Double Q-learning

Gautam Bhattacharya
Comp-762 Report

1 Introduction

Q-learning is a popular algorithm for off-policy TD control. The Q-learning update rule is given by:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \quad (1)$$

One of Q-learning's shortcomings is that it suffers from a bias due to the use of the maximum action value as an approximation for the maximum expected action value. This leads to poor performance in certain stochastic environments where Q-learning overestimates the action values. Double Q-learning provides a solution to the bias problem by making use of two estimators. While also providing a biased estimate of maximum expected action value, it underestimates the action values as opposed to Q-learning.

An intuitive example of maximization bias is provided in the Sutton and Barto textbook. Consider a single state **s** where there are many actions a whose true values, q(s,a), are all zero but whose estimated values, Q(s, a), are uncertain and thus distributed some above and some below zero. The maximum of the true values is zero, but the maximum of the estimates is positive. This type of situation is encountered in the game of Roulette, where there is one state and several actions (corresponding to the numbers that can be bet on). When Q-learning is used to estimate values for these different actions, it grossly overestimates them.

2 Analysis

Given a set of random variables $X = X_1, ..., X_M$, the task of approximating $\max_i E\{X_i\}$ can be broken into two parts. The first step involves estimating $E\{X_i\}$. This is typically done by sampling each variable and an unbiased estimate of the expected value is obtained by calculating sample averages such that $E\{X_i\} = E\{\mu_i\}$. Where μ_i is an estimator of $E\{X_i\}$. Thus the simplest approximation to the maximum would be:

$$\max_i E\{X_i\} = \max_i E\{\mu_i\} \quad (2)$$

As the name suggests, this approach makes use of two sets of approximators, μ^A and μ^B for the random variable X. The key idea here is that an estimator a^* that maximizes μ^A can be used as an estimate for $\max_i E_i^B$, such that:

$$\max_i E\{X_i\} = \max_i \mu_i^B \approx \mu_{a^*}^B \quad (3)$$

3 Double Q-Learning

The double Q-learning algorithm applies this idea of two estimators to the Q-learning paradigm. Specifically, two separate estimates of the Q values are maintained, Q_A and Q_B. Like standard Q-learning, an action is selected from Q_A and Q_B (e.g. epsilon-greedy policy) however only one of the maintained Q-estimates is updated at a time. This choice is made randomly. The update itself is as follows:

if update Q^A:
$a^* = argmax_a Q^A(\acute{s}, a)$
$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q^B(\acute{s}, a^*) - Q(s,a)]$

if update Q^b:
$b^* = argmax_a Q^A(\acute{s}, a)$
$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma Q^A(\acute{s}, b^*) - Q(s,a)]$

Similar to the double estimator, action a^* may not be the action that maximizes the expected Q function $\max_a E\{QA(\acute{s}, a)\}$. In general $E\{Q^B(\acute{s}, a^*)\} ? \max_a E\{Q^A(\acute{s}, a^*)\}$, and underestimations of the action values can occur.

4 Experiments

In the Double Q-learning paper the author demonstrates the performance of the Q-learning and double Q-learning on the game of Roulette. Q-learning overestimates the action values, with all non-terminating (i.e. betting) actions being valued at approx. 22$, on the other hand, double Q-learning does not suffer from underestimation in this setting.

We attempted to duplicate this experiment in the OpenAI gym environment. The Roulette environment is quite similar to the one in the Double Q-learning paper. The game is 0-to-36 Roulette, where at each spin the agent bets on a number. The agent receives a positive reward if only if its bet matches the number spun. There agent can also walk away from the game ending the episode.

Figure. 1 was generated by computing the mean action value over all actions for 10,000 trials. In each trial, all 38 actions are updated synchronously. We see a similar pattern between the action values estimated by Q-learning and double Q-learning as reported in the paper. Q-learning overestimates the action values at approx. 23$, wheres as double Q-learning estimates the action vales at approx. $0.

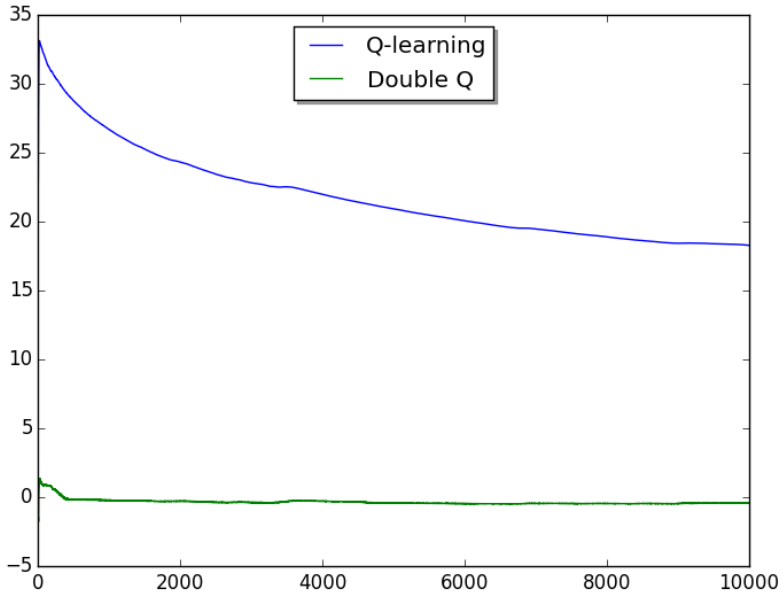

Figure 1: Q-learning Vs Double Q-learning