Cognitive Mechanisms for Reinforcement Learning Refinement

Krueger, Paul
pmk@berkeley.edu, #26969749

Daniels, Dylan
dylandaniels@berkeley.edu, #29655144

May 4, 2016

Abstract

blah blah blah we need to put stuff here.

Introduction

Methods

To explore theories about reinforcement learning in cognition, we constructed a simple two-dimensional maze game whereby an agent was tasked with learning how to reach a terminal goal state from a start state. The terminal goal state confers the agent a reward of 1 point, and resets the agent's location to the starting point (Figure something of maze). We study two different mazes: a *sparse* maze and a *dense* maze. The sparse maze is easier for the agent to solve, but it is also easier for the agent to get stuck in a suboptimal path. The dense maze, on the other hand, has only one clear path to the solution; as a result, we expect heuristic-based pseudorewards to often fail to find the optimal path.

All of our methods are based on a common model-free reinforcement learning paradigm known as Q-learning [Sutton and Barto, 1998], which learns an optimal policy π^* over time. Let $\mathcal{S}$ be the set of states and let $\mathcal{A}$ be the set of actions. For each $(s, a) \in \mathcal{S} \times \mathcal{A}$, a value $Q(s, a)$ is learned via the following algorithm. Initially all $Q(s, a)$ are zero. At each state s, with probability $1 - \epsilon$, the agent chooses the action $a \in \mathcal{A}$ with the highest value $Q(s, a)$. With probability ϵ it chooses an action uniformly at random (ϵ is a hyperparameter that calibrates the explore-exploit tradeoff). Then, after completing the selected action a, the agent moves to s' and updates Q by

$$Q(s, a) \leftarrow Q(s, a) + \alpha(R_(s, s') + \gamma \max_{a'} Q(s', a') - Q(s, a)) \tag{1}$$

where α is the learning rate, $R_(s, s')$ is the reward received at state s', and γ is the discount factor. Q-learning will converge with probability one to the optimal policy.

DYNA Planning

Using the DYNA framework described in [Sutton and Barto, 1998], we can improve upon the naive Q-learning algorithm by recalling a random set of past moves after each step. From a cognitive science perspective, this type of optimization is interesting because it is as if we are replaying past memories. In other words, is it cognitively advantageous to think about past experiences, and as a result, learn more from them?

Planning is implemented by re-updating p Q-values randomly at the end of each step using Equation 1. Only the most recent action for each state is updated, so DYNA planning favors recency.

Pseudorewards

Pseudorewards are an intelligible way of conferring extra information to an agent about the reward landscape. Essentially, a small reward is given to the Q-learner whenever they take an action that helps the agent move towards the goal. Pseudorewards are defined by *shaping functions* F. Instead of the agent receiving actual reward $R(s, s')$ when moving from state $s \to s'$, the agent receives an augmented reward $R'(s, s')$ where

$$R'(s, s') = R(s, s') + F(s, s') \tag{2}$$

In [Ng et al., 1999], conditions for which the optimal policy π^* remains invariant under a *shaping function* are developed. If the shaping function does not possess this invariance policy, it is possible that Q-learning will converge to a suboptimal solution. The simplest example of an invariant shaping function uses the difference in optimal values between the agent's current state and next state:

$$F(s, s') = \gamma V_{\pi^*}(s') - V_{\pi^*}(s) \tag{3}$$

$$V_{\pi^*}(s) = \max_a R(s, s') + \gamma V_{\pi^*}(s') \tag{4}$$

We call this method the *optimal policy pseudoreward*–it encourages the agent to always move down the optimal path from its current state. If $\epsilon = 0$, the agent would move directly to the goal along the shortest path.

While the *optimal policy pseudoreward* performs well in practice, it's a bit unrealistic for the agent to have such a complete information set in most applications. To compute the optimal policy, the agent must solve a linear program and have full information about states, actions, transitions, and goal states. A more realistic set of pseudorewards can be derived by approximating the distance to the goal. Intuitively, this corresponds to the agent generally knowing which direction to move in. We call this the *Manhattan pseudoreward*.

For our maze environment, we use a modified Manhattan distance metric to implement distance-based pseudorewards. We define the Manhattan distance metric as $\Phi(s) = \gamma^T$, where T is the Manhattan distance, to form the pseudoreward

$$F(s, s') = \gamma\Phi(s') - \Phi(s) \tag{5}$$

which fulfills the conditions in [Ng et al., 1999] to be an invariant reward transformation. A comparison the pseudoreward landscape for each of the types of the pseudorewards considered is shown in Figure 1.

In addition, we also test the sensitivity of both pseudoreward methods to additive white noise. For each pseudoreward $F(s, s')$ conferred, let

$$\tilde{F}(s, s') = F(s, s') + e_t \tag{6}$$

where $e_t \sim N(0, \sigma^2)$. The larger the value of σ, the more noisy the pseudoreward $\tilde{F}$. In general pseudorewards $\tilde{F}$ will not lead to invariant policies.

Experiments

To analyze the performance of our agent in each of our two maze environments: *sparse* and *dense*, we ran 100 simulations of reinforcement learning for each condition. In each simulation, we allowed the agent to learn for 50 episodes; the agent automatically advanced to the next episode without reward if a maximum of 2000 steps was reached. For all of our experiments, we set the learning rate $\alpha = 0.1$, the exploratory probability $\epsilon = 0.25$, and the discount factor $\gamma = 0.95$.

As a first pass, we compare the DYNA architecture with 10 replays with the optimal policy pseudoreward architecture (Figure 2). All methods converge to the optimal number of steps: 20 for sparse, 24 for dense. DYNA converges more quickly than regular Q-learning, validating the hypothesis that replaying memories speeds up the learning process. We also see that the optimal policy pseudoreward converges nearly instantly to the optimal value, which is expected because the agent is incentivized to follow the optimal path from the start. When the optimal policy pseudoreward is combined with DYNA for replaying memories, it converges just as fast.

Next, in Figure 3 we analyze the performance of DYNA by varying the number of moves replayed at the conclusion of each episode. As the number of replays increases, the convergence rate of Q learning increases. It is interesting to note that as a function of episodes, the dense maze appears to have a concave shape while the sparse maze has a convex shape. This means that the majority of learning happens later for the dense maze than for the sparse maze. This is probably due to the fact that it takes longer for the agent to find the optimal path in the dense maze, but once it finds it, learning happens rapidly after.

We finally compare the performance of our two pseudoreward strategies. As can be seen in Figure something,

Discussion

References

References

[Ng et al., 1999] Ng, A. Y., Harada, D., and Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, volume 99, pages 278–287.

[Sutton and Barto, 1998] Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT press.

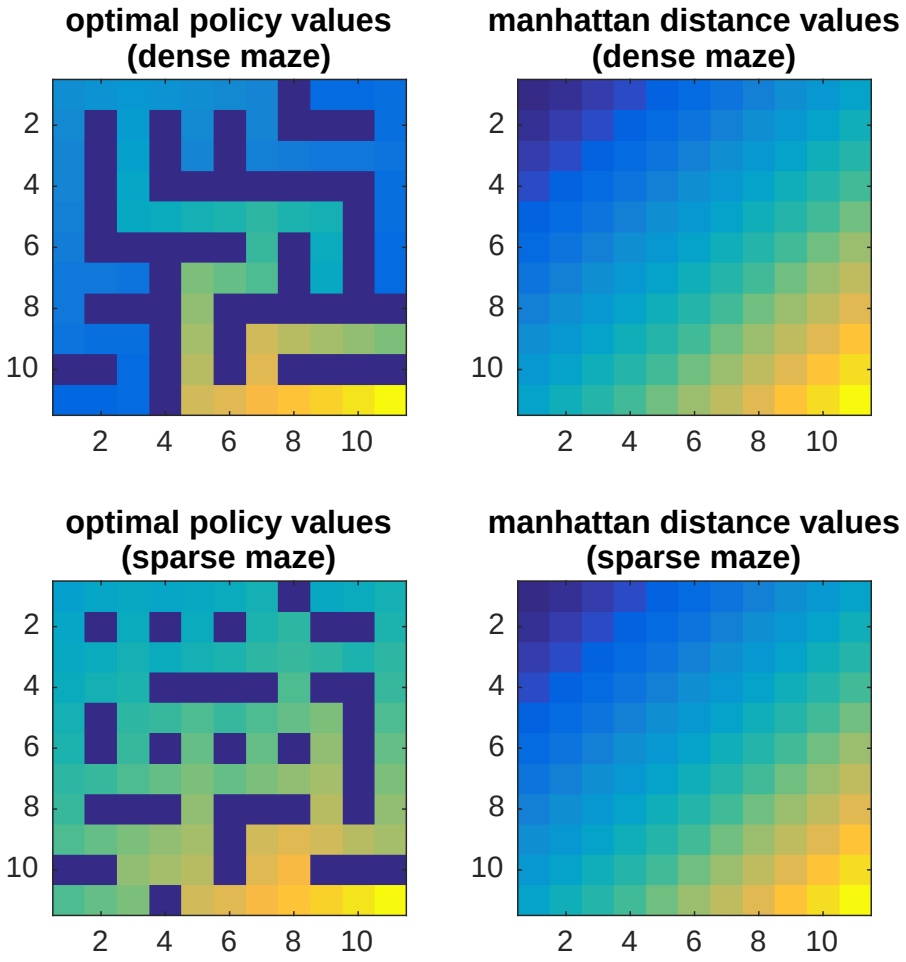

Figure 1: The landscape of pseudorewards for each maze and each pseudoreward type. Pseudorewards are concocted so that the agent is incentivized to move towards the goal state in the lower rightmost square.

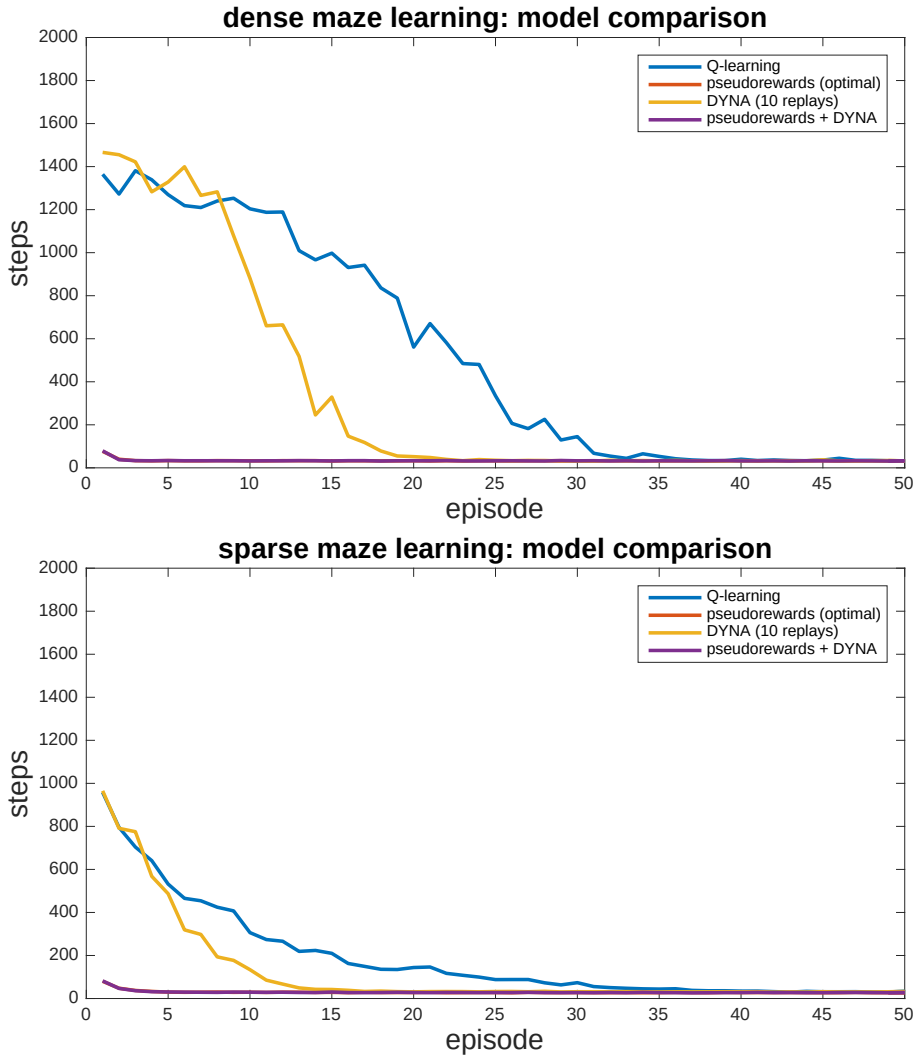

Figure 2: The mean number of steps taken for each episode are plotted above for Q-learning and 3 variants. The mean is taken over 100 simulations of 50 episodes.

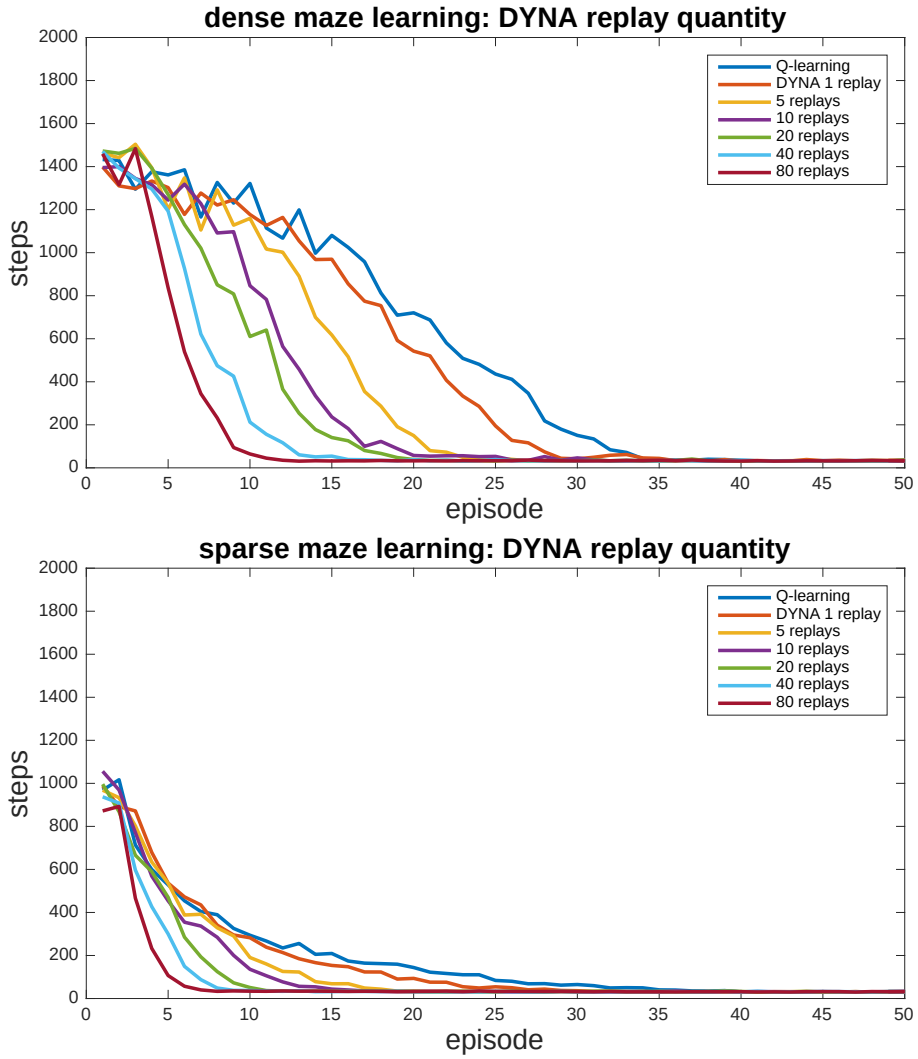

Figure 3: The mean number of steps taken for 100 simulations of DYNA learning. The convergence rate increases with the number of replays.