Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Where to Add Actions in Human-in-the-Loop Reinforcement Learning

Travis Mandel,[1] **Yun-En Liu,**[2] **Emma Brunskill,**[3] **Zoran Popović**[1,2]
[1]Center for Game Science, Computer Science & Engineering, University of Washington, Seattle, WA
[2]Enlearn™, Seattle, WA
[3]School of Computer Science, Carnegie Mellon University, Pittsburgh, PA
{tmandel, zoran}@cs.washington.edu, yunliu@enlearn.org, ebrun@cs.cmu.edu

Abstract

In order for reinforcement learning systems to learn quickly in vast action spaces such as the space of all possible pieces of text or the space of all images, leveraging human intuition and creativity is key. However, a human-designed action space is likely to be initially imperfect and limited; furthermore, humans may improve at creating useful actions with practice or new information. Therefore, we propose a framework in which a human adds actions to a reinforcement learning system over time to boost performance. In this setting, however, it is key that we use human effort as efficiently as possible, and one significant danger is that humans waste effort adding actions at places (states) that aren't very important. Therefore, we propose Expected Local Improvement (ELI), an automated method which selects states at which to query humans for a new action. We evaluate ELI on a variety of simulated domains adapted from the literature, including domains with over a million actions and domains where the simulated experts change over time. We find ELI demonstrates excellent empirical performance, even in settings where the synthetic "experts" are quite poor.

1 Introduction

Consider building a system to decide which hint to give a student playing an educational game, what to say to a user of a spoken dialog system, or what pictorial ad to show to generate the most revenue for a small business. In domains such as these, the space of all possible actions (pieces of text, soundwaves, or images) is far too large to explore from scratch without unreasonable quantities of data. A typical reinforcement learning (RL) approach is for human practitioners to first use their intuition and creativity to create a small set of discrete actions, and then use reinforcement learning techniques to learn to behave near-optimally within this more tractable space.

However, this initial action set is likely to be limited and highly imperfect, and so the performance of even the truly optimal policy with respect to the limited action space may be far below what is truly achievable. Because of this, we desire systems that add new actions to the set over time to come closer to optimal performance. Indeed, in practice it is common to try to improve performance further by creating new content, for example by writing new lines of dialog or designing new ads.

Unfortunately, automatic guidance on this process has been extremely limited. It has typically been the task of domain experts to determine not just *what* new content to produce, but *where* to produce it. For example, experts must determine where to add a new line of dialog in a dialog system or a new hint in a educational game. In an RL framework, we can think of this as selecting the state where the next action should be added. Unfortunately, this task is difficult, as it requires the human to possess substantial amounts of domain expertise, machine learning expertise, and be able to understand large quantities of data. Ideally, we could develop automated methods that answer this question, deciding *where* to direct human effort to best maximize performance.

Related work in human-in-the-loop reinforcement learning has looked at where to query experts for demonstrations. A significant amount of work has focused on on querying the expert in a different setting where the agent's goal is merely to imitate (Chernova and Veloso 2009; Judah et al. 2014). Related work by Clouse 1996 studied at which state a reinforcement learning agent should query an imperfect expert for advice in an existing action set. However, this is a very different setting, as the assumption is that the human, though not perfect, can perform the task fairly well, and that the action space is sufficiently small to be amenable to autonomous exploration.

In this paper, we propose Expected Local Improvement (ELI), as a heuristic to select the state where the next action should be added. ELI intelligently combines data gathered by a reinforcement learning algorithm, knowledge of the structure of the reinforcement learning problem (if present) and the behavior of the human experts it is interacting with to select states that will be most likely to boost performance. Although the ultimate goal is to run ELI with real humans on important real-world problems, similar to prior work (Clouse 1996; Amir et al. 2016) we evaluate algorithms in settings where both the RL environment and the "human" expert are simulated, as this allows for inexpensive and well-controlled comparisons. We find that ELI performs well across a variety of domains adapted from the literature, including a domain with over a million actions, a domain where the experts improve over time, and even a domain where the experts are quite poor.

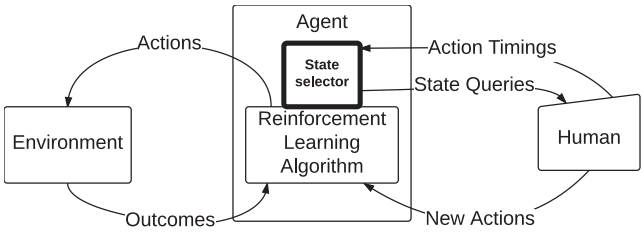

Figure 1: Proposed Human-in-the-Loop RL framework, in which a human provides new actions in response to state queries. Here we focus on the design of the state selector.

2 Problem Setup

Although there has been much recent work focusing on the problem of effective generalization over a large, high dimensional state space in reinforcement learning, this task becomes much more complicated when the set of actions is not fixed but can be extended. As such, in this paper we focus on the more traditional RL setting where a relatively small, discrete state space is known. Although we wish to consider Markov Decision Process (MDP) environments, a typical "tabula rasa" MDP setting does not fit many real-world scenarios where more structure is known.[1] Therefore, similar to past work (Leffler, Littman, and Edmunds 2007; Asmuth et al. 2009; Mandel et al. 2016), we consider MDP domains where dynamics are specified in terms of relative outcomes.[2] Specifically, we assume a discrete state space $\mathcal{S}$ and a set of relative outcomes $\mathcal{O}$. We assume experience comes in episodes of maximum length τ. The agent starts with at least one action in $\mathcal{A}_{0s}$ for each s. At the start of the ℓ^{th} episode, generally $\mathcal{A}_{\ell,s} = \mathcal{A}_{\ell-1,s}$. However, if the human decides it is time to add a new action, the agent must pick a state s from $\mathcal{S}$, and the human proceeds to add an action[3] to $\mathcal{A}_{\ell,s}$. During the course of the ℓ^{th} episode, the agent is at a state $s \in \mathcal{S}$, takes an action $a \in \mathcal{A}_{s,\ell}$ and receives an outcome $o \in \mathcal{O}$. The agent knows the reward function $R(s, o)$ which deterministically outputs a scalar reward, and the transition function $T(s, o)$, which deterministically outputs a next state s'. However, it does not know the distribution over relative outcomes at each state and action, nor the process by which the human generates actions. The goal of the agent is to learn from experience to maximize the sum of rewards over time, both by picking good actions in the existing set during an episode and by selecting good states at which to add actions. In this paper, we assume the former task is handled by a traditional reinforcement learning algorithm and focus on developing new algorithms for the latter problem. For convenience, let $S = |\mathcal{S}|$, $A_\ell = \max_s |\mathcal{A}_{s,\ell}|$, and $O = |\mathcal{O}|$. An overview of the considered framework can be seen in Figure 1.

[1]For example, consider a gridworld where we know an agent can go in cardinal directions, and we know the location of the goal state but not where the walls are.

[2]Note, however, that in principle outcomes can encode any MDP with discrete rewards and states.

[3]We also allow humans to add extra actions at other states, e.g. if they feel the action they developed is appropriate elsewhere.

3 Related work

Active Imitation Learning One related setting is active imitation learning (Grollman and Jenkins 2007; Chernova and Veloso 2009; Judah, Fern, and Dietterich 2011; Silver, Bagnell, and Stentz 2012; Judah et al. 2014). Here the agent's goal is to query an expert for demonstrations at states that will best help it improve its performance. However, in this setting the agent is not trying to optimize an external reward function, just imitate the expert, and therefore the challenges involved in combining autonomous reinforcement learning with human expertise do not arise. Additionally, state-of-the-art work in this area (Judah et al. 2014) assumes the action space is small and the experts are deterministic, making the techniques proposed ill-suited for our purpose.

Interactive/Human-in-the-Loop Reinforcement Learning There have been several studies of how humans should collaborate with reinforcement learning agents. Several interesting types of human feedback have been considered: Reward signals (Thomaz and Breazeal 2006), policy quality information (Griffith et al. 2013), entire trajectories (Gil, Stern, and Edan 2009), and more. We are only aware of very limited work (in the bandit setting) considering added actions (Abernethy et al. 2013; Williams et al. 2016), which did not consider automated guidance on where to add actions.

Some work has focused on the related problem of deciding at which state to query expert for advice on the next action to take. For instance, Doshi-Velez et al. 2008 studies this problem but assumes the expert is perfect and the reward signal is inaccurate. Most closely related to our work is Clouse 1996, who proposes a heuristic method to determine when (at what state) an imperfect expert should be asked for advice in an online RL setting. The setting is different as Clouse deals with giving advice in an existing (small) action space, whereas we consider where one should extend the action space. In addition, the proposed heuristic asks for states at which the agent is uncertain, which does not make sense in our setting, since if we are uncertain about the values of the current actions at some state we should refrain from adding additional actions until we are more certain.

Large Action Spaces We propose leveraging human intuition to enable rapid learning in vast action spaces. However, an alternative approach is to use function approximation to generalize over a large action space. Work in large discrete action spaces has typically assumed actions are featurized in a highly informative way (Sallans and Hinton 2004; Dulac-Arnold et al. 2015) and can still show poor performance when learning from scratch without prior knowledge. Past work in learning in large action continuous control problems (Lillicrap et al. 2016) exploits knowledge of physical principles and large amounts of simulator data.

In the domain of conversational agents, related work has learned which actions to consider taking from a corpus of human conversations (Li et al. 2016). However, this work focuses on making use of a passively collected dataset of actions, we show in this paper we can do better by intelligently directing human effort.[4]

[4]An additional advantage to adding actions online is that infor-

4 Expected Local Improvement (ELI)

To ground our discussion, we assume that each state s is associated with an (unknown) distribution over actions with pdf $p_s(x)$, so that when we request an action at s the expert draws a (possibly continuous) action x sampled i.i.d. according to $p_s(x)$. [5] In some settings this stochastic assumption is reasonable, but in others it may not be, such as experts learning to improve the provided actions over time. We will empirically evaluate robustness to improving experts, as well as show a non-starving property (Lemma 1) under weaker assumptions.

Although our goal is to select states optimally under uncertainty, this is not a straightforward application of a standard exploration/exploitation approach (e.g. bandits), since:

- We wish to leverage the structure of the underlying RL problem to select states intelligently, instead of treating the problem in a tabula rasa fashion.
- When a bandit algorithm encounters a positive reward, it is inclined to keep selecting the action that led to that reward. But here, a positive (high-value) action means just the opposite: we have found an action that performs well at this state so we should avoid selecting this state in favor of trying to improve performance elsewhere.
- Typical bandit algorithms learn the mean of the reward distribution, but here optimal behavior depends on higher moments of the action distribution. For example, imagine the action distributions at s_1 and s_2 are distributed normally (in terms of the action values) with identical means, but the distribution at s_1 has much higher variance than that at s_2. Then we should greatly prefer to select s_1 as we are much more likely to get a high-value action.
- We need to consider the uncertainty not just over the unknown distribution of new actions but also the unknown values of existing actions.

We desire methods that carefully direct human effort boost performance quickly, but not at the expense of long-term performance. Additionally, we want a heuristic that is scalable and easy to compute, without, for example, requiring explicitly planning over sequences of action additions.[6]

One heuristic that seems natural to consider is greedily adding the action that maximizes expected global improvement. However, here greediness becomes a significant detriment to long-term performance, as we are not incentivized to add actions that do not immediately improve performance but instead open up pathways to new states that have the potential to be high-value after adding more actions.

Perhaps surprisingly, a less global approach alleviates this issue. We propose maximizing the expected local improvement (ELI), that is, selecting the state s where adding the next action most improves the (optimal) value $V(s)$. Formally, we define the ELI of state s as follows:

$$\int (V(s|x, \mathcal{A}_{s,\ell}) - V(s|\mathcal{A}_{s,\ell}))p_s(x)dx \tag{1}$$

where $V(s|x, \mathcal{A}_{s,\ell})$ is optimal value we would get at state s given our current set of actions and an additional action x.

This will result in a bottom up approach, where values of states where there is a high potential for immediate reward will be improved first and, once they are improved, we try to add actions to other states that transition to the high-value leaves. One potential downside of this approach is that we may prioritize adding actions at states that are (near) impossible to reach. However, in practice our approach for dealing with unknown dynamics alleviates this issue (see below).

Now, observe that for x such that $V(s|x, \mathcal{A}_{s,\ell}) = V(s|\mathcal{A}_{s,\ell})$ (i.e. actions which do not improve the value) the integral is zero so it suffices to take the integral over x such that $V(s|x, \mathcal{A}_{s,\ell}) > V(s|\mathcal{A}_{s,\ell})$:

$$\int (V(s|x, \mathcal{A}_{s,\ell}) - V(s|\mathcal{A}_{s,\ell}))p_s(x)dx \tag{2}$$

$$= \int_{x:V(s|x,\mathcal{A}_{s,\ell})>V(s|\mathcal{A}_{s,\ell})} (V(s|x, \mathcal{A}_{s,\ell}) - V(s|\mathcal{A}_{s,\ell}))p_s(x)dx \tag{3}$$

Next, consider estimating $V(s|x, \mathcal{A}_{s,\ell})$. An underestimate is dangerous here, as it may lead to ignoring (starving) states where there is a large potential for improvement. Therefore we follow the well-known principle of *optimism under uncertainty*, and overestimate the improvement by $V_{max}(s) = \max_x V(s|x, \mathcal{A}_{s,\ell})$, which is easy to compute with discrete outcomes as long as we have a way of estimating $V(s|\mathcal{A}_{s,\ell})$ for future states (see below). This gives us:

$$\int_{x:V(s|x,\mathcal{A}_{s,\ell})>V(s|\mathcal{A}_{s,\ell})} (V(s|x, \mathcal{A}_{s,\ell}) - V(s|\mathcal{A}_{s,\ell}))p_s(x)dx \tag{4}$$

$$\leq \int_{x:V(s|x,\mathcal{A}_{s,\ell})>V(s|\mathcal{A}_{s,\ell})} (V_{max}(s) - V(s|\mathcal{A}_{s,\ell}))p_s(x)dx \tag{5}$$

$$= (V_{max}(s) - V(s|\mathcal{A}_{s,\ell})) \int_{x:V(s|x,\mathcal{A}_{s,\ell})>V(s|\mathcal{A}_{s,\ell})} p_s(x)dx \tag{6}$$

$$= (V_{max}(s) - V(s|\mathcal{A}_{s,\ell}))P(V(s|x, \mathcal{A}_{s,\ell}) > V(s|\mathcal{A}_{s,\ell})) \tag{7}$$

However, calculating this requires estimating two unknown quantities: $P(V(s|x, \mathcal{A}_{s,\ell}) > V(s|\mathcal{A}_{s,\ell}))$ and $V(s|\mathcal{A}_{s,\ell})$. We discuss how this can be done below.

Estimating the value of existing actions

Let us first consider estimating $V(s|\mathcal{A}_{s,\ell})$, the value of a state given existing actions. Intuitively, if we don't know if the existing actions are high-value it seems wasteful to add another action there. Therefore, we propose using an optimistic estimate of the current value (and thus underestimate how much improvement is left). At first this seems contradictory, as we were trying to overestimate the improvement. However, we must consider the *source* of the uncertainty.

mation such as values of the current actions can be displayed to humans, to hopefully guide them to produce better actions.

[5] In some settings, such as text, the action space is finite or countably infinite, in which case we could use probabilities and sums instead of pdfs and integrals. However, here we solely consider the continuous case to simplify exposition.

[6] One could consider casting the problem in a Bayesian RL-type framework (e.g. (Wang et al. 2005; Whittle 1981)). However, the computational expense of this approach is prohibitive.

If the source is a low number of added actions, using the principle of optimism under uncertainty we should overestimate the improvement of the next action, as adding an additional action will reduce uncertainty. But if the source is a low number of samples of existing actions, we should apply the principle of optimism under uncertainty again to the values of existing actions, as even though we cannot directly act there, when the RL algorithm does it will reduce our uncertainty. These two forms of optimism under uncertainty in ELI act synergistically by properly accounting for the source of the uncertainty and selecting states accordingly.

An additional benefit of overestimating $V(s|\mathcal{A}_{s,\ell})$ is that we will not select (nearly) impossible to reach states because we are extremely uncertain about their current values.

Fortuitously, the task of optimistically estimating the value of a transition/reward distribution given limited samples has been well-studied in the RL community as a way to drive exploration. These methods deal with optimistic long-term value in a simple dynamic programming fashion; calculating the optimistic $V(s,t)$ by treating the previously-calculated optimistic $V(s,t+1)$ as fixed and calculating an optimistic outcome distribution for each (s,a,t) tuple. The key difference among methods is in how the optimistic outcome distribution is calculated. We try a number of approaches for this component (see the appendix[7] for details):

UCRL Following UCRL2 (Jaksch, Ortner, and Auer 2010), this computes the most optimistic outcome distribution subject to the constraint that the L1 norm varies from the MLE by at most $\sqrt{\frac{14\log(SA_\ell\tau\ell/\delta)}{\max(N,1)}}$ where ℓ is the number of episodes, N is the number of transition samples, and δ is a user-specified confidence parameter we set to 0.05 following Osband et al. 2013.

MBIE Adapting a version of MBIE (Strehl and Littman 2004; 2008) to our setting, this computes the most optimistic outcome distribution subject to the constraint that the L1 norm varies from the MLE by at most $\sqrt{\frac{2\log((2^O-2)/\delta)}{\max(N,1)}}$.

(Optimistic) Thompson Sampling This weakly optimistic approach (Thompson 1933; May et al. 2012) draws a single sample from the posterior over outcome distributions under the constraint that the sample has greater value than the expected value under the posterior.

BOSS This approach (Asmuth et al. 2009) samples from the posterior over outcome distributions J times and returns the sample with maximum value. We set $J=10$ based on the results in Asmuth et al. 2009.

Of all the optimistic estimators, BOSS seems most attractive, since it combines prior information with a significant amount of optimism. Therefore for simplicity of exposition, we hereafter refer to ELI-BOSS as just ELI.

Estimating the probability of improvement

We now examine estimating $P(V(s|x,\mathcal{A}_{s,\ell}) > V(s|\mathcal{A}_{s,\ell}))$. A naïve approach is to replace this probability by 1 to trivially upper bound the integral, which intuitively corresponds to optimistically believing the next action will certainly result in a (maximal) improvement, avoiding i.i.d assumptions. The issue here becomes that the non-starving property is clearly violated, as we may simply sample the same state forever if no improvement is truly possible. A simple way to ensure this problem does not occur is simply to set a limit of L, such that we move on to another state if the selected state already has L actions. If all visited states have L actions we increment L. In general, it is unclear how to initialize L, so we set the minimum L of 1, allowing L to grow automatically to find the best value. We call this method **ELI-Limit**.

Intuitively, however, if we have selected a state numerous times already, it seems unlikely the next action will result in an improvement. To formalize this intuition, we define a binary variable X_m for each m such that $X_m=1$ if the next action has Q-value greater than m and $X_m=0$ otherwise. Now, since X_m is binary, we can learn it in a Bayesian fashion using a $Beta(\alpha_m+n_{1m},\beta_m+n_{0m})$ posterior where α_m and n_{1m} represent the prior psuedocount and observed count of actions with Q-value greater than m, and likewise β_m and n_{0m} represent the prior psuedocount and observed count of actions with Q-value less than m. For the prior we let $\alpha_m=\beta_m=1$ for all m, which corresponds to one pseudocount of maximum and minimum Q-value. Now in particular we are interested in the probability of further improvement, which corresponds to letting m be the maximum value over the existing actions ($m=\max_{a\in\mathcal{A}_{s,\ell}}Q(s,a)$). Therefore, after $|\mathcal{A}_{s,\ell}|$ added actions our posterior over the probability of improvement is $Beta(1,|\mathcal{A}_{s,\ell}|+1)$ regardless of the Q-values.[8] Note that for a fixed m, the observed Q-values do matter, but since we keep changing m to be the maximum Q-value seen so far, the variable of concern X_m keeps changing, and so they do not.

Given this posterior distribution over $\theta = P(V(s|x,\mathcal{A}_{s,\ell}) > V(s|\mathcal{A}_{s,\ell}))$, a Bayesian approach is to integrate the objective function over all possible values of θ weighted by their posterior probability $P(\theta|\mathcal{A}_{s,\ell})$. Specifically, this changes equation (7) into:

$$\int (V_{max}(s)-\hat{V}(s|\mathcal{A}_{s,\ell}))\theta P(\theta|\mathcal{A}_{s,\ell})d\theta \tag{8}$$

$$=(V_{max}(s)-\hat{V}(s|\mathcal{A}_{s,\ell}))\int \theta P(\theta|\mathcal{A}_{s,\ell})d\theta \tag{9}$$

$$=(V_{max}(s)-\hat{V}(s|\mathcal{A}_{s,\ell}))E[\theta|\mathcal{A}_{s,\ell}] \tag{10}$$

where the last line comes simply from the definition of expectation. The expected value of θ under our $Beta(1,|\mathcal{A}_{s,\ell}|+1)$ posterior derived above is simply $\frac{1}{|\mathcal{A}_{s,\ell}|+2}$. So the overall ELI heuristic is:

$$=\frac{1}{|\mathcal{A}_{s,\ell}|+2}(V_{max}(s)-\hat{V}(s|\mathcal{A}_{s,\ell})) \tag{11}$$

Since we are in a finite-horizon setting, values depend not just on s but on s,t. Our RL setting is undiscounted, so we

[7] The appendix can be found at http://grail.cs.washington.edu/projects/stateselection/

[8] Note that this is not correct in the edge case where we already have an action with maximum value, as the probability of improvement there should always be zero. However, in that case $(V_{max}(s)-\hat{V}(s|\mathcal{A}_{s,\ell}))$ should be zero resulting in zero score.

simply sum up the scores for each s, t to generate an overall score for each s, and select the state with maximum score.

One property we desire is that ELI does not starve states as long as there is possible improvement remaining. This is a fairly weak property, as we could achieve it (at the cost of performance) by mixing in an ϵ-amount of uniformly random state selection. Intuitively, it seems as though ELI already has this property, as it does not starve states as long as there is still some difference between $\hat{V}(s)$ and $V_{max}(s)$. However, if the human adds rapidly adds large numbers of actions to all successor states s' of s and they are insufficiently explored, their values may go to $V_{max}(s')$ (because we take the max over a larger and larger set of random samples) and so (since $\hat{V}(s)$ is defined in terms of the $\hat{V}(s')$) it is possible that there becomes less and less difference between $\hat{V}(s)$ and $V_{max}(s)$ even if we do not add further actions at s. So showing ELI does not starve states as long as there is possible improvement is a bit nontrivial and requires additional assumptions. We do not wish to restrict the RL algorithm or the behavior of the human (w.r.t. action timings etc.), so we prove ELI is non-starving in a fairly general class of environments similar (just slightly more restricted) to previous posterior sampling work by Osband et al. 2013.

Lemma 1. ***(Non-starving)*** *Consider ELI using a prior distribution f which consists of an independent Dirichlet prior on outcome distributions of $\alpha_i = c$ for some $c > 0$. Assume for a given $\epsilon > 0$, after N_ϵ actions are added to each state, additional actions improve the value of each state by at most ϵ. Let $\mathcal{C}$ be an arbitrary class of models with N_ϵ actions which has non-zero probability under our chosen prior. Assume that the true model M for the first N_ϵ actions is drawn from $\mathcal{C}$ according to $f(M|\mathcal{C})$.Finally, assume that for each s there exists $o_1, o_2 \in \mathcal{O}$ such that $T(s, o_1) = T(s, o_2); R(s, o_1) \neq R(s, o_2)$. Then, as the number of actions added by ELI goes to infinity, our ELI approach will eventually uncover actions at each state such that the optimal policy in the MDP with added actions is at least ϵ-optimal (with respect to the full set of actions).*

Proof sketch (full proof in appendix): The only way for us to starve a state is for its ELI score to go to zero. Since we only select it a finite number of times, the only way for this to occur is if $\hat{V}(s, t)$ approaches $V_{max}(s, t)$. But the cases such that this occurs have zero probability under the posterior.

5 Simulations

We show results using both simulated environments and simulated humans, similar to prior work (Clouse 1996; Amir et al. 2016), as this allows us to compare algorithms in an inexpensive and well-controlled setting. Each state starts with a single action (similar to real-world settings where there is already a default strategy) and every 20 episodes a new action is added at the state the agent selects. We transform standard domains to this new setting by sampling a new (or initial) action uniformly at random from the ground space of actions, unless otherwise specified. Even with a fairly small space of ground actions, this is still a challenging problem, as (for example) the agent may unknowingly get the same suboptimal action repeatedly and need to select a state many times to get the optimal action.[9] All result graphs are averaged over 1000 runs.

ELI leaves open the choice of underlying RL algorithm. We use Posterior Sampling Reinforcement Learning (PSRL) (Osband, Russo, and Van Roy 2013; Strens 2000) due to its ability to explore efficiently. An additional advantage is that some variants of ELI can share the posterior used for PSRL.

We constrain all methods to only select states the agent has visited before.[10] We compare variations of ELI, **Random**, which selects a visited state uniformly at random, and **Frequency**, which selects states with probability proportional to the number of times they have been visited (i.e., explored) by the RL algorithm.

We examine performance on three environments adapted from the literature. **Riverswim** is a chain MDP with 6 states, 5 outcomes, and 2 ground actions per state that requires efficient exploration (Osband, Russo, and Van Roy 2013). **Marblemaze** (Asmuth et al. 2009; Russell and Norvig 1994) is a gridworld MDP with 36 states, 5 outcomes, and 4 ground actions per state that allows significant prior knowledge to be encoded in the outcomes framework. We use a modified version of the **Large Action Task** (Sallans and Hinton 2004), an environment with 13 states, 273 outcomes, and 2^{20} ground actions per state, a ground action space far too large to explore directly. Details of these domains are in the appendix.

Comparing algorithms The results in Riverswim (Figure 2) show that all variants of ELI except ELI-UCRL outperform the Random and Frequency baselines. In fact, we see that Frequency underperforms Random, because it focuses too heavily on high traffic states where no improvement is possible. ELI-UCRL likely underperforms due to its conservative confidence intervals, as it avoids selecting states unless they have been extremely well-sampled. We see ELI-Limit performs worse than our refined approach based on the Beta posterior. In this setting, however, the more strongly optimistic ELI-MBIE approach does seems to slightly outperform the proposed BOSS method, which in part indicates that our adapted MBIE confidence intervals are a better fit for our setting than the ones used in UCRL2.

In Marblemaze (Figure 3) we see similar trends to Riverswim, except that the boost in performance from Random to ELI is extremely large, and ELI also shows a large improvement over ELI-Limit. The performance improvements come from the fact that that ELI is able to leverage the small space of outcomes to focus on adding actions at states that are likely to be part of the optimal path to the goal.

In the Large Action Task (Figure 4) ELI again shows a strong improvement over random, and performs as well or better than the other ablated baselines. Here Frequency does better than Random, indicating that focusing on high-traffic states plays a much bigger role in proper selection than in

[9] We may be able to prevent humans from duplicating an action, but there is still the possibility that humans generate an action with a very similar outcome distribution, which is roughly equivalent.

[10] This is partially motivated by the fact that in real-world domains it may be difficult to visualize unvisited states to humans.

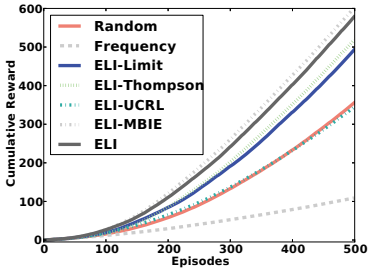

Figure 2: Riverswim results with a small outcome space.

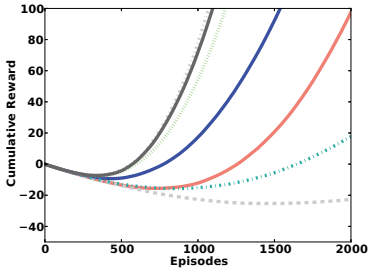

Figure 3: Marblemaze results with a small outcome space.

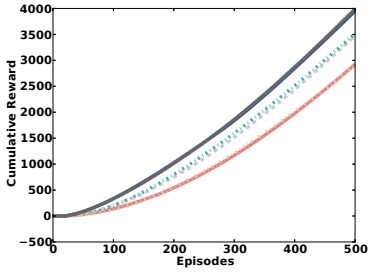

Figure 4: Results on the large-action task with random action generation.

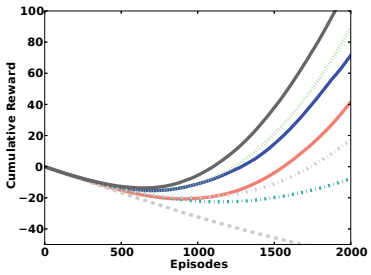

Figure 5: Marble Maze Results with a large outcome space.

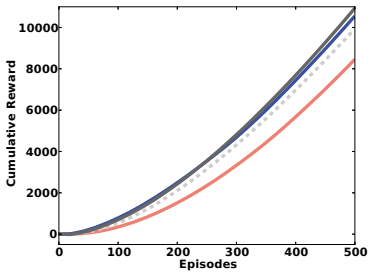

Figure 6: Results on the Large-Action task with improving action generation.

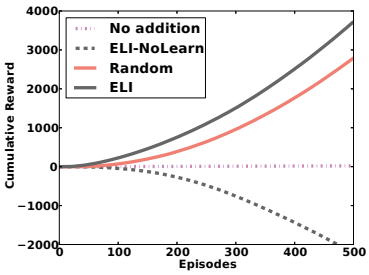

Figure 7: Results on the Large-Action task with poor action generation.

Riverswim and Marblemaze, which makes sense as states in this environment are unlikely to come close to their maximal values and so re-sampling is usually valuable. ELI-MBIE, which performed very well in riverswim and marblemaze, performs quite poorly here. We suspect this has to do with the larger set of outcomes used in the large action task, which we study with our next experiment.

Informative vs Uninformative Outcomes Although in some scenarios a small and informative set of outcomes is known, in other settings we unfortunately do not know this additional structure. To test how performance changes in this setting we modified the marblemaze setup to have 96 outcomes, one for each possible reward and next state (see appendix). In Figure 5 we see that although using uninformative outcomes did decrease the gap between ELI and Random compared to the informative outcomes case in Figure 3, ELI still performs quite well. These results confirm the poor performance of ELI-MBIE in large outcome spaces, which turns out to be due to the fact that the confidence interval used in MBIE causes it to become far too optimistic when the number of possible outcomes becomes large. The BOSS method does not suffer from this issue, though it is still optimistic enough to outperform ELI-Thompson here.

Different types of experts We now test a simulated expert who improves over time. When queried at a state the expert generates an action with equal (50%) or better (50%) immediate reward compared to the best action added at this state so far (details in appendix). In Figure 6 we see that even though this setting is nonstationary and does not match the assumptions built when designing ELI, the ELI method still manages to perform well, outperforming both Random and Frequency baselines. Though the ELI-Limit method does not explicitly make these assumptions it does not manage to improve upon ELI here.

Often, human-in-the-loop RL approaches are thought to only be helpful if the human has enough expertise to perform the task reasonably well. To examine this we simulate a "poor" expert, where the probability of generating an action decreases linearly as the immediate reward of that action increases (see appendix). For reference we display performance without action addition, and also ELI-NoLearn, which counterfactually measures how performance of a uniform random policy changes as ELI (fed with data from PSRL) adds actions. In Figure 7 we see that adding actions even at the states recommended by ELI simply worsens the quality of a random policy. Yet because the simulated human has a small probability of returning something high-value, the agent is able to leverage this to achieve excellent performance in this setting, especially when using ELI.

6 Conclusion

In this paper we propose a new framework for human-in-the-loop reinforcement learning, where an automated agent leverages human intuition to learn effectively in vast action spaces. We identify a key challenge in this space, namely selecting a state at which to develop a new action, and present a new algorithm, Expected Local Improvement (ELI), which effectively addresses this challenge. Note that although we have focused on this specific challenge, there are many other practical issues that arise when implementing a Human-in-

the-Loop RL system similar to Figure 1 with real humans. For example, what UIs and visualizations cause humans to develop the most useful actions in response to a request? Or, how constrained should the language of actions be? Or, how should we guide humans on *when* (not just where) they should add actions? Future work will study these questions.

Acknowledgements

This work was supported by the NSF BIGDATA grant No. DGE-1546510, the Office of Naval Research (ONR) Young Investigator Award N00014-16-1-2241, ONR grant N00014-12-C-0158, the Bill and Melinda Gates Foundation grant OPP1031488, the Hewlett Foundation grant 2012-8161, Adobe, Google, and Microsoft.

References

Abernethy, J.; Amin, K.; Kearns, M.; and Draief, M. 2013. Large-scale bandit problems and KWIK learning. In *ICML*, 588–596.

Amir, O.; Kamar, E.; Kolobov, A.; and Grosz, B. 2016. Interactive teaching strategies for agent training. In *IJCAI*. AAAI Press.

Asmuth, J.; Li, L.; Littman, M. L.; Nouri, A.; and Wingate, D. 2009. A Bayesian sampling approach to exploration in reinforcement learning. In *UAI*, 19–26. AUAI Press.

Chernova, S., and Veloso, M. 2009. Interactive policy learning through confidence-based autonomy. *Journal of Artificial Intelligence Research* 34(1):1.

Clouse, J. A. 1996. An introspection approach to querying a trainer. Technical report, University of Massachusetts.

Doshi, F.; Pineau, J.; and Roy, N. 2008. Reinforcement learning with limited reinforcement: Using bayes risk for active learning in pomdps. In *Proceedings of the 25th international conference on Machine learning*, 256–263. ACM.

Dulac-Arnold, G.; Evans, R.; Sunehag, P.; and Coppin, B. 2015. Reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*.

Gil, A.; Stern, H.; and Edan, Y. 2009. A cognitive robot collaborative reinforcement learning algorithm. *World Academy of Science, Engineering and Technology*.

Griffith, S.; Subramanian, K.; Scholz, J.; Isbell, C.; and Thomaz, A. L. 2013. Policy shaping: Integrating human feedback with reinforcement learning. In *Advances in Neural Information Processing Systems*, 2625–2633.

Grollman, D. H., and Jenkins, O. C. 2007. Dogged learning for robots. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2483–2488. IEEE.

Jaksch, T.; Ortner, R.; and Auer, P. 2010. Near-optimal regret bounds for reinforcement learning. *JMLR* 11(Apr):1563–1600.

Judah, K.; Fern, A. P.; Dietterich, T. G.; et al. 2014. Active lmitation learning: formal and practical reductions to iid learning. *JMLR* 15(1):3925–3963.

Judah, K.; Fern, A.; and Dietterich, T. 2011. Active imitation learning via state queries. In *Proceedings of the ICML Workshop on Combining Learning Strategies to Reduce Label Cost*. Citeseer.

Leffler, B. R.; Littman, M. L.; and Edmunds, T. 2007. Efficient reinforcement learning with relocatable action models. In *AAAI*, volume 7, 572–577.

Li, J.; Monroe, W.; Ritter, A.; Galley, M.; Gao, J.; and Jurafsky, D. 2016. Deep reinforcement learning for dialogue generation. In *EMNLP*.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *ICLR*.

Mandel, T.; Liu, Y.-E.; Brunskill, E.; and Popović, Z. 2016. Efficient bayesian clustering for reinforcement learning. In *IJCAI*. AAAI Press.

May, B. C.; Korda, N.; Lee, A.; and Leslie, D. S. 2012. Optimistic bayesian sampling in contextual-bandit problems. *JMLR* 13(Jun):2069–2106.

Osband, I.; Russo, D.; and Van Roy, B. 2013. (More) efficient reinforcement learning via posterior sampling. In *NIPS*, 3003–3011.

Russell, S., and Norvig, P. 1994. *Artificial Intelligence A Modern Approach*. Englewood Cliffs, NJ: Prentice Hall.

Sallans, B., and Hinton, G. E. 2004. Reinforcement learning with factored states and actions. *JMLR* 5(Aug):1063–1088.

Silver, D.; Bagnell, J. A.; and Stentz, A. 2012. Active learning from demonstration for robust autonomous navigation. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, 200–207. IEEE.

Strehl, A. L., and Littman, M. L. 2004. An empirical evaluation of interval estimation for Markov decision processes. In *ICTAI 2004*, 128–135. IEEE.

Strehl, A. L., and Littman, M. L. 2008. An analysis of model-based interval estimation for Markov decision processes. *Journal of Computer and System Sciences* 74(8):1309–1331.

Strens, M. 2000. A Bayesian framework for reinforcement learning. In *ICML*, 943–950.

Thomaz, A. L., and Breazeal, C. 2006. Reinforcement learning with human teachers: Evidence of feedback and guidance with implications for learning performance. In *AAAI*, volume 6, 1000–1005.

Thompson, W. R. 1933. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika* 285–294.

Wang, T.; Lizotte, D.; Bowling, M.; and Schuurmans, D. 2005. Bayesian sparse sampling for on-line reward optimization. In *ICML*, 956–963. ACM.

Whittle, P. 1981. Arm-acquiring bandits. *The Annals of Probability* 9(2):284–292.

Williams, J. J.; Kim, J.; Rafferty, A.; Maldonado, S.; Gajos, K. Z.; Lasecki, W. S.; and Heffernan, N. 2016. Axis: Generating explanations at scale with learnersourcing and machine learning. In *Learning@ Scale*, 379–388. ACM.