Cognitive Mechanisms for Reinforcement Learning Refinement

Paul Krueger and Dylan Daniels

May 4, 2016

Abstract

blah blah

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 call each goal-finding session an episode. 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 (some reference here? Sutton?), 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 step in 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 {some reference}, we can improve upon the naive Q-learning algorithm by recalling certain moves made. 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 learn 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 episode using Equation 1.

Pseudorewards

Pseudorewards are an intelligible way of conferring extra information to the 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 $R(s)$ at step s, the agent receives an augmented reward $R'(s)$ where

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

In {that Andrew Ng paper}, 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 the 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 {Andrew Ng paper} to be an invariant reward transformation.

In addition, we also test how sensitive both of the above pseudorewards are 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 $\tilde{F}$.

Experiments

Discussion

References