Comparing Methods Of Action-Value Estimation In The Context Of A Non-Stationary k-armed Bandit Problem

Holt Spalding

, September 19, 2018

---

1. Introduction

In this short demonstration, I conduct an experiment in order to compare two methods of action-value estimation on a simple *nonassociative* reinforcement learning task. The task in question is the so-called "*k-armed bandit*" problem, the simplest form of which is described by Sutton & Barto[1] as follows...

> "You are faced repeatedly with a choice among k 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 1000 action selections, or time steps."

This task is commonly known as the k-armed bandit problem because, when framed in more concrete terms, it's (somewhat) analogous to pulling the arms of k different slot machines to predict which will win you the most money. If the reward returned by each slot machine after one arm-pull is sampled from some probability distribution individual to that machine, then the task is essentially to determine which slot machine will provide the largest cumulative reward after t arm-pulls given our estimation of each machine's internal reward distribution. The task is nonassociative because the visible state of each machine/bandit plays no role in the action selection (in fact, each bandit has no visible state). In other words, there is nothing visible to us that we could associate with higher rewards that might help us in the

task. If we knew, for example, that a particular brand of machine tended to yield higher rewards, then the problem would be considered associative.

In the simplified form of the problem described above, each bandit returns a reward that has been randomly sampled from a stationary probability distribution. However, the problem may take other forms. Let say these probability distributions were in fact non-stationary. Would all methods of action-value estimation that were acceptable for the stationary case be acceptable in the non-stationary case? The answer is of course *no*. In this experiment I have compared the performance (in terms of reward maximization) of the "sample-average" and "exponential recency-weighted average" methods of action-value estimation for the non-stationary "k-armed bandit" problem.

2. Action-value Methods

2.1. Sample-Average Methods

In the context of the k-armed bandit problem, the goal of our agent is to anticipate which bandit will return the highest cumulative reward. We let $q_*(a) = \mathbb{E}[R_t|A_t = a]$ denote the expected reward R returned after selecting a particular action a at time step t (an action in this context is the selection of a particular bandit). Since the reward distribution of each bandit is hidden to us, we can only estimate this value. We therefore let $Q_t(a)$ represent the estimated reward of a particular action a at time step t. Different action-value methods model Q differently, and each method can be appropriate depending on the task, but all action-value methods seek to estimate $Q_t(a)$ so it's as close to $q_*(a)$ as possible. In the sample-average method, $Q_t(a)$ is calculated as follows:

$$Q_t(a) = \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

where t represents the current time step ($t = 1$ at trial start), R_i represents the reward received at time step i, A_i represents the action taken at time i, and $\mathbb{1}_{A_x} = a$ denotes a random variable that is either 1 if $A_x = a$ is true and 0 if $A_x = a$ is false. This is known as the sample-average method because the expected reward of each action is estimated in terms of the average reward received from performing a particular action.

When running experiments for a large number of time steps, recalculating $Q_t(a)$ (where a is the most recent action performed) at every time step can become very costly. Instead of calculating the above formula at every time step, we can just update $Q_t(a)$ incrementally and compute a small constant time computation for Q_t at each time step. If we let $Q_n = \frac{R_1+R_2+\cdots+R_{n-1}}{n-1}$ represent the sample-average expected reward of a given action after n selections of that action, then we know ...

$$Q_{n+1} = \frac{1}{n}\sum_{i=1}^{n} R_i$$

$$= \frac{1}{n}(R_n + \sum_{i=1}^{n-1} R_i) = \frac{1}{n}(R_n + \sum_{i=1}^{n-1} R_i)$$

$$= \frac{1}{n}(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i)$$

$$= \frac{1}{n}(R_n + (n-1)Q_n$$

$$= \frac{1}{n}(R_n + nQ_n - Q_n$$

$$= Q_n + \frac{1}{n}[R_n - Q_n]$$

Therefore, after choosing some action x, we need only to update it's expected reward Q_n (which we should already have a record of), by calculating $= Q_n + \frac{1}{n}[R_n - Q_n]$.

2.2. Exponential Recency-Weighted Average

The sample-average method of action-value estimation is appropriate in the context of a stationary bandit problem. However, when dealing with a non-stationary problem, there are other superior methods. One such method is the exponential recency-weighted average method. In this method, the expected reward of a given action after $n + 1$ selections is calculated as follows...

$$Q_{n+1} = Q_n + \alpha[R_n - Q_n]$$

where $\alpha \in (0, 1]$ is a constant hyperparamter set before runtime. This decomposes thusly...

$$Q_{n+1} = Q_n + \alpha[R_n - Q_n]$$

$$= \alpha R_n + (1-\alpha)Q_n$$

$$= \alpha R_n + (1-\alpha)[\alpha R_{n-1} + (1-\alpha)Q_{n-1}]$$

$$= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1}$$

$$= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} + \ldots$$
$$\ldots \qquad (1-\alpha)^{n-1}\alpha R_1 + (1-\alpha)^n Q_1$$

$$= (1-\alpha)^n Q_1 + \textstyle\sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i$$

As you can see, this action-value estimation method is a weighted average of of all rewards recieved from a given action, where the most recent rewards (R_n geing the most recent) are weighted more heavily than the oldest rewards. In the context of a non-stationary problem, this is obviously a superior method of action-value estimation since the most recent rewards should be most representative of the rewards that will be recieved from a given action in the near future.

3. The Experiment

3.1. Method

For this demonstration I created a short python script to simulate a 10-armed bandit testbed. All bandits start out with the same reward distribution (normal distribution centered around 0), and then at each time step their distribution's mean, or $q_*(a)$, is incremented by some random number sampled from a normal distribution with mean 0 and standard deviation 0.01. The sample-average and weighted exponential-recency average methods do not prescribe which action to take at any given time step, they only provide the estimated reward of a given action at a particular time step. Therefore, actions were actually selected by means of a simple epsilon-greedy algorithm. This algorithm essentially selects an action at a given timestep by sampling

a Bernoulli distribution in which the action with the highest expected reward has a $(1 - \epsilon)$ probability of being selected and a random action has a probability of ϵ. The probabilistic aspect to this greedy algorithm promotes exploration and prevents our simple agent from determining a suboptimal action to be optimal. In the case of this experiment $\epsilon = 0.1$, and $\alpha = 0.1$

3.2. Results

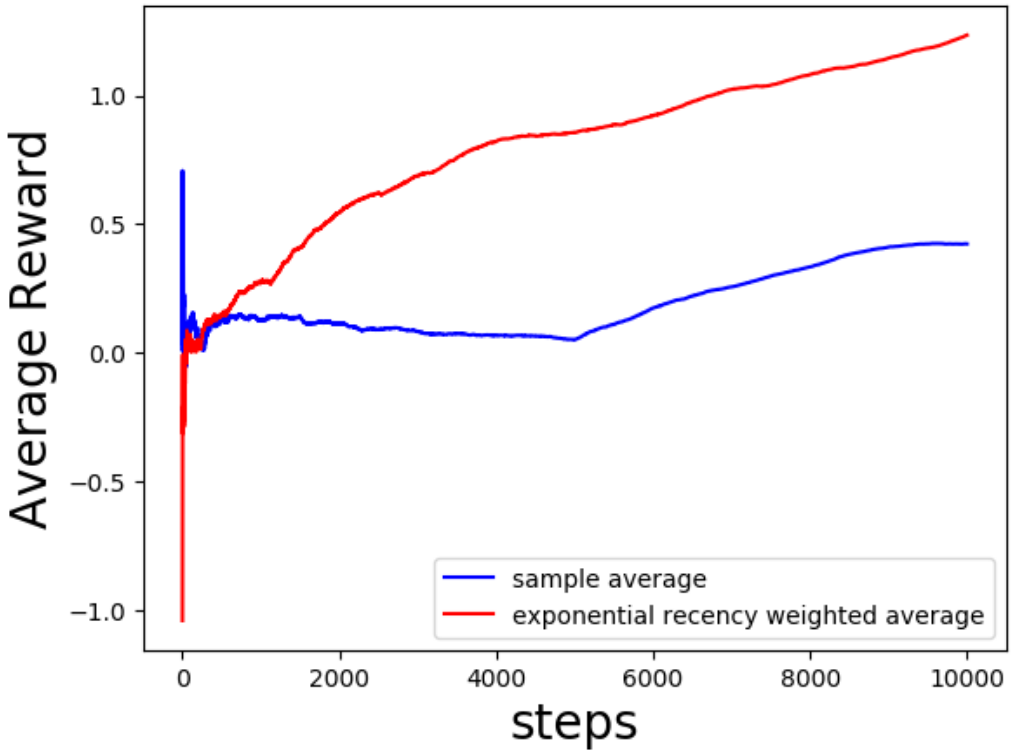

Figure 1: Average overall reward after 10,000 action selections

As expected, our agent which employed the ERWA method outperformed the SA agent rather significantly after only 10,000 time steps.

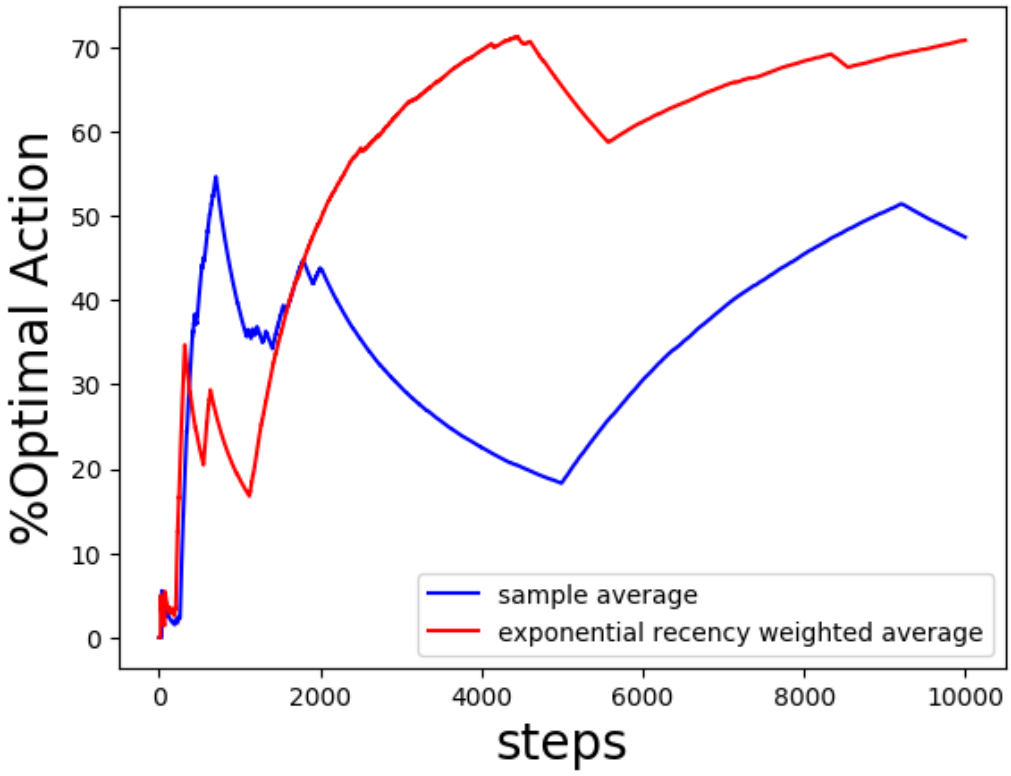

Figure 2: Percentage of action selections that are optimal after 10,000 action selections

4. Brief Discussion

As we've seen, not all action-value estimation methods are suited for all tasks. Even the ERWA method was only somewhat effective because our reward distributions drifted relatively slowly. For non-stationary problems where there is a lot of change in reward distribution across time steps, ERWA is ill-equipped. Even in this experiment there were occasionally trials in which the SA agent would perform as well as the ERWA agent (because of the randomness of the problem). While this experiment was very simple, it was definitely demonstrative of something pretty profound about reinforcement learning agents. I hope to play with more robust action-value estimation methods in the future.

5. References

1. Sutton, R. S., & Barto, A. G. (1998). Reinforcement learning: An introduction. MIT press.