Lecture 2: EE675A Introduction to Reinforcement Learning

11/01/23

Lecturer: Subramanya Peruru *Scribe: Avinash Bhashkar 21205262, Toshith 22105270*

Nomenclature

$\bar{\mu}(a)$ Sample mean of arm a

$\mathcal{A}$ Set of all arms

$\mu(a)$ True mean of arm a

a_t Arm chosen at round t

K Total number of arms

T Total number of rounds

1 Reinforcement Learning (RL)

Reinforcement learning is learning what to do—how to map situations to actions—so as to maximize a numerical reward signal[2]. The learner is not told which actions to take, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may a ect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two characteristics—trial-and-error search and delayed reward—are the two most important distinguishing features of reinforcement learning.

Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. The training is based on the input, the model will return a state and the user will decide to reward or punish the model based on its output. The model keeps continues to learn. The best solution is decided based on the maximum reward.

Reinforcement important terms:

- State
- Actions
- Environment $(S_t, A_t, R_{t+1}, S_{t+1}, ...S_T)$
- Reward
- Return (i.e Cumulative Reward) |
- Policy
  - Deterministic Policy
  - Stochastic Policy
- Bandit (Special Case of RL)

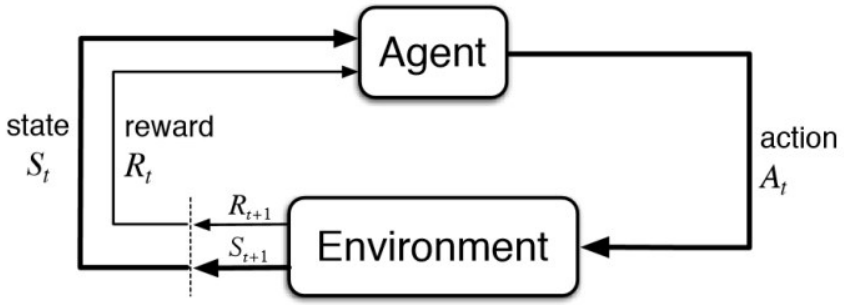

Figure 1: Basic diagram of reinforcement Learning

2 Stochastic Bandits

In the Stochastic Bandits[1], we study the evaluative aspect of reinforcement learning in a simplified setting, one that does not involve learning to act in more than one situation. This non-associative 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. Studying this case enables us to see most clearly how evaluative feedback diers from, and yet can be combined with, instructive feedback.

Objective: Best-arm identification

Stochastic Bandits : K arms, T rounds (both known); reward distribution D_a for each arm a (unknown). In each round $t\epsilon[T]$:

1. Algorithm picks some arm at
2. Reward $r_t[0,1]$ is sampled independently from distribution $D_a, a = a_t$
3. Algorithm collects reward r_t, and observes nothing else.

How do we argue whether an algorithm is doing a good job across different problem instances? The problem is, some problem instances inherently allow higher rewards than others. One standard approach is to compare the algorithm's cumulative reward to the *best-arm benchmark* $\mu^* T$: the expected reward of always playing an optimal arm, which is the best possible total expected reward for a particular problem instance. Formally, we define the following quantity, called *regret* at round T:

$$R(T) = \mu^*.T - \sum_{t=1}^{T} \mu(a_t)$$

2.1 Explore Then Commit

This is a very naive algorithm. The initial segment of rounds is dedicated to the exploration and the remaining rounds to exploitation.

Explore-First Algorithm with parameter N:

1. Exploration phase: Explore each arm N times.
2. Select arm $\hat{a}$ with the highest sample average reward.
3. Exploitation phase: Play arm $\hat{a}$ in all remaining rounds.

2.1.1 Hoeffding inequality(Prerequisite)

To analyze the regret of Explore Then Commit algorithm one must understand Hoeffding inequality constraint. Hoeffding inequality is a mathematical result that provides a bound on the probability that the sum of independent random variables deviates from its expected value by more than a certain amount. Intuitively, it says that as the number of random variables (or "trials") increases, the likelihood of the sum deviating significantly from its expected value becomes smaller. It is often used in statistics and machine learning to understand and analyze the performance of algorithms.

The Hoeffding inequality can be represented mathematically as:

$$\mathbb{P}(|\overline{\mu}(a) - \mu(a)| \geq \epsilon) \leq 2e^{(-2\epsilon^2 \times N)}$$

Where:

- $\mathbb{P}$ is the probability
- $\overline{\mu}(a)$ is the average reward for each action $'a'$ after exploration
- $\mu(a)$ is the true expected rewards
- t is a positive constant
- n is the number of independent random variables.
- $\epsilon = \sqrt{2log(T)/N}$

$$\mathbb{P}(|\overline{\mu}(a) - \mu(a)| \leq \epsilon) \geq 1 - 2/T^4$$

$\mu(a) \in [\overline{\mu}(a) - \epsilon, \overline{\mu}(a) + \epsilon]$ A known interval containing some scalar quantity is called the *confidence interval* for this quantity.

We define the *clean event* to be the event that satisfies the above inequality for all arms simultaneously, and the *bad event* – the complement of the clean event. With this approach, one does not need to worry about probability in the rest of the analysis. Indeed, the probability has been taken care of by defining the clean event and observing that the above inequality holds therein. We do not need to worry about the bad event, either: essentially, because its probability is so tiny.

The inequality states that the probability that the absolute value of the difference between the sum of the random variables and its expected value is greater than or equal to a positive constant t, is less than or equal to e^{-2t^2N}.

In simpler terms, the probability that the sum of the random variables deviates from its expected value by more than t (in either direction) becomes smaller as the number of random variables increases.

For simplicity, let us start with the case of K = 2 arms. Consider the clean event. If we chose the worse arm, it is not so bad because the expected rewards for the two arms would be close.

Let the best arm be a^*, and suppose the algorithm chooses the other arm $a \neq a^*$. This is because its average reward was better than that of a^*: $\bar{\mu}(a) > \bar{\mu}(a^*)$. Since this is a clean event, we have:

$$\mu(a) + \epsilon \geq \bar{\mu}(a) > \bar{\mu}(a*) \geq \mu(a^*) - \epsilon$$

We get

$$\bar{\mu}(a*) - \bar{\mu}(a) \leq 2\epsilon$$

Thus, each round in the exploitation phase contributes at most 2ϵ to regret. Each round in exploration trivially contributes at most 1. We derive an upper bound on the regret, which consists of two parts: for exploration, when each arm is chosen N times, and then for the remaining $T2N$ rounds of exploitation:

$$R(T) \leq N + 2\epsilon(T - 2N) < N + 2\epsilon T$$

Recall that we can select any value for N, as long as it is given to the algorithm in advance. So, we can choose N so as to (approximately) minimize the right-hand side. Since the two summands are, respectively, monotonically increasing and monotonically decreasing in N, we can set N so that they are approximately equal. Therefore,

$$R(T) \leq N + 2\epsilon T$$

$$R(T) \leq N + 2\left(\sqrt{\frac{a \log T}{N}}\right) T$$

$$N = T^{2/3}(\log T)^{1/3}$$

$$R(T) \leq O(T^{2/3}(\log T)^{1/3})$$

$$\mathbb{E}[R(T)] = \mathbb{E}[R(T)|\text{ clean event }] \times \mathbb{P}[\text{ clean event }] + \mathbb{E}[R(T)|\text{ bad event }] \times \mathbb{P}[\text{ bad event }]$$

$$\leq \mathbb{E}[R(T)|\text{ clean event }] + T \times O(T^4)$$

$$\leq O\left((logT)^{1/3} \times T^{2/3}\right)$$

Hence, expected regret for explore first then commit is:

2.2 ϵ Greedy

Consider the following example for understanding ϵ Greedy algorithm:

Epsilon-Greedy Algorithm with exploration probabilities$(\epsilon_1, \epsilon_2, ...)$ For *each round* $t = 1, 2, ...$

1. Toss a coin with success (say, the head is a success) with probability ϵ_t
2. If coin lands in head:
   - Pick any arm at random

   else
   - Pick the arm with the best sample average so far

$$\mu^* = \max_{i \,\epsilon \mathbb{A}} \mu_i$$

$$\delta(a) = \mu^* - \mu(a)$$

if $\epsilon_t \approx t^{-1/3}$

Therefore, Epsilon-greedy regret bound is

$$\mathbb{E}[\text{regret till } t] \leq t^{2/3}.O((K \log t)^{1/3}]$$

The proof relies on a more refined clean event.

Explore-first and *Epsilon-greedy* do not adapt their exploration schedule to the history of the observed rewards. This property is known as **Non-Adaptive Exploration**.

References

[1] A. Slivkins. Introduction to multi-armed bandits, 2019.

[2] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.