A Strategy-Aware Technique for Learning Behaviors from Discrete Human Feedback

Robert Loftin
North Carolina State University
rtloftin@ncsu.edu

James MacGlashan
Brown University
james_macglashan@brown.edu

Bei Peng
Washington State University
bei.peng@wsu.edu

Matthew E. Taylor
Washington State University
taylorm@eecs.wsu.edu

Michael L. Littman
Brown University
mlittman@cs.brown.edu

Jeff Huang
Brown University
jeff@cs.brown.edu

David L. Roberts
North Carolina State University
robertsd@csc.ncsu.edu

Abstract

This paper introduces two novel algorithms for learning behaviors from human-provided rewards. The primary novelty of these algorithms is that instead of treating the feedback as a numeric reward signal, they interpret feedback as a form of discrete communication that depends on both the behavior the trainer is trying to teach and the teaching strategy used by the trainer. For example, some human trainers use a lack of feedback to indicate whether actions are correct or incorrect, and interpreting this lack of feedback accurately can significantly improve learning speed. Results from user studies show that humans use a variety of training strategies in practice and both algorithms can learn a contextual bandit task faster than algorithms that treat the feedback as numeric. Simulated trainers are also employed to evaluate the algorithms in both contextual bandit and sequential decision-making tasks with similar results.

Introduction

A significant body of work exists on the problem of learning from human trainers (Thomaz and Breazeal 2006; Khan, Zhu, and Mutlu 2011; Cakmak and Lopes 2012), and specifically on the problem of learning from trainer-provided feedback (Knox and Stone 2009; Griffith et al. 2013). Much of this work focuses on learning from demonstration, which uses trainer-provided examples of a target behavior, while work on learning from feedback often treats the problem as one of maximizing numerical reward. While there have been exciting developments in both areas, we argue that neither is always an appropriate model of learning from human teachers. First, providing demonstrations is not always feasible or desirable. Second, the positive or negative feedback given by humans does not represent numerical reward.

Feedback is a form of discrete communication between a trainer and a learning agent, and that communication can follow many different *training strategies* describing how trainers choose what feedback to give. We show that training strategies followed by human teachers vary in the amounts of positive and negative feedback given. A trainer may, for example, provide positive feedback when the learner takes a correct action, but provide no response when the it takes an incorrect action. Knowing this, the learner could interpret the lack of a response as a form of feedback itself. If only positive feedback is given, then the lack of feedback is implicitly negative and *vice versa*. We report results of user studies that demonstrate human trainers using strategies under which a lack of feedback is meaningful.

We derive two Bayesian policy learning algorithms designed to model and leverage these feedback strategies. Our algorithms, which we call *Strategy-Aware Bayesian Learning* (SABL) and *Inferring Strategy-Aware Bayesian Learning* (I-SABL), are designed to learn with fewer discrete feedbacks than existing techniques, while taking as few exploratory actions as possible. We first describe our representation of trainer strategy and the SABL algorithm, for *contextual bandit* domains. We then describe the I-SABL algorithm, which can infer a trainer's unknown strategy based on the feedback they have given. Lastly, we extend these algorithms to sequential domains.

We demonstrate the effectiveness of these algorithms in both online user studies and experiments with simulated trainers. Results indicate that, with human trainers, our algorithms learn behaviors with fewer actions and less feedback (and so, less effort on the part of the trainers) than baseline algorithms that treat feedback as numerical reward. We also show that the I-SABL algorithm is able to infer trainers' strategies from the feedback provided and use that knowledge to improve learning performance. Results with simulated trainers in both a contextual bandit and a sequential domain demonstrate the generality and robustness of SABL and I-SABL, and show that our algorithms can be adapted to work in sequential domains with goal states.

Related Work

Our work is part of a growing literature on learning from human feedback. Thomaz and Breazeal (2006) treated human feedback as a form of guidance for an agent trying to solve a reinforcement learning (RL) (Sutton and Barto 1998) problem. Human feedback did not change the numerical reward from the RL problem, or the optimal policy, but improved exploration and accelerated learning. Their results show hu-

mans give reward in anticipation of good actions, instead of rewarding or punishing the agent's recent actions.

COBOT (Isbell et al. 2001) 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 (Knox and Stone 2009) has been shown to be effective for learning from human feedback in a number of task domains common in the RL research community. This algorithm is modeled after standard RL methods which learn a value function from human-delivered numerical rewards. At each time step the algorithm updates its estimate of the reward function for a state-action pair using *cumulative* reward.

More similar to our work, Knox et al. (2012) examine 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 increase the rate of feedback. Our work differs because we focus on leveraging how humans naturally provide feedback when teaching, not how to manipulate that feedback.

Of existing work, the approach by Griffith et al. (2013) is most similar to the algorithms presented in this paper. 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 the traditional approach of treating feedback as numeric reward. Both our work and theirs uses a model of the feedback distribution to estimate a posterior distribution over the trainer's policy. In contrast to that work, ours focuses on handling different trainer feedback strategies. Griffith et al. assume episodes without explicit trainer feedback were uninformative as to the trainer's policy (though still informative about the underlying MDP). The algorithms presented in this work, however, use knowledge of the trainer's strategy to extract policy information from episodes without explicit feedback. Further, our algorithms can infer this strategy from experience, and so can adapt to a particular trainer's strategy.

In addition to the work on learning from feedback, there is a growing body of work that examines how humans can teach agents by providing demonstrations of a sequential decision task (Cakmak and Lopes 2012), or by selecting a sequence of data in a classification task (Khan, Zhu, and Mutlu 2011). We argue that the algorithms presented in this work could be extended to combine information from feedback with information from task demonstrations, or from sequences of tasks, based on knowledge of how the trainer communicates using those modalities.

Motivation: Trainer Strategies

In our training paradigm, the learning agent takes an action and then *may* receive positive or negative feedback from the trainer. We hypothesize that trainers can differ in how they provide feedback, even when teaching the same behavior. For example, when the learner takes a correct action, one trainer might provide an explicit positive feedback while, another might provide no response at all.

Table 1: Breakdown of strategies observed in user studies

Strategy	Number of Participants
balanced feedback	93
reward-focused	125
punishment-focused	6
inactive	3

We classify a trainer's strategy by the cases in which they give explicit feedback. Under a *balanced feedback* strategy a trainer typically gives explicit reward for correct actions and explicit punishment for incorrect ones. A *reward-focused* strategy typically provides an explicit reward for correct actions and no response for incorrect actions, while a *punishment-focused* strategy typically provides no response for correct actions and explicit punishment for incorrect ones. An *inactive* strategy rarely gives explicit feedback of any type. Under a reward-focused strategy, the lack of feedback can be interpreted as an *implicit* negative feedback, while under a punishment-focused strategy, it can be interpreted as implicitly positive. To a strategy-aware learner, the lack of feedback can be as informative as explicit feedback.

Table 1 shows the number of participants who used each of the four strategy types. Balanced feedback means the user gave explicit feedback for correct and incorrect actions more than half of the time, while inactive means they gave explicit feedback less than half the time in both cases. Reward-focused means correct actions received explicit feedback more than half the time and incorrect actions received it less than half the time; punishment-focused is the opposite case. Note that all four types were employed, but that a large percentage of users followed a reward-focused strategy.

Methods

We represent the learning environment as a *contextual bandit* (Lu, Pal, and Pal 2010). We divide learning into *episodes*, in which an observation occurs, the learner takes an action, and the trainer may provide feedback. We assume that the trainer has an observation-action mapping λ^*, a *policy*, they wish to train the learner to follow. The trainer can provide discrete feedback for each action, which can be positive or negative, and each feedback has a fixed magnitude; that is, there are not different degrees of punishment or reward.

The SABL Algorithm

Here we present the Strategy-Aware Bayesian Learning (SABL) algorithm. SABL assumes the trainer's feedback depends only on the most recent observation and action taken. In this model, the trainer first determines if the action was consistent with the target policy λ^* for the current observation, with probability of error ϵ. If the trainer interprets the learner's action as correct, 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 $(1-\epsilon)(1-\mu^+)$, explicit punishment with probability $\epsilon(1-\mu^-)$, and no feedback with probability $(1-\epsilon)\mu^+ + \epsilon\mu^-$.

Algorithm 1 The SABL algorithm. The feedback distribution $p(f_t|o_t, a_t, \lambda^*(o_t) = a')$ is described by Equations 1, 2 and 3. $takeAction(a_t)$ does not return until the episode finishes.

$\forall o \in O, a \in A$: $P[o, a] \leftarrow \frac{1}{|A|}, t \leftarrow 0$
while user has not terminated learning **do**
  $o_t \leftarrow observeWorld()$
  $a_t \leftarrow \text{argmax}_{a' \in A} P[o_t, a']$
  $takeAction(a_t)$
  $f_t \leftarrow lastFeedback()$
  for all $a' \in A$ **do**
    $P[o_t, a'] \leftarrow p(f_t|o_t, a_t, \lambda^*(o_t) = a')P[o_t, a']$
  end for
  $P[o_t, *] \leftarrow normalize(P[o_t, *])$
  $t \leftarrow t + 1$
end while

Parameters μ^+ and μ^- encode the trainer's strategy. For example, μ^+=0 and μ^-=0 correspond to a balanced strategy where explicit feedback is always given for an action, while $\mu^+ = 0$ and $\mu^- = 1$ correspond to a reward-focused strategy, where only actions interpreted as correct receive explicit feedback. Putting these elements together, for episode t, we have a distribution over the feedback f_t conditioned on the observation o_t, action a_t, and the target policy λ^*,

$$p(f_t{=}f^+|o_t, a_t, \lambda^*){=}\begin{cases}(1-\epsilon)(1-\mu^+), & \lambda^*(o_t){=}a_t \\ \epsilon(1-\mu^+), & \lambda^*(o_t){\neq}a_t,\end{cases} \quad (1)$$

$$p(f_t{=}f^-|o_t, a_t, \lambda^*){=}\begin{cases}\epsilon(1-\mu^-), & \lambda^*(o_t){=}a_t \\ (1-\epsilon)(1-\mu^-), & \lambda^*(o_t){\neq}a_t,\end{cases} \quad (2)$$

$$p(f_t{=}f^0|o_t, a_t, \lambda^*){=}\begin{cases}(1-\epsilon)\mu^+ + \epsilon\mu^-, & \lambda^*(o_t){=}a_t \\ \epsilon\mu^+ + (1-\epsilon)\mu^-, & \lambda^*(o_t){\neq}a_t.\end{cases} \quad (3)$$

Here, f^+ is an explicit positive feedback, f^- is an explicit negative feedback, and f^0 is the lack of feedback. Using this model of feedback, SABL computes a maximum likelihood estimate of the trainer's target policy λ^* given the feedback that the user has provided; that is, it computes

$$\text{argmax}_\lambda\, p(h_{1\ldots t}|\lambda^* = \lambda),$$

where h_t is the training history of actions, observations, and feedback. If a user provides multiple feedbacks during an episode, SABL only considers the most recent, allowing a user to correct a mistaken feedback. Algorithm 1 is an outline of SABL. Note that only the current likelihood distribution is needed to compute the likelihood given a new episode.

I-SABL: Inferring unknown strategies

While SABL will perform well when it knows the trainer's μ^+ and μ^- parameters, in practice the trainer's strategy will likely be unknown. If, however, the learner knows from explicit feedback the correct action for some observations, it can infer the strategy by looking at the history of feedback for those observations. If, for example, more explicit feedback is given for correct actions than incorrect ones, then the strategy is likely reward-focused. Under SABL's probabilistic model we can treat the unknown μ values representing the trainer's strategy as hidden parameters, and can marginalize over possible strategies to compute the likelihood of a possible target policy λ. Inferring-SABL, or I-SABL, finds a maximum likelihood estimate of the target policy, given the training data. I-SABL attempts to find

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

where S is the set of possible training strategies (μ^+, μ^- values), $p(s)$ is uniform for all $s \in S$, and $h_{1\ldots t}$ is the training history up to the current time t.

The space of possible policies may be exponential in the number of observations, and so algorithms for approximate inference may be needed. Here, we use Expectation Maximization (Dempster, Laird, and Rubin 1977) to compute a maximum likelihood estimate of the target policy, and treat the unknown μ^+ and μ^- parameters as continuous, hidden variables ranging from 0 to 1. The ith EM update step is then

$$\lambda_{i+1}{=}\text{argmax}_{\lambda \in P}\int_0^1\int_0^1 p(\mu^+, \mu^-|h, \lambda_i) \ln p(h, \mu^+, \mu^-|\lambda) d\mu^+ d\mu^-,$$

where λ_i is the current estimate of the policy and λ_{i+1} is the new estimate of the policy. This can be simplified to maximizing the following for a policy's action for each observation o (details omitted for space):

$$\lambda_{i+1}(o) = \text{argmax}_{a \in A}\left[\alpha(h_a^{o,+} - h_a^{o,-}) + \beta h_a^{o,0}\right], \quad (4)$$

where $h_a^{o,+}$ is the number of positive feedbacks received for observation o and action a during training history h, and $h_a^{o,-}$ and $h_a^{o,0}$ are analogous terms for negative feedback and no feedback respectively. Additionally we define

$$\alpha = \ln\left[\frac{(1-\epsilon)}{\epsilon}\right]\int_0^1\int_0^1 p(h|\mu^+, \mu^-, \lambda_i) d\mu^+ d\mu^-, \text{ and}$$

$$\beta = \int_0^1\int_0^1 p(h|\mu^+, \mu^-, \lambda_i) \ln\left[\frac{(1-\epsilon)\mu^+ + \epsilon\mu^-}{\epsilon\mu^+ + (1-\epsilon)\mu^-}\right] d\mu^+ d\mu^-,$$

values of a simplification of the expectation step, which can be computed once for each EM update. Algorithm 2 is an outline of I-SABL.

Algorithm 2 The I-SABL algorithm. The $EMupdate(\lambda, h)$ function computes a new policy according to Equation 4.

$\lambda \leftarrow randomPolicy(), h \leftarrow \langle\rangle, t \leftarrow 0$
while user has not terminated learning **do**
  $o_t \leftarrow observeWorld()$
  $a_t \leftarrow \lambda(o_t)$
  $takeAction(a_t)$
  $f_t \leftarrow lastFeedback()$
  $h \leftarrow \langle h_0, \ldots h_{t-1}, (o, a, f)\rangle$
  $\lambda \leftarrow randomPolicy()$
  repeat
    $\lambda' \leftarrow \lambda$
    $\lambda \leftarrow EMupdate(\lambda, h)$
  until $\lambda = \lambda'$
  $t \leftarrow t + 1$
end while

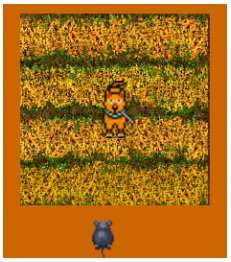


Figure 1: A screenshot of the study interface

Experiments

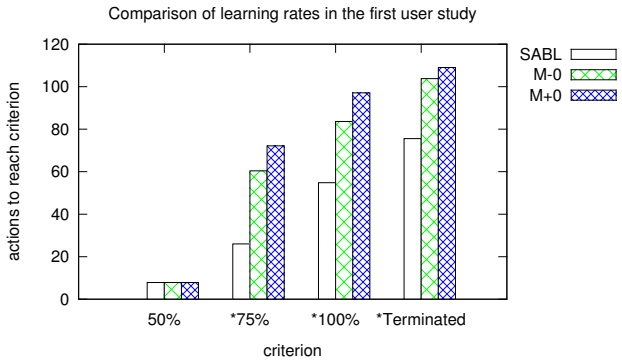
We compare SABL and I-SABL with variants of two algorithms from the literature on learning from human feedback via maximizing numerical reward. Both algorithms maintain an estimate of the expected reward associated with actions for each observation, but differ in their interpretation of no feedback. The first, denoted M_{-0}, is similar to TAMER (Knox and Stone 2009) and ignores episodes without feedback. The second, denoted M_{+0}, is similar to COBOT (Isbell et al. 2001) and includes episodes with zero reward — value estimates for actions will return to zero after enough episodes with no feedback. Both algorithms associate $+1$ with positive and -1 with negative feedback. Unlike SABL and I-SABL, M_{-0} and M_{+0} use the cumulative value of all feedback given during an episode.

User Studies

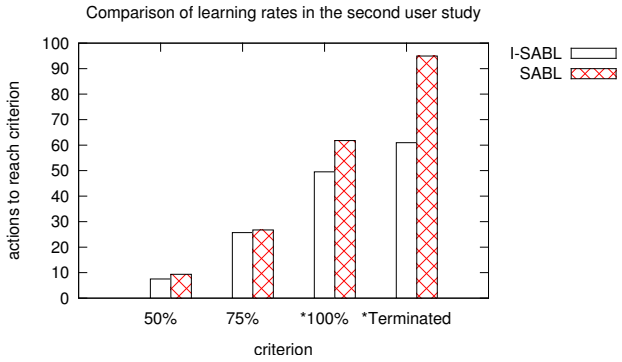
To evaluate their performance when learning from human trainers, we ran an online study in which participants trained learning agents using either SABL (with $\mu^+ = \mu^- = 0.1$), I-SABL, M_{-0}, or M_{+0}, to perform a contextual bandit task. We recruited two groups of users via email, online forums, and social networks to participate in our study: university students represented how the average computer-savvy user might train an agent, and amateur dog trainers represented users with experience training using discrete feedback.

Participants were asked to train an animated dog to chase rats away from a corn field. The dog was drawn at the center of the screen (Figure 1), and rats came one at a time every two seconds from three points along each of the four edges (twelve total observations). The learning algorithms were knew nothing about the spatial relationship between observations. The dog (learner) could move up, down, left, or right. Participants were instructed to provide rewards and/or punishments (using the keyboard) to teach the learner to move in the direction the rat was approaching from. Users decided when to terminate experiments, and were told to do so when they felt that either the dog had learned the task sufficiently well, or that it would not be able to learn further.

We ran two studies with this setup, first with participants from both the dog-training forums and the university, then with only participants from dog-training forums. The first study compared SABL against M_{-0} and M_{+0}, and had 126 users, of which 71 completed training at least one learner. The second compared I-SABL against SABL and had 43 users, of which 26 completed training at least one learner.



(a) First user study, comparing SABL, M_{-0} and M_{+0}



(b) Second user study, comparing SABL and I-SABL

Figure 2: Average number of episodes required to teach a policy that was correct for at least 50%, 75%, or 100% of observations, and until the participants terminated the session. (* indicates that differences were statistically significant for that column)

Our performance measure was the average number of steps it took each agent to reach each of four predetermined criteria. Three of the criteria were when the learner's estimate of the policy was 50%, 75%, and 100% correct. The fourth criterion was the number of steps before the user terminated the experiment. Results from the first user study show that learners using SABL tended to outperform those using M_{-0} and M_{+0}. Figure 2(a) shows the number of steps to reach each of the four criteria. The bars for SABL are lower than their counterparts for the other algorithms, showing that on average the SABL learner took fewer steps to reach the 75%, 100%, and the user termination criteria. Unpaired two sample t-tests show that the differences between the SABL learner and the M_{-0} and M_{+0} learners, for the 75%, 100% and termination criteria, were statistically significant ($p < 0.05$). In addition, a larger percentage of sessions using SABL reached 50%, 75%, and 100% policy correctness than using M_{-0} or M_{+0}. Pearson's χ^2 tests show that the differences between the number of times the SABL learner and the M_{-0} and M_{+0} learners reached the 100% criteria were statistically significant ($p < 0.01$), with the SABL, M_{-0} and M_{+0} learners reaching 100% correctness 53%, 17% and 19% of the time respectively.

In the second study, we compared I-SABL against SABL using the same performance criteria to test whether inferring trainers' strategies improves learning performance. Figure 2(b) shows the number of steps for each algorithm to reach the criteria. Of interest are the very small (statistically insignificant) differences between SABL and I-SABL for the 50% and 75% policy correctness criteria. The difference becomes much larger at the 100% and user-selected termination criteria, where I-SABL reaches each criteria in significantly fewer steps. This is expected, as improvements in learning performance for I-SABL will be most pronounced when the agent has received enough feedback for some observations to infer the trainer's strategy. Unpaired t-tests show these performance differences are statistically significant, with $p = 0.01$ for the 100% and $p < 0.05$ for the termination criteria. A larger percentage of sessions using I-SABL reached 50%, 75%, and 100% policy correctness before termination than using SABL. Pearson's χ^2 tests show that the differences between the number of times the I-SABL learner and the SABL learner reached the 100% criteria were significant ($p < 0.01$), with the I-SABL learner reaching 100% policy correctness 50% of the time, and the SABL learner reaching it 23% of the time, respectively.

We note that SABL took more episodes on average to learn in the second study than it did in the first study. We attribute this difference to the fact that users with dog training experience, who were much more common in the second study than in the first, were more likely to use a reward-focused training strategy. As the SABL algorithm assumed a balanced feedback strategy, it ignored episodes without feedback, and so performed more poorly under reward-focused strategies which provided fewer explicit feedbacks.

Simulated Trainer Experiments

To help understand how strategy inference allows I-SABL to outperform SABL, we ran several experiments with simulated trainers in contextual bandit domains, comparing I-SABL against SABL (with SABL's $\mu^+ = \mu^- = 0.1$). The simulated trainer chose a target policy at random, and generated feedback using the same probabilistic model underlying SABL and I-SABL. We tested each learning agent on tasks consisting of 2, 5, 10, 15 and 20 observations and 2, 3, or 4 actions. These experiments were conducted for a range of pairs of μ^+ and μ^- values for the simulated trainer. Each μ parameter was varied, from 0.0 to 0.8, such that $\mu^- + \mu^+ \leq 1$. The trainer's error rate ϵ=0.2, matching SABL and I-SABL's assumed value. Learners in these studies took actions at random but kept an estimate of the most likely policy.

The results show that I-SABL is able to take advantage of information from episodes where no explicit feedback is given. Figure 3 shows two curves representing the number of steps it took the SABL and I-SABL agents to find the correct policy, for varying μ parameters. The difference in performance between I-SABL and SABL increases (in favor of I-SABL) as the trainer's μ parameters diverge from the balanced strategy that SABL assumes. I-SABL compares well to SABL even when the trainer follows a balanced strategy.

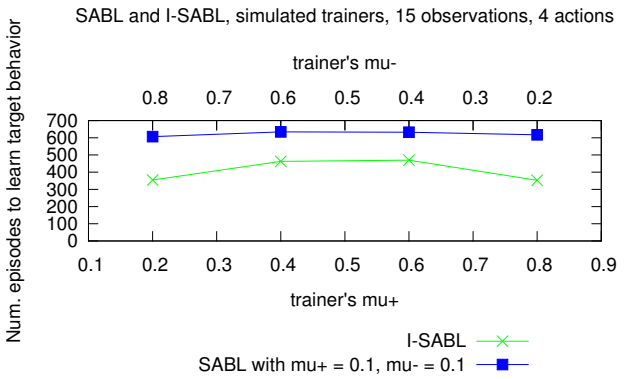


Figure 3: Performance of I-SABL and SABL (μ^-=μ^+=0.1) with simulated trainers. The bottom x-axis is the trainer's μ^+, the top x-axis is μ^-, and the y-axis is the number of episodes to find the target policy. As the difference between μ^+ and μ^- grows, so too does the performance difference between SABL and I-SABL.

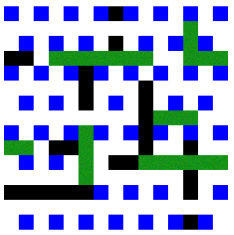


Figure 4: The sequential domain. Blue squares represent possible goal states, black squares represent obstacles of type one and grainy green squares represent obstacles of type two.

Sequential Tasks

Results presented so far show SABL and I-SABL in contextual bandit domains. We can also apply these algorithms to sequential decision making domains. For efficiency, we limit the set of policies considered by SABL and I-SABL, by assuming that the trainer teaches an optimal policy for some reward function from a set of reward functions defined over the domain. In a grid world, for example, the trainer could to teach the agent to reach a goal location where some reward will be received. Note the use of explicitly defined reward functions here is a syntactic convenience, not a requirement.

We tested SABL and I-SABL for sequential domains in a 15 by 15 grid world with a simulated trainer. The algorithms considered 48 possible goal states, as well as two special kinds of "obstacles"—states the agent could move in to or out of but may have needed to avoid—depending on the reward function. Each possible reward function returned a value of one when the agent reached the goal location, -100 when the it entered an obstacle type that was to be avoided, and zero otherwise. There were four different obstacle conditions (no obstacles, avoid type one, avoid type two, avoid both types), resulting in $48 \times 4 = 192$ possible reward functions. Figure 4 shows the grid world used. Note that the learners did not actually receive any numeric reward, and so could only learn the correct behavior based on trainer

Table 2: For all algorithm and simulated trainer pairs tested, the average number of steps before the agent correctly identified the intended policy as the most likely, and the average number of explicit feedbacks that were provided before the intended task was identified as the most likely. "N/A" indicates that the algorithm was unable to learn the correct policy in the majority of training runs.

Trainer's Strategy	Learning Algorithm	Identify Policy	95% Conf. Int.	# Explicit Feedbacks	95% Conf. Interval
balanced	I-SABL	44.4	±11.7	39.1	±10.4
	SABL - balanced feedback	46.7	±9.3	40.5	±8.1
	SABL - reward-focused	67.3	±21.1	60.0	±19.3
	SABL - punishment-focused	65.6	±20.6	58.1	±18.5
reward-focused	I-SABL	68.7	±20.5	54.1	±17.7
	SABL - balanced feedback	152.8	±27.9	71.4	±18.2
	SABL - reward-focused	65	±23.8	50.8	±20.4
	SABL - punishment-focused	N/A	N/A	N/A	N/A
punishment-focused	I-SABL	76.2	±25.4	14.8	±3.9
	SABL - balanced feedback	190.9	±27.3	37.4	±4.5
	SABL - reward-focused	N/A	N/A	N/A	N/A
	SABL - punishment-focused	51.3	±17.9	11.1	±2.8

feedback. In the sequential case, SABL and I-SABL simply assumed that the trainer's target policy was optimal for one of the possible reward functions. Before running SABL and I-SABL, the reward functions were converted to policies by solving the associated Markov Decision Processes.

In this case SABL and I-SABL only considered a small, finite set of possible μ parameter combinations, representing balanced, reward-focused, and punishment-focused trainer strategies. Additionally, to leverage this simplification rather than use EM on the entire feedback history at each step, we adapted I-SABL to update its prior belief in each strategy and policy to the posterior probability distribution given by the most recent feedback and the current distribution over trainer strategies. Trainer strategies were defined by $\{\mu^+, \mu^-\} = \{0.1, 0.1\}$ for the balanced feedback strategy, $\{\mu^+, \mu^-\} = \{0.1, 0.9\}$ for the reward-focused strategy, and $\{\mu^+, \mu^-\} = \{0.9, 0.1\}$ for the punishment-focused strategy. We did not consider the inactive strategy, as it was uncommon in the user study. For all strategies, $\epsilon = 0.05$.

Table 2 summarizes the results for all algorithm and trainer strategy pairs. For all simulated trainers, I-SABL and SABL using the correct feedback strategy identified the intended policy the fastest, again demonstrating that I-SABL does not suffer significantly from initial uncertainty about the trainer strategy. When the simulated trainer used a balanced strategy, SABL using incorrect strategy assumptions performed worse, but not significantly worse, likely due the fact that the simulated trainer almost always gave explicit feedback. Regardless of their strategy assumption, SABL learners always interpret explicit feedback in the same way. However, when the trainer does not employ a balanced strategy, incorrect SABL assumptions will be more problematic. If SABL assumes a balanced feedback strategy while the trainer follows a reward-focused strategy, the policy can be learned, but more steps are needed to do so because many steps receive no explicit feedback and so are ignored. If SABL assumes the opposite strategy (*e.g.*, assuming punishment-focused when it is actually reward-focused), then the agent may never learn the correct policy. Assuming the opposite strategy likely performs so poorly because it misinterprets what a lack of feedback means. If SABL assumes a punishment-focused strategy when it's actually a reward-strategy, it will interpret the lack of feedback when it's action is incorrect as evidence that it is correct.

In these results it is interesting to note how few explicit feedbacks are required for I-SABL and SABL (with a correct strategy assumption) to learn when the trainer follows a punishment-focused strategy. As it learns, more of the agent's actions are correct, resulting in less explicit feedback; since I-SABL (and SABL assuming a punishment-focused strategy) correctly interpret this lack of explicit feedback as positive, it does not hinder learning.

Conclusion

Initially we argued that most existing work on learning from feedback, which treats trainer feedback as a numerical reward, is not always sufficient for describing the ways in which human trainers provide feedback. We presented empirical data indicating that humans deliver discrete feedback and follow different training strategies when teaching. We have developed two Bayesian learning algorithms, SABL and I-SABL, that can leverage knowledge about those strategies. SABL encodes assumptions about trainer strategies as the probabilities of explicit feedback given the correctness of actions, and I-SABL infers those probabilities online.

Our user studies and simulation experiments demonstrate that the SABL and I-SABL algorithms learn in substantially fewer episodes, and with less feedback, than algorithms modeled after existing numerical-reward-maximizing algorithms. We have demonstrated this advantage even in cases where the trainer's strategy is initially unknown. Further, we have shown this approach can be effective both in contextual bandit and sequential decision making domains.

Future work would expand the space of feedback strategies considered by SABL and I-SABL to allow temporal delays and variable feedback distributions, and to incorporate knowledge from trainer demonstrations. Future work would also consider how these algorithms can be applied to sequential tasks without enumerating possible reward functions, allowing them to be used in more complex domains.

Acknowledgements

This work was supported in part by NSF IIS-1149917 and NSF IIS-1319412.

References

Cakmak, M., and Lopes, M. 2012. Algorithmic and human teaching of sequential decision tasks. In *Proceedings of AAAI*.

Dempster, A. P.; Laird, N. M.; and Rubin, D. B. 1977. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* 39(1):pp. 1–38.

Griffith, S.; Subramanian, K.; Scholz, J.; Isbell, C.; and Thomaz, A. L. 2013. Policy shaping: Integrating human feedback with reinforcement learning. In Burges, C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 2625–2633.

Isbell, C.L., J.; Shelton, C.; Kearns, M.; Singh, S.; and Stone, P. 2001. A social reinforcement learning agent. In *Proceedings of the International Conference on Autonomous Agents*, 377 – 384.

Khan, F.; Zhu, X. J.; and Mutlu, B. 2011. How do humans teach: On curriculum learning and teaching dimension. In *Proceedings of NIPS*, 1449–1457.

Knox, W. B., and Stone, P. 2009. Interactively shaping agents via human reinforcement: The tamer framework. In *Proceedings of the fifth international conference on Knowledge capture*, 9 – 16.

Knox, W. B.; Glass, B. D.; Love, B. C.; Maddox, W. T.; and Stone, P. 2012. How humans teach agents - a new experimental perspective. *I. J. Social Robotics* 4(4):409–421.

Lu, T.; Pal, D.; and Pal, M. 2010. Contextual multi-armed bandits. In *Proceedings of the 13th international conference on Artificial Intelligence and Statistics*, 485–492.

Sutton, R., and Barto, A. 1998. *Reinforcement learning: An introduction*. Cambridge, MA, USA: MIT Press, 1st edition.

Thomaz, A. L., and Breazeal, C. 2006. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, 1000. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.