not consider them here.

In this paper, we will present a model for learning from human-delivered feedback that can be applied in both a bandit domain and sequential decision-making tasks with language learning. We will describe human-subject results for the bandit domain and simulation results for sequential tasks. Then, we will discuss the result of a human-subjects experiment to validate the language-learning setting and determine properties that affect human feedback strategies.

II. RELATED WORK

Our work is part of a growing literature on learning from human feedback. Thomaz and Breazeal [10] treat human feedback as a form of guidance for an agent trying an RL problem. There, human feedback did not change the optimal policy for the RL problem, but improved exploration and accelerated learning. Their results show humans give reward in anticipation of good actions, instead of rewarding or punishing the agent's recent actions. Work by Knox et al. [11] examined how users want to provide feedback, finding that: 1) there is little difference in a trainer's feedback whether they think that the agent can learn or that they are critiquing a fixed performance; and 2) humans can reduce the amount of feedback they give over time, and having the learner make mistakes can encourage more feedback.

COBOT [12] was an online chat agent with the ability to learn from human agents using RL techniques. It learned how to promote and make useful discussion in a chat room, combining explicit and implicit feedback from multiple human users. The TAMER algorithm [13] has been shown to be effective for learning from human feedback in a number of task domains common in RL research. This algorithm is modeled after standard RL methods that learn a value function from human-delivered numerical rewards.

Of existing work, [14] is perhaps the most similar to ours. In that work and in ours, trainer feedback was interpreted as a discrete communication that depended probabilistically on the trainer's target policy, rather than on some numeric reward. Both our work and theirs use a model of the feedback distribution to estimate the trainer's policy.

In addition to the work on learning from feedback, there is a growing body of work on how humans can teach agents by providing demonstrations of a sequential decision task [15], or by selecting a sequence of data in a classification task [16].

A. Contextual Bandit with Human Feedback

In previous work [8], [9], we demonstrated that algorithms that account for the different strategies employed by users to train virtual agents can outperform algorithms that ignored trainer strategy. We have developed two algorithms, SABL and I-SABL, which are able to infer the behavior desired by the trainer based on the feedback given, using a probabilistic model of how trainers provide feedback.

Our probabilistic model assumes that the trainer first determines if the action taken was consistent with their target policy λ^* (a mapping from observations to actions), with some probability of error ε. The trainer then decides whether to give explicit feedback or simply do nothing. If the trainer interprets the action as correct, then she will give an explicit reward with probability $1-\mu^+$, and if she interprets the action as incorrect, will give explicit punishment with probability $1-\mu^-$. So, if the learner takes a correct action, it will receive explicit reward with probability $(1-\varepsilon)(1-\mu^+)$, explicit punishment with probability $\varepsilon(1-\mu^-)$, and no feedback with probability $(1-\varepsilon)\mu^+ + \varepsilon\mu^-$.

The parameters μ^+ and μ^- encode the trainer's strategy, which we assume to be the same for all states, and constant throughout a learning session (though we consider the possibility that the latter assumption is not always true). The values of these parameters can be specified as part of the algorithm, or can be learned online for a given trainer. Using the $R+$ and $P+$ notation from behaviorism literature, we see that $\mu^+ = 0.1, \mu^- = 0.1$ correspond to an R+/P+ strategy where nearly every action receives explicit feedback, while $\mu^+ = 0.1, \mu^- = 0.9$ correspond to an R+/P- strategy. What is important to note about this model is that, depending on the strategy used, the lack of feedback may be more probable for correct actions than incorrect actions, or *vice versa*. Therefore, the correct inference to make from a lack of feedback depends on the training strategy being used.

Using this model of feedback, SABL computes the maximum likelihood estimate of the trainer's target policy λ^* given the feedback that the user has provided. SABL requires prior knowledge of the trainer's strategy. If, however, the agent knows the correct action for some observations, it can infer the strategy by looking at the history of feedback for those observations. We can treat the unknown μ values representing the strategy as hidden parameters, and marginalize over possible strategies to compute the likelihood of a policy λ. Inferring-SABL, or I-SABL, finds a maximum likelihood estimate of the target policy, given the training data, that is

$$\operatorname*{argmax}_{\lambda} \sum_{s \in S} p(h_{1\ldots t}|s, \lambda^* = \lambda) p(s|\lambda^* = \lambda),$$

where S is the set of possible training strategies.

B. Goal-directed Sequential Learning

In the contextual bandit setting, SABL and I-SABL directly learned the policy to follow for each state from human feedback. In a large sequential domain, however, we believe trainers are often more interested in communicating environment-independent task goals for the agent to complete. For example, a dog trainer may teach a dog a newspaper retrieval task in which the goal is for the dog to get the newspaper and take it back to the trainer. Once a such a goal is learned, the dog will be able to carry it out regardless of where the trainer is in a home

environment. Note that this paradigm is analogous to *inverse reinforcement learning* [17] where the goal is to derive a policy from demonstrations indirectly by learning a reward function that reproduces the observed demonstrations. In both cases, learning a goal rather than a policy allows the agent to act correctly even in states where no feedback (implicit or explicit) has been given. The agent can simply take the action that is optimal for the known goal.

In previous work [8] we adapted SABL and I-SABL to this goal-directed setting by assuming that goals are represented by MDP reward functions and that the agent has access to an MDP planning algorithm that computes the optimal policy π^g for any goal-based reward function $g \in G$. Then, SABL's typical formulation of a "correct action" is redefined to be an action that is consistent with the optimal policy of the true goal being trained: $a \in \pi^{g^*}(s)$, where a is the action taken by the agent and $\pi^{g^*}(s)$ is the set of optimal actions in MDP state s for the actual goal $g^* \in G$. An "incorrect action" is an action that is inconsistent with the optimal policy of the true goal being trained: $a \notin \pi^{g^*}(s)$.

C. Language Learning with Reward and Punishment

Our goal is to enable people to naturally and effectively train an artificial agent to carry out a variety of different tasks with reward and punishment. One way to accomplish this goal would be for a person to manually specify each new task they are training and when they wanted an agent to perform a previously trained task, provide an interface that allowed them to select from the previously trained tasks. However, a more natural interface would be to connect the task learning with a natural language model. In this setting, a trainer could give a novel command and reward and punish the agent until the agent successfully completed the task. As the trainer taught additional tasks, the agent would become better at interpreting the language, thereby enabling the agent to successfully interpret and carry out novel commands without any reward and punishment. For example, an agent might learn the interpretation of "red" and "chair" from the command "move the red chair," and the interpretation of "blue" and "bag" from the command "bring me the blue bag," thereby allowing correct interpretation of the novel command "bring me the red bag." To enable language learning from agents trained with reward and punishment, we developed a probabilistic model [18] that connected the IBM Model 2 (IBM2) language model [19] with a factored generative model of tasks, and goal-directed SABL for learning from human feedback.

Using this probabilistic model, an iterative training regime proceeds as follows. First, the trainer gives an English command. From this command, a distribution over the possible tasks for the current state of the environment is inferred using Bayesian inference. This task distribution is used as a prior for the goals in goal-directed SABL. The agent is then trained with SABL for a series of time steps. After completing training, a new posterior distribution over tasks is induced and used to update the IBM2 model via weakly-supervised learning. The process repeats with a new command.

We previously tested this model using one of the authors as a trainer in a simplified home environment MDP [18]. Specifically, the environment was a series of colored rooms connected by doors with different kinds of colored objects (such as chairs and bags). The agent was trained on seven commands including "take the blue chair to the yellow room" and "take the purple bag to the blue room." As training proceeded, the agent became better at correctly interpreting commands and required less feedback to disambiguate the intended meaning. After training, the agent was able to correctly interpret novel commands that described different combinations of objects, object colors, and room colors. For example, the agent correctly interpreted the command "take the purple chair to the green room" despite never being trained with commands involving a purple chair or taking an object to a green room.

III. Toward sequential and language learning with human feedback

The previous section summarized our previous work, the most relevant results being:

- An agent can learn from human feedback in a contextual bandit setting.
- Three different sets of human populations successfully trained in the contextual bandit setting (college students, dog trainers, and Turkers).
- Human feedback can be treated as categorical, rather than numerical, in nature.
- One of the authors successfully provided human feedback to learn 1) a reward function and 2) a language model to act in a sequential task.

This section presents our current research directions and experimental setup.

A. Motivation

Our goal is to show that non-expert humans (i.e., workers on Amazon's Mechanical Turk, also known as "Turkers") can provide categorical feedback to an agent so that it can learn a policy appropriate for a given command. In these experiments, we fix the commands, limiting the human to providing rewards and punishment feedback. Future work will allow users to provide feedback and select their own wording to command goals.

In addition to the question "Can Turkers successfully train an agent?" we will also investigate three additional questions. First, we would like to better understand how users want to train agents in sequential domains. Are users more likely to use R−/P+? Does this change over time (e.g., will the user tire of providing rewards and be more likely to use R−/P−)? Will the user change their strategy if the agent is more (or less) successful initially?

Second, we are interested in how the properties of the sequential domain affect the user's strategy. The two dimensions we consider are *step size* and *step interval*, where each dimension has two settings. The agent could take large or small steps, and it could take fast or slow steps. An agent taking large steps at a fast interval would be able to finish

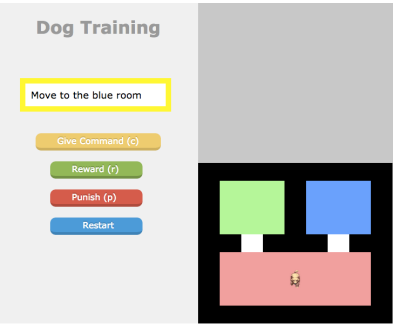


Fig. 1. The GUI used to train the agent with given commands.

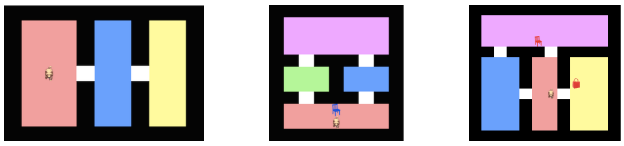
the task the fastest (in wall clock time). An agent taking small steps at a fast interval could finish the task in the same amount of time as an agent taking large steps at a slow interval. The fourth combination, an agent taking small steps at a slow interval, would take the longest amount of wall clock time. Our hypotheses are that users will 1) provide more feedback per action when the agent is slower and 2) be more likely to punish when the agent is slower.

Third, there may be a difference between when people are "actively" training the agent and "passively" testing its performance. Our hypothesis is that the amount of human feedback will decrease over time [20], and that there will be an even larger decrease in the amount of feedback once the user believes the agent has learned the task. We will test this by seeing how the amount of feedback changes over time, the number of times the agent has performed the task, and the correctness of the agent's policy. Depending on these results, we may need to implement an explicit testing phase into the experiment where the user observes the agent acting without being able to provide feedback. If the user accepts the agent's behavior, she can continue to the next task. If not, the user can return to the training phase and provide additional feedback to the agent.

B. Experimental Setup

To study how humans want to train the agent in sequential domains, we have developed a user study in which participants train a virtual agent to accomplish pre-specified commands by giving reward and/or punishment. Our domain is a simplified simulated home environment. The domain and user study GUI are shown in Figure 1. The domain consists of four object classes: agent, room, block, and door. The visual representation of the agent is a virtual dog. It can deterministically move one unit north, south, east, or west and move blocks by moving into them. The blocks can be chairs or bags; rooms and blocks can be red, yellow, green, blue, and purple. Doors (shown in white) connect two rooms so that the agent can move from one room to another one. The possible commands given to the agent include moving to a specified colored room (e.g., "move to the blue room") and taking a block with specified shape and color to a colored room (e.g., "move the blue chair to the purple room").

In our user study, users need to pass a color blind test before starting the experiment since the training task requires



(a) "Move to the yellow room"

(b) "Move the blue chair to the purple room"

(c) "Move the red bag to the blue room"

Fig. 2. The three training environments and their corresponding commands.

the ability to identify different colored objects. After passing the color blind test, users fill out a background survey indicating their age, gender, education, history with dog ownership, dog training experiences, and with which dog training techniques they are familiar. Once completing this initial survey, users are taken through a tutorial that explains how to interactively reward and punish the virtual dog based on its behavior. The user is told that punishment can be treated as a signal that the dog should consider a different task than the one it was executing. After the tutorial, but before beginning a series of training sessions, users are tested in the same environment as the tutorial to verify that they understood the interface.

Following the tutorial and verification test, users are requested to train the dog in a series of three environments shown in Figure 2. Each environment has a different level of complexity and are presented to users in a random order. The step size and step interval of the dog for all three environments is randomly selected from one of the four conditions discussed previously (small-slow, small-fast, large-slow, and large-fast).

Following training in the three environments, the users are asked to repeat training with a new dog in the same three environments. However, in this second sequence users are assigned to a different random step size and step interval condition. Upon finishing the second sequence of tasks, users are asked to describe the strategy used when training the agent. Finally, users are asked to provide any additional comments about the experiment that they have.

IV. RESULT ANALYSIS

The user study was published on Amazon Mechanical Turk as a set of Human Intelligence Tasks. We consider data from the 30 unique workers who passed successfully trained the task after the tutorial, verifying that they understood the interface. Each participant trained a dog in two sequences of the three environments. The two sequences were assigned the same three environments in random order, with two different training conditions. Each training condition was randomly picked from one of the four conditions (small-slow, small-fast, large-slow, and large-fast) with different combinations of step size and step interval of the agent. There were the same number of tasks for each of the four training conditions.

A. Can Turkers successfully train an agent?

The results show that the agent could learn the policy appropriate for the given command in 90% of the tasks. All 30 non-expert workers could successfully train the agent

TABLE I

TRAINING ACCURACY FOR EACH GIVEN COMMAND

#	Command Given	Accuracy
1	Move to the yellow room	98.3%
2	Move the blue chair to the purple room	81.7%
3	Move the red bag to the blue room	91.7%

TABLE II

TRAINING ACCURACY FOR EACH TRAINING CONDITION

#	Training Condition	Accuracy
1	small-fast	77.8%
2	large-fast	91.1%
3	small-slow	91.1%
4	large-slow	93.3%

TABLE III

BREAKDOWN OF STRATEGIES USED IN DIFFERENT TASKS

Command #	R+/P+	R+/P-	R-/P+	R-/P-	Total #
1	1	0	27	57	85
2	2	7	7	104	120
3	2	4	8	88	102

TABLE IV

BREAKDOWN OF STRATEGIES USED IN DIFFERENT SETTINGS

Condition	R+/P+	R+/P-	R-/P+	R-/P-	Total #
small-fast	2	7	16	87	112
large-fast	0	1	3	100	104
small-slow	3	2	16	43	64
large-slow	0	1	11	60	72

to accomplish more than half of the tasks. 66.7% of users were able to teach the agent to successfully execute all 6 commands. The percentage of participants who could train the dog to learn more than 5 commands was 80%.

Table I shows the performance of Turkers in training the agent to execute different commands. As we expected, command 1 achieved the highest accuracy since it was easiest, as there are no objects in the environment — the agent only needed to learn the words corresponding to room colors. The second task actually turned out to be harder than the third one, even though there was only one block (a blue chair) to move in the second environment but two blocks (a red bag and a red chair) in task 3. A one-way ANOVA test shows that the accuracy differences between these three commands were statistically significant ($p < 0.01$), demonstrating that the task complexity did affect the performance of participants in training the agent.

Table II shows the effects of training conditions on the user performance. A one-way ANOVA test shows that the training accuracy was not significantly different ($p = 0.08$) between the four different training conditions. However, what is interesting to note is that the small-fast setting was the hardest for the user while other three training conditions were very similar in performance. Based on our observation that users tended to give more explicit feedback per action in this setting, we suggest that this difference reflects difficulty on the part of the agent in responding to explicit feedback (especially negative feedback) quickly while taking small steps with a fast interval. More frequent successive feedback could confuse the agent while it was planning, making it harder for the agent to learn the policy. For instance, if two successive punishments were given to the agent's current action, the agent would plan a different action based on the first punishment, but receive the second punishment before executing it, which might prevent it from taking the new action and continue executing the current action.

B. Training Strategies

To better understand how users want to train agents in this sequential domain, Table III summarizes the distribution of strategies used for different training commands. To categorize the training strategies, we determined if each executed action of the agent was consistent with one of the optimal policies or not, then we checked if the user gave positive, negative or implicit feedback when the agent was taking the correct or incorrect action (e.g., consistant or inconsistent with an optimal policy). If correct actions received explicit positive feedback more than half of the time, it would be classified as R+. If incorrect actions received explicit negative feedback more than half of the time, it would be classified as P+. Under an R+/P+ strategy, correct actions typically received an explicit reward and incorrect actions typically got explicit punishment. An R-/P- strategy rarely gave explicit feedback of any type. Unlike our previous dog training domain where people were more likely to choose R+/P+ strategies [9], the dominant strategies here were R-/P-. R-/P+ strategies were still common, and occurred much more frequently than R+/P- or R+/P+ strategies. This is not surprising since the dog often behaved correctly without receiving any explicit feedback in this sequential domain. The agent's frequent correct behaviors motivated the user to ignore good behaviors. In contrast, if the agent acted incorrectly, users were more likely to provide punishments to correct the incorrect behavior. Another observation is that more users used R-/P+ strategies in command 1 compared to other two commands. We believe that this is due to the very low frequency of incorrect behaviors in the easier task, which biased the users towards providing more negative feedback to incorrect actions. Fisher's exact test shows that the number of times each of the four strategies was used was significantly different ($p \ll 0.01$) between different training commands. The training times for each command also show the level of complexity of each task (command 1 was easiest while command 2 was hardest).

We also explore whether the properties of the domain affect users' choice of training strategies. The breakdown of strategies used in different training conditions are shown in Table IV. Fisher's exact test shows that the difference in the frequency of the four strategies used in the different training conditions was statistically significant ($p \ll 0.01$). We hypothesize that users were more likely to select R-/P+ strategies so as to provide more punishment in the slow interval case, as the user might lose patience if the agent was not able to respond to feedback quickly enough. However, it is interesting that punishments were used about as frequently in the slow case as in the small-fast case. We also notice

that workers tended to do much less training (37% less) when the agent moved slower. One possible explanation was that the slow interval lead to more user fatigue, as the agent could take longer to learn the command. However, there is also some evidence that the slow interval did not affect the workers' performance since similar training accuracies were achieved in both the fast and slow cases.

We observed that the users still provided some rewards or punishments during the training while using the most popular strategy R-/P-. Therefore, it is interesting to explore under what conditions were they more likely to give explicit feedback. Future work would study whether the user tended to give more rewards than punishments (or *vice versa*) if the agent was in start state (started a new policy), end state (terminal state or goal state) or a door way.

C. Does the strategy change over time?

We are interested if users' training strategies changed over time, and under what conditions were they more likely to change. One possible factor was the time that the user was allowed to train the agent. The trainer would be more likely to change the strategy if she was allowed to train the dog as long as she liked. However, each Turker was allowed only 30 minutes to accomplish the task. This time was sufficient: the average time for Turkers to finish the HIT was 12 minutes.

It is possible that the agent's initial success could lead the user to "passively" test rather than "actively" train it. A change might also occur if the user grew tired of providing explicit feedback. The results show that 26 users changed their strategies (within the same task or between different tasks) at least once during the whole training process. Based on our first hypothesis about strategy change, we only consider tasks that were trained multiple times, where testing might take place. Out of 180 tasks, 54 were trained more than once by 20 users. 13 users changed their training strategies in 21 tasks. 71% of strategy changes were from R+/P- or R-/P+ to R-/P-, which matches our expectation that the user would tire of providing rewards or punishments and be more likely to use R-/P-. We also found that out of all the cases where the strategy switched from R-/P+ to R-/P-, 90% of time the agent was successfully trained to perform the command using R-/P+, and then the user changed the strategy to R-/P- to "passively" test agent's performance. This supports our hypothesis that the user tends to change the strategy if the agent is more successful initially.

Considering all 54 tasks that were trained multiple times by users, the number of explicit feedback given by the users decreased over time in 70.4% of the tasks. Our hypothesis was that there could be a decrease in feedback if the user believed the agent had learned the task. When evaluating whether the decreases in the number of explicit feedback were affected by the initial success of the agent, we found that 84% of time the agent reached the goal state in one training session and then the users reduced the number of feedback in next training session. It The number of human feedback decreased the most in second task, which was verified to be the hardest to train in three tasks.

V. LONG-TERM GOALS AND CONCLUSION

We believe our human subjects will find it natural to train our simulated agent to carry out commands. Designing environments so that the learner can make sensible pragmatic implicatures, however, is likely to be more challenging for people. Future work will attempt to improve the design of agents to better learn from human feedback. Our long-term goal is to build trainable agents that can learn from human intentions to create their own powerful representations. Successful representations not only speed up the learning of sophisticated behaviors, but also form the basis of ever richer representations.

REFERENCES

[1] B. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009.

[2] M. Babes, V. N. Marivate, M. L. Littman, and K. Subramanian, "Apprenticeship learning about multiple intentions," in *Proc. of ICML*, 2011.

[3] J. MacGlashan, M. Babes-Vroman, K. Winner, R. Gao, M. desJardins, M. Littman, and S. Muresan, "Learning to interpret natural language instructions," in *Proceedings of AAAI Workshop on Grounding Language for Physical Systems*, 2012.

[4] T. J. Walsh, K. Subramanian, M. L. Littman, and C. Diuk, "Generalizing apprenticeship learning across hypothesis classes," in *Proc. of ICML*, 2010.

[5] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[6] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 1998.

[7] B. F. Skinner, *Science and Human Behavior*. Macmillan, 1953.

[8] R. Loftin, J. MacGlashan, B. Peng, M. E. Taylor, M. L. Littman, J. Huang, and D. L. Roberts, "A Strategy-Aware Technique for Learning Behaviors from Discrete Human Feedback," in *Proc. of AAAI*, 2014.

[9] R. Loftin, B. Peng, J. MacGlashan, M. L. Littman, M. E. Taylor, J. Huang, and D. L. Roberts, "Learning Something from Nothing: Leveraging Implicit Human Feedback Strategies," in *Proc. of RO-MAN*, 2014.

[10] A. L. Thomaz and C. Breazeal, "Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance," in *Proc. of AAAI*, 2006.

[11] W. B. Knox, B. D. Glass, B. C. Love, W. T. Maddox, and P. Stone, "How humans teach agents - a new experimental perspective," *I. J. Social Robotics*, vol. 4, no. 4, pp. 409–421, 2012.

[12] J. Isbell, C.L., C. Shelton, M. Kearns, S. Singh, and P. Stone, "A social reinforcement learning agent," in *Proc. of Agents*, 2001.

[13] "Interactively shaping agents via human reinforcement: The TAMER framework," in *Proc. of K-CAP*.

[14] S. Griffith, K. Subramanian, J. Scholz, C. Isbell, and A. L. Thomaz, "Policy shaping: Integrating human feedback with reinforcement learning," in *Proc. of NIPS*, 2013.

[15] M. Cakmak and M. Lopes, "Algorithmic and human teaching of sequential decision tasks," in *Proc. of AAAI*, 2012.

[16] F. Khan, X. J. Zhu, and B. Mutlu, "How do humans teach: On curriculum learning and teaching dimension," in *Proc. of NIPS*, 2011.

[17] A. Y. Ng and S. Russell, "Algorithms for inverse reinforcement learning," in *Proc. of ICML*, 2000.

[18] J. MacGlashan, M. Littman, R. Loftin, B. Peng, D. Roberts, and M. E. Taylor, "Training an agent to ground commands with reward and punishment," in *Proceedings of the AAAI Machine Learning for Interactive Systems Workshop*, 2014.

[19] P. F. Brown, J. Cocke, S. A. D. Pietra, V. J. D. Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin, "A statistical approach to machine translation," *Computational Linguist*, vol. 16, pp. 79–85, June 1990.

[20] G. Li, H. Hung, S. Whiteson, and W. B. Knox, "Using informative behavior to increase engagement in the TAMER framework," in *Proc. of AAMAS*, 2013.