Brief Introduction of Multi-armed Bandits

L. KONG

EE@CityU

June 23, 2021

A toy case

Figure 1: Two-arm bandits

Time	1	2	3	4	5	6	7	8	9	10
Left arm	10	0			10	10	0			
Right arm			10	0						

Which arm would you play next?

Exploration vs. Exploitation

Online decision-making involves two stages:

1. **Exploitation:** Make the best decision by current information
2. **Exploration:** Gather more information

- The best long-term strategy may involve short-term sacrifices
- Gather enough information to make the best overall decisions

Example: Online Recommendation systems (movies/news/etc)

- **Exploitation:** Show the most successful item
- **Exploration:** Show a different item

A k-armed Bandit Problem

Assumptions:

- The algorithm observes only the reward for the selected action
- The reward for each action be IID and unknown
- Per-round reward be bounded

Problem model:

- Consider k actions (arms) and T rounds
- In each round t, the algorithm chooses an **action** $a_t \in \mathcal{A} = \{1, 2, \ldots, k\}$
- Observe the **reward** $\mu(a_t) \sim P_{a_t}$, where P_{a_t} be reward distribution

Target: maximize total **reward** $r(T) = \mathbb{E}[\sum_{t=1}^{T} \mu(a_t)]$

A k-armed Bandit Problem

Regret:

- The **optimal value** of round t be $\mu(a_t^*) = \max\limits_{a_t \in \mathcal{A}} \mu(a_t)$
- The **gap** $\Delta(a_t) = \mu(a_t^*) - \mu(a_t) \geq 0$
- The **count** of action a in round t be

$$N(a_t) = \begin{cases} 1 \text{ , if } a \text{ be chose} \\ 0 \text{ , otherwise} \end{cases}$$

- The total **regret** be $R(T) = \mathbb{E}[\sum_{t=1}^{T} \Delta(a_t) N(a_t)]$

Target: minimise total regret $\equiv$ maximize total reward

The 10-armed Testbed

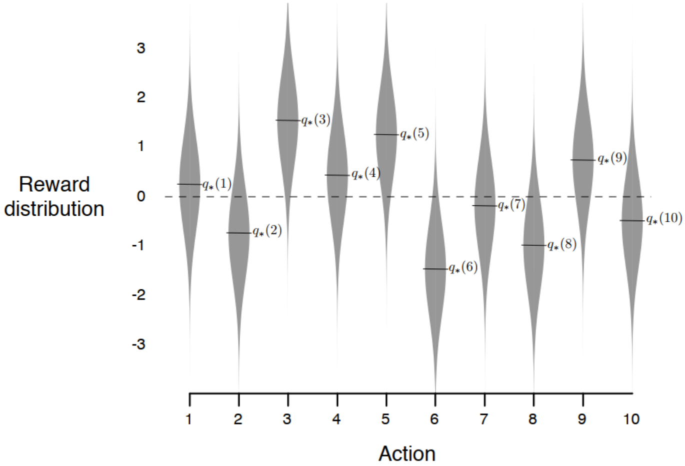

Figure 2: The reward be with a unit-variance normal distribution with the mean value $q_*(a) \sim N(0, 1)$, as suggested by these gray area

Greedy Algorithm

- The **action-value** be the average reward of action a until round t be

$$Q_t(a) = \frac{\sum_{i=1}^{t} \mu(a_i)}{\sum_{i=1}^{t} N(a_i)}$$

- Greedy Algorithm

1. Exploration phase: try each arm (totally k) N times
2. Select the arm with the highest average reward

$$\hat{a} = \underset{a \in \mathcal{A}}{\operatorname{arg}max}\ Q_{kN}(a)$$

3. Exploitation phase: play this arm in all remaining $T - Nk$ rounds

Epsilon-Greedy Algorithm

for each round $t = 1, 2, \ldots, T$ **do**
    Toss a coin with success probability ϵ_t
    if success **then**
        explore: choose an arm uniformly at random
    else
        exploit: choose the arm $\hat{a}$ with the highest $Q_t(a)$
    end
end

- $\epsilon_t = 0 \rightarrow$ Greedy Algorithm
- How to choose the value of ϵ_t?

Average Performance

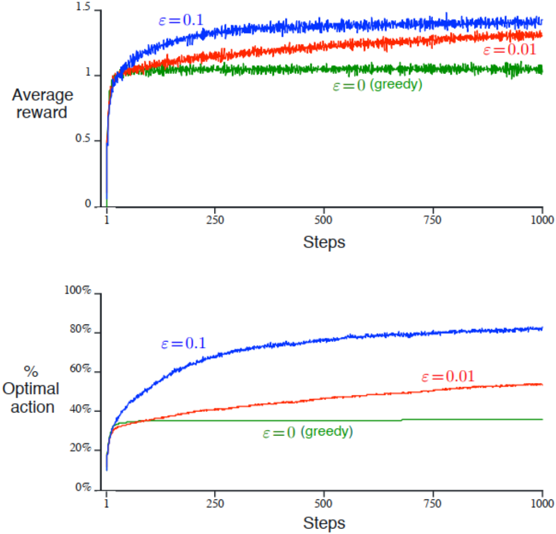

Figure 3: Average performance of ϵ-greedy algorithms on the 10-armed testbed.

Average Performance

- The expected rewards increase with time
- Analyze of ϵ:
  - $\epsilon = 0$: Improved slightly faster at the very beginning, be lowest finally
  - $\epsilon = 0.01$: Improved more slowly, but eventually would perform best (which would not be shown on the figure above)
  - $\epsilon = 0.1$: Explored more, and usually found the sub-optimal action earlier
- Any better choice of ϵ?
  - YES!
  - Reduce the value of ϵ over time
  - Analysis of **regret bound**

Improvement

Incremental Implementation

- The new average of all n rewards
  $Q_{n+1} = \frac{1}{n}\sum_{t=1}^{n} r_t = Q_n + \frac{1}{n}[r_n - Q_n]$
- Compute in a computationally efficient manner
- Work with ϵ-greedy algorithm

Initialize, for $a = 1$ to k:
  $Q(a) \leftarrow 0$
  $N(a) \leftarrow 0$

Loop forever:
  $A \leftarrow \begin{cases} \text{argmax}_a\, Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$
  $R \leftarrow bandit(A)$
  $N(A) \leftarrow N(A) + 1$
  $Q(A) \leftarrow Q(A) + \frac{1}{N(A)}[R - Q(A)]$

Weighted Average Reward

- $Q_{n+1} = Q_n + \alpha_n[r_n - Q_n]$, α_n be the step-size
- Give more weight to recent rewards than to long-past rewards
- Reward probabilities be **non-stationary**

Improvement

Optimistic Initial Values

- Set higher initial action values $Q_1(a)$
- Encourage algorithms to explore

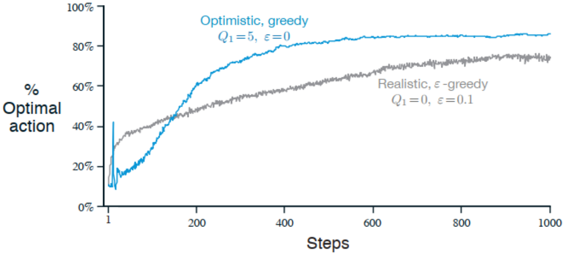

Figure 4: The effect of optimistic initial methods on the 10-armed testbed. Both methods used a constant step-size parameter, $\alpha = 0.1$

UCB Algorithm

Upper Confidence Bound

Consider the **confidence radius** $\rho_t(a)$ of arm a in round t, suppose the reward distribution be bounded in $[0, 1]$ and $\mathbb{E}[Q_t(a)] = Q$

By the **Hoeffding's inequality**, $Pr(|\bar{Q}_t(a) - Q| \geq \rho_t(a)) \leq 2exp(-\frac{2\rho_t^2(a)N_t(a)}{\sum_{i=0}^{N_t(a)}(1-0)^2}) = 2exp(-2\rho_t^2(a)N_t(a)) = p \sim t^{-c}$, where $N_t(a)$ be the number of times that arm a has been selected prior to round t and c controls the degree of exploration

Solving for $\rho_t(a) = c\sqrt{\frac{\ln t}{N_t(a)}}$, so the **upper confidence bound** of action-value be $\texttt{UCB}_t(a) = Q_t(a) + \rho_t(a)$

UCB Algorithm

UCB1 Algorithm

- In each round, choose the arm with highest UCB:

$$\underset{a \in \mathcal{A}}{\arg max}\ \mathtt{UCB}_t(a)$$

- Choose arms according to their potential for being optimal
- Generally performs better than ϵ-greedy algorithm, except in the first some steps

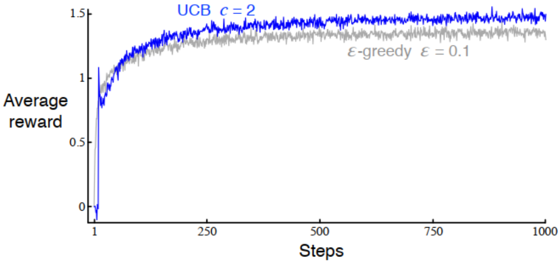

Figure 5: UCB1 vs. ϵ-greedy

Gradient Bandit Algorithm

Boltzmann "soft-max" distribution

- The probability of taking arm a at round t be
  $Pr(A_t = a) = \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} = \pi_t(a)$
- $H_t(a) \in \mathbb{R}$ be the preference for arm a at round t
- Mapping $H_t(\mathcal{A}) \rightarrow \pi_t(\mathcal{A}) \in [0, 1]$

Update the arm preference $H_t(a)$

- $H_{t+1}(A_t) = H_t(A_t) + \alpha(\mu(A_t) - Q_t(A_t)))(1 - \pi_t(A_t))$,
  if $a = A_t$
- $H_{t+1}(a) = H_t(a) - \alpha(\mu(a_t) - Q_t(a))\pi_t(a)$, for all $a \neq A_t$
  - α be the step-size parameter
  - $Q_t(a)$ be a baseline with which the current reward is compared

Gradient Bandit Algorithm

Performance

- The expected reward $q_*(a)$ are chosen to be near $+4$ rather than near 0 (baseline-free)
- With baseline: Adapts to the new level of $q_*(a)$
  - $\alpha = 0.1$: Short-term worse, long-term better
  - $\alpha = 0.4$: Short-term better, long-term worse

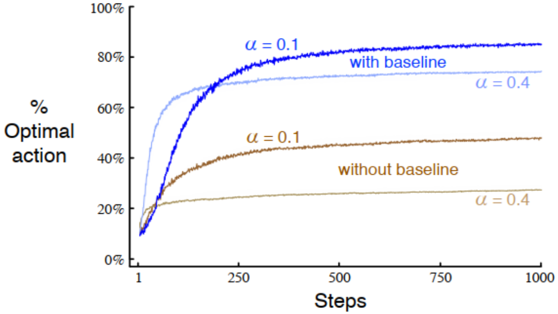

Figure 6: Average performance of the gradient bandit algorithm with and without a reward baseline on the 10-armed testbed

Gradient Bandit Algorithm

Review of Stochastic Gradient Ascent (SGA)

- Consider maximizing an average of functions $\max_x \frac{\sum_{i=1}^m f_i(x)}{m}$
  - Gradient ascent: Update x at the $k+1$ round by

$$x^{(k+1)} = x^{(k)} + \alpha^{(k+1)} \frac{\sum_{i=1}^m \nabla f_i(x^{(k)})}{m}, k = 1, 2, \ldots$$

  - SGA: Update x at the $k+1$ round by

$$x^{(k+1)} = x^{(k)} + \alpha^{(k+1)} \nabla f_{i_{k+1}}(x^{(k)}), k = 1, 2, \ldots,$$

    where $i_k \in \{1, \ldots, m\}$ be the sample index at the round $k+1$

Work as SGA in page 38-40

- Only one sample $a = A_t$ be needed to update the $H_t(\mathcal{A})$ and $\pi_t(\mathcal{A})$
- $\mathcal{O}(kt) \rightarrow \mathcal{O}(t)$

Contextual Bandits

Contextual Bandits (CB)
The reward μ_t in each round t depends on the chosen arm a_t and the context x_t, which be **NEW** to here.

Problem protocol for each round t:

1. Algorithm observes a "context" x_t
2. Algorithm picks an arm a_t
3. Reward μ_t be observed

Go back to the toy case with two lights (Green or Red)

Time	1	2	3	4	5	6
Left arm (light)	10(G)	0(R)	(G)	(G)	10(G)	(R)
Right arm (light)	(R)	(G)	10(R)	0(R)	(R)	(G)

Which arm would you choose for the next round?

Contextual Bandits

Table 1: MAB vs. CB vs. RL

Problem model	MAB	CB	RL
# State	1	>1	>1
States be changed by Actions?	No	No	Yes

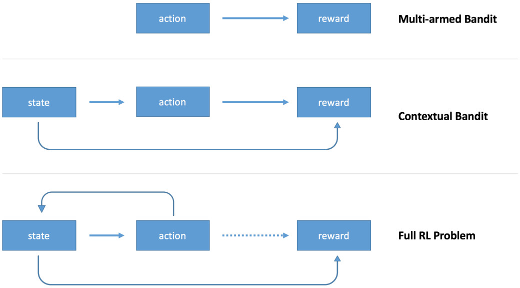

Figure 7: Work flows of three models

References

1. R. S. Sutton and A. G. Barto, Reinforcement Learning: an introduction, A Bradford Book; second edition, 2018.
2. Aleksandrs Slivkins, Introduction to Multi-Armed Bandits (Foundations and Trends(r) in Machine Learning), Now Publishers Inc, 2019.