# Reinforcement Learning Notes

## Multiarmed Bandits

The most important feature distinguishing RL from other types of learning is that it uses training information that *evaluates* the actions taken rather than *instructs* by giving correct actions. This is what creates the need for active exploration, for an explicit search for good behavior. Purely evaluative feedback indicates how good the action taken was, but not whether it was the best or the worst action possible. Purely instructive feedback, on the other hand, indicates the correct action to take, independently of the action actually taken. This kind of feedback is the basis of supervised learning, which includes large parts of pattern classification, artificial neural networks, and system identification. In their pure forms, these two kinds of feedback are quite distinct: evaluative feedback depends entirely on the action taken, whereas instructive feedback is independent of the action taken.

In this Section we study the evalualtive aspect of RL in a simplified setting, one that does not involve learning to act in more than one situation. This nonassociative setting is the one in which most prior work involving evaluative feedback has been done and it avoids much of the complexity of the full reinforcement learning problem. Studing this case allows us to see how evaluative feedback differs from, and yet can be combined with, instructive feedback.

The particular nonassociative, evaluative feedback problem that we explore is a simple version of the -armed bandit problem.

## A -armed Bandit Problem

Consider the following learning problem. You are faced repeatedly with a choice among different options or actions. After each choice you receive a numerical reward chosen from a stationary probability distribution that depends on the action you selected. Your objective is to maximize the expected total reward over some time period, for example over 1,000 action selections or *time steps*.

This is the original form of the k-armed bandit problem, so named by analogy to a slot machine or “one-armed bandit”, except that it has levers instead of one. Each action selection is like a play of one of the slot machine’s levers, and the rewards are the payoffs for hitting the jackpot. Through repeated action selections you are to maximize your winnings by concentrating your actions on the best levers.

In out k-armed bandit problem each of the k actions has an expected or mean reward given that that action is selected; we will call this the *value* of that action. We denote the action selected on time step as , and the corresponding reward as . The value then of an arbitrary action , denoted , is the expected reward given that is selected:

(1)

If you knew the value of each action, then it would be trivial to solve the k-armed bandit problem:

you would always select the action with the highest value. We assume that you do not know the action values with certainty, although you may have estimates. We denote the estimated value of action at time step as . We would like to be close to .

If we maintain estimates of the action values, then at any time step there is at least one action whose estimated value is greatest. We call these the ***greedy*** actions. When the agent selects one of these actions, we say that it is ***exploiting*** its current knowledge of the values of the actions. If instead, the actor selects one of the nongreedy actions, then we say that the agent is exploring, because the selection enables it to improve its estimate of the nongreedy action’s value. Exploitation is the right thing to do to maximize the expected reward on the one step, but exploration may produce the greater total reward in the long run. For example, suppose a greedy action’s value is known with certainty, while several other actions are estimated to be nearly as good but with substantial uncertainty. The uncertainty is such that at least one of these other actions probably is actually better than the greedy action, but we don’t know which one. If we have many time steps ahead on which to make action selections, then it may be better to explore the nongreedy actions and discover which of them are better than the greedy action. Reward is lower in the short run, during exploration, but higher in the long run because after the agent has discovered better actions, those can be exploited many times. Because it is not possible both to explore and to exploit with any single action selection, one refers to the “conflict” between exploration and exploitation.

In any specific case, whether it is better to explore, or exploit depends in a complex way on the precise values of the estimates, uncertainties and the number of remaining steps. There are many sophisticated methods for balancing exploration and exploitation for particular mathematical formulations of the -armed bandit and related problems.

## Action-value Methods

We begin by looking more closely at methods for estimating the values of actions and for using the estimates to make action selection decisions, which we collectively call ***action-value methods***. Recall that the true value of an action is the mean reward when that action is selected. One natural way to estimate this is by averaging the rewards actually received:

(2)

where denotes the random variable that is 1 if *predicate* is true and 0 if it is not. If the denominator is zero, then we instead define as some default value, such as 0. As the denominator goes to infinity, by the law of large numbers, converges to . We call this the *sample-average* method for estimating action values because each estimate is an average of the sample of relevant rewards. Of course, this is just one way to estimate action values, and not necessarily the best one. Nevertheless, for now let us stay with this simple estimation method and turn to the question of how the estimates might be used to select actions.

The simplest action selection rule is to select one of the actions with the highest estimated value, that is, one of the greedy actions. If there is more than one greedy action, then a selection is made among them in some arbitrary way, perhaps randomly. We write this greedy action selection method as:

(3)

Where denotes the action for which the expression that follows is maximized (with ties broken arbitrarily). Greedy action selection always exploits current knowledge to maximize immediate reward; it spends no time at all sampling apparently inferior actions to see if they might really be better. A simple alternative is to behave greedily most of the time, but every once in a while, say with small probability , instead select randomly from among all actions with equal probability, independently of the action-value estimates. We call methods using this near-greedy selection rule **-greedy methods**.

An advantage of these methods is that, in the limit as the number of steps increases, every action will be sampled an infinite number of times, thus ensuring that all the converge to their respective . This of course implies that the probability of selecting the optimal action converges to greater than , that is, to near certainty. These are just asymptotic guarantees and say little about the practical effectiveness of the methods.

## The 10-armed Testbed

To roughly assess the relative effectiveness of the greedy and the -greedy action-value methods, we compare them numerically on a suite of test problems. This is a set of 2000 randomly generated k-armed bandit problems with . For each bandit problem such the one shown on the Figure below, the action values, were selected according to a normal (Gaussian) distribution with mean 9 and variance 1.

A graph of a function

Description automatically generated

Figure: An example bandit problem from the 10-armed testbed. The true value of each of the ten actions was selected according to a normal distribution with mean zero and unit variance, and then the actual rewards were selected according to a mean , unit-variance normal distribution, as suggested by these gray distributions.

//TODO: finish the multi-armed bandits discussion