Machine Learning Topics **Summer 2017**

Topic 2: Reinforcement Learning

Author: Andy Lee

2.1 Introduction

Reinforcement learning is a framework used to model the interactions between an agent and an environment, where the interactions play out across discrete time steps t. In this set of notes, we provide an overview of the reinforcement learning problem and state the equations involved in this framework.

2.1.1 Notation

We will consider the following model.

- Let $s \in S$ be some state, where S defines some state space.
- Let $a \in A$ be some action, where A defines some action space.
- Let $\pi(a|s)$ be a conditional probability density over the action space. π is referred to as a *policy.*
- Let $p(s'|s, a)$ denote the transition probability of going from $s \to s'$ after taking action a.
- Let $R : S \times A \to \mathbb{R}$ be a reward function that maps state, action pairs to a scalar value.

2.1.2 Motivation

In the reinforcement learning setting, our agent is in some state s_t at some time step t. The agent then selects some action $a_t \sim \pi$ according to its policy. Upon performing this action, the agent receives a reward $r_t = R(s_t, a_t)$. The goal in reinforcement learning is to learn a policy π such that the agent maximizes its cummulative expected reward across time. In short, we are interested in the following question. How do we update the policy of the agent as we get more information about the rewards that we receive? As such, reinforcement learning algorithms prescribe update rules to an agent's policy after observing the rewards generated over time.

One should note that the reinforcement learning framework is flexible and can be used to model many different types of applications. In robotics, reinforcement learning can be used to do optimal planning in uncertain environments. In AI, reinforcement learning is used to learn the optimal sequence of controls in classical games like atari and tetris. All of these applications fall under a common theme of trying to produce some desired behavior according to the rewards given by the environment.

There is one final point that we would like to make and that is the distinction between reinforcement learning and inverse reinforcement learning. In reinforcement learning, the reward function is specified and the goal is to learn the *policy* that maximizes cummulative expected reward. In inverse reinforcement learning, we have a set of behaviors that we want to imitate and the goal is to learn the *reward function* that allows a fixed policy to match those behaviors.

2.2 Reinforcement Learning Problem

This section sets up the mathematics behind the reinforcement learning problem.

2.2.1 Markov Decision Process

We begin by mentioning that a reinforcement learning task that satisfies the Markov property is called a **Markov Decision Process** (MDP). Historically, reinforcement learning has been synonomous with MDPs as most RL problems are framed in the MDP setting. In this set of notes, we will deal with RL problems specifically in the MDP settting.

2.2.2 Value Functions

The reinforcement learning problem relies on 2 major equations defined below: state-value function and action-value function. These functions encode notions of expected accumulated reward if an agent follows a fixed policy π. We will see that our RL algorithms will try to esimate these functions based on the rewards that it receives from environment. Note we use the notation $r_t = R(s_t)$ to denote the reward gained when the agent is at state s_t.

State-Value Function
The state-value function is denoted by $V_\pi : S \to \mathbb{R}$. This function defines the cummulative expected discounted reward that an agent receives if it starts at state s and follows policy π onwards.

$$\begin{aligned} V_\pi(s) &= \mathop{\mathbb{E}}_{a\sim\pi}\left[R(s_{t+1}) + \gamma R(s_{t+2}) + \gamma^2 R(s_{t+3}) + \cdots \middle| s_t = s\right] \\ &= \mathop{\mathbb{E}}_{a\sim\pi}\left[\sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k} \middle| s_t = s\right] \end{aligned}$$

Action-Value Function
The action-value function is denoted by $Q_\pi : S \times A \to \mathbb{R}$. This function is nearly identical the state-value with one small difference. It defines the cummulative expected discounted reward that an agent receives if it starts at state s *and* takes an immediate action a, subsequently following policy π onwards.

$$\begin{aligned} Q_\pi(s,a) &= \mathop{\mathbb{E}}_{a\sim\pi}\left[R(s_{t+1}) + \gamma R(s_{t+2}) + \gamma^2 R(s_{t+3}) + \cdots \middle| s_t = s, a_t = a\right] \\ &= \mathop{\mathbb{E}}_{a\sim\pi}\left[\sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k} \middle| s_t = s, a_t = a\right] \end{aligned}$$

2.2.3 Bellman Equations

We can note a fundamental recursive property of the state-value function that will come up again and again in reinforcement learning and dynamic programming.

Let $p_\pi(s'|s)$ denote the probability of transition from state s to state s' following policy π.

$$p_\pi(s') = \sum_a \pi(a|s) p(s'|a, s)$$

We can derive the recursive equation, known as the **Bellman equation**.

$$\begin{aligned}V_\pi(s) &= \mathop{\mathbb{E}}_{a\sim\pi}\left[\sum_{k=1}^{\infty}\gamma^{k-1}r_{t+k}\middle|s_t = s\right]\\ &= \mathop{\mathbb{E}}_{a\sim\pi}\left[r_{t+1} + \sum_{k=2}^{\infty}\gamma^{k-1}r_{t+k}\middle|s_t = s\right]\end{aligned}$$

To write out the expectation explicitly, we have to consider all the states s' to which our current state s can transition. Specifically, the reward for transition to state s' is given to be $R(s') + \gamma V_\pi(s')$ and associated probability is $p_\pi(s'|s)$.

$$\begin{aligned}V_\pi &= \sum_{s'} p_\pi(s'|s)(R(s') + \gamma V_\pi(s'))\\ &= \sum_{s'}\left[\left(\sum_a \pi(a|s)p(s'|a,s)\right)\left(R(s') + \gamma V_\pi(s')\right)\right] \text{ (substitute } p_\pi(s'|s))\\ &= \sum_{s'}\sum_a \pi(a|s)p(s'|a,s)(R(s') + \gamma V_\pi(s'))\\ &= \sum_a \pi(a|s)\sum_{s'} p(s'|a,s)(R(s') + \gamma V_\pi(s'))\end{aligned}$$

Therefore, Bellman equation is compactly written as

$$V_\pi(s) = \sum_a \pi(a|s)\sum_{s'} p(s'|a,s)(R(s') + \gamma V_\pi(s')) \tag{2.1}$$

2.2.4 Optimal Value Functions

Often, we will be interested in the following optimal value functions, where the max operators are taken over all $s \in S$ for the state-value function and all $(s,a) \in S \times A$ for the action-value function.

$$\begin{aligned}V^*(s) &= \max_\pi V_\pi(s)\\ Q^*(s,a) &= \max_\pi Q_\pi(s,a)\end{aligned}$$

In addition, we can expand the equations into a more expressive form as follows.

$$\begin{aligned}V^*(s) &= \max_a Q^*(s,a)\\ &= \max_a \sum_{s'} p(s'|s,a)(R(s') + \gamma V^*(s'))\end{aligned}$$

$$\begin{aligned}Q^*(s,a) &= \mathbb{E}[r_{t+1} + \gamma V^*(S_{t+1})|S_t = s, A_t = a]\\ &= \sum_{s'}\left(p(s'|s,a)[R(s') + \gamma \max_{a'} Q^*(s',a')\right)\end{aligned}$$

References

[1] Richard S. Sutton and Andrew G. Barto. *Reinforcement learning: an introduction.* The MIT Press, 2012.