Class 3: Multi-Arm Bandit

Sutton and Barto, Chapter 2

Sutton slides and Silver
Multi-Arm Bandits

Sutton and Barto, Chapter 2

The simplest reinforcement learning problem
The Exploration/Exploitation Dilemma

Online decision-making involves a fundamental choice:

- **Exploitation**  Make the best decision given current information
- **Exploration**  Gather more information

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

Restaurant Selection
- **Exploitation** Go to your favourite restaurant
- **Exploration** Try a new restaurant

Online Banner Advertisements
- **Exploitation** Show the most successful advert
- **Exploration** Show a different advert

Oil Drilling
- **Exploitation** Drill at the best known location
- **Exploration** Drill at a new location

Game Playing
- **Exploitation** Play the move you believe is best
- **Exploration** Play an experimental move
You are the algorithm! (bandit1)

- **Action 1** — Reward is always 8
  - value of action 1 is \( q_*(1) = \)

- **Action 2** — 88% chance of 0, 12% chance of 100!
  - value of action 2 is \( q_*(2) = .88 \times 0 + .12 \times 100 = \)

- **Action 3** — Randomly between -10 and 35, equiprobable
  - \( q_*(3) = \)

- **Action 4** — a third 0, a third 20, and a third from \{8,9,...,18\}
  - \( q_*(4) = \)
The $k$-armed Bandit Problem

- On each of a sequence of time steps, $t=1,2,3,\ldots$, you choose an action $A_t$ from $k$ possibilities, and receive a real-valued reward $R_t$

- The reward depends only on the action taken; it is identically, independently distributed (i.i.d.):

$$q_*(a) = \mathbb{E}[R_t | A_t = a], \ \forall a \in \{1, \ldots, k\}$$

- These true values are unknown. The distribution is unknown

- Nevertheless, you must maximize your total reward

- You must both try actions to learn their values (explore), and prefer those that appear best (exploit)
The Exploration/Exploitation Dilemma

- Suppose you form estimates
  \[ Q_t(a) \approx q_*(a), \quad \forall a \]
  action-value estimates

- Define the greedy action at time \( t \) as
  \[ A^*_t \doteq \arg \max_a Q_t(a) \]

- If \( A_t = A^*_t \), then you are exploiting
- If \( A_t \neq A^*_t \), then you are exploring

- You can’t do both, but you need to do both

- You can never stop exploring, but maybe you should explore less with time. Or maybe not.
Regret

The action-value is the mean reward for action $a$,

- $q^*(a) = E[r | a]$

The optimal value $V^*$ is

- $V^* = Q(a^*) = \max_{a \in A} q^*(a)$

The regret is the opportunity loss for one step

- $l_t = E[V^* - Q(a_t)]$

The total regret is the total opportunity loss

$$L_t = E\left[\sum_{\tau=1}^{t} V^* - Q(a_{\tau})\right]$$

- Maximise cumulative reward $\equiv$ minimise total regret
The count $N_t(a)$ is expected number of selections for action $a$

The gap $\Delta_a$ is the difference in value between action $a$ and optimal action $a^*$, $\Delta_a = V^* - Q(a)$

Regret is a function of gaps and the counts

$$L_t = \mathbb{E} \left[ \sum_{\tau=1}^{t} V^* - Q(a_\tau) \right]$$

$$= \sum_{a \in A} \mathbb{E} [N_t(a)] (V^* - Q(a))$$

$$= \sum_{a \in A} \mathbb{E} [N_t(a)] \Delta_a$$

A good algorithm ensures small counts for large gaps

Problem: gaps are not known!
If an algorithm **forever** explores it will have linear total regret

If an algorithm **never** explores it will have linear total regret Is it possible to achieve sublinear total regret?
Complexity of regret

- The performance of any algorithm is determined by similarity between optimal arm and other arms.
- Hard problems have similar-looking arms with different means.
- This is described formally by the gap $\Delta_a$ and the similarity in distributions $KL(\mathcal{R}^a\|\mathcal{R}^{a^*})$.

**Theorem (Lai and Robbins)**

Asymptotic total regret is at least logarithmic in number of steps.

$$\lim_{t \to \infty} L_t \geq \log t \sum_{a | \Delta_a > 0} \frac{\Delta_a}{KL(\mathcal{R}^a\|\mathcal{R}^{a^*})}$$
Overview

• Action-value methods
  – Epsilon-greedy strategy
  – Incremental implementation
  – Stationary vs. non-stationary environment
  – Optimistic initial values
• UCB action selection
• Gradient bandit algorithms
• Associative search (contextual bandits)
Basics

• Maximize total reward collected
  – vs learn (optimal) policy (RL)
• Episode is one step
• Complex function of
  – True value
  – Uncertainty
  – Number of time steps
  – Stationary vs non-stationary?
Action-Value Methods

- Methods that learn action-value estimates and nothing else

- For example, estimate action values as *sample averages*:

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

- The sample-average estimates converge to the true values *if* the action is taken an infinite number of times

\[ \lim_{N_t(a) \to \infty} Q_t(a) = q_*(a) \]

The number of times action \( a \) has been taken by time \( t \)
$\varepsilon$-Greedy Action Selection

- In greedy action selection, you always exploit.

- In $\varepsilon$-greedy, you are usually greedy, but with probability $\varepsilon$ you instead pick an action at random (possibly the greedy action again).

- This is perhaps the simplest way to balance exploration and exploitation.
A simple bandit algorithm

Initialize, for \( a = 1 \) to \( k \):
\[
\begin{align*}
Q(a) &\leftarrow 0 \\
N(a) &\leftarrow 0
\end{align*}
\]

Repeat forever:
\[
\begin{align*}
A &\leftarrow \begin{cases} 
\arg \max_a Q(a) & \text{with probability } 1 - \varepsilon \\
\text{a random action} & \text{with probability } \varepsilon
\end{cases} \quad \text{(breaking ties randomly)} \\
R &\leftarrow \text{bandit}(A) \\
N(A) &\leftarrow N(A) + 1 \\
Q(A) &\leftarrow Q(A) + \frac{1}{N(A)} [R - Q(A)]
\end{align*}
\]
One Bandit Task from

The 10-armed Testbed

Figure 2.1: An example bandit problem from the 10-armed testbed. The true value $q(a)$ 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 $q(a)$ unit variance normal distribution, as suggested by these gray distributions.
ε-Greedy Methods on the 10-Armed Testbed
Averaging ⟷ learning rule

• To simplify notation, let us focus on one action
  
  • We consider only its rewards, and its estimate after \( n+1 \) rewards:

\[
Q_n = \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1}
\]

• How can we do this incrementally (without storing all the rewards)?
  
• Could store a running sum and count (and divide), or equivalently:

\[
Q_{n+1} = Q_n + \frac{1}{n} \left[ R_n - Q_n \right]
\]

• This is a standard form for learning/update rules:

\[
\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize} \left[ \text{Target} - \text{OldEstimate} \right]
\]
Derivation of incremental update

\[ Q_n = \frac{R_1 + R_2 + \cdots + R_{n-1}}{n-1} \]

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

\[ = \frac{1}{n} \left( R_n + \sum_{i=1}^{n-1} R_i \right) \]

\[ = \frac{1}{n} \left( R_n + (n-1) \frac{1}{n-1} \sum_{i=1}^{n-1} R_i \right) \]

\[ = \frac{1}{n} \left( R_n + (n-1)Q_n \right) \]

\[ = \frac{1}{n} \left( R_n + nQ_n - Q_n \right) \]

\[ = Q_n + \frac{1}{n} \left[ R_n - Q_n \right] , \]
Tracking a Non-stationary Problem

• Suppose the true action values change slowly over time
  • then we say that the problem is nonstationary

• In this case, sample averages are not a good idea (Why?)

• Better is an “exponential, recency-weighted average”:

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

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

where \( \alpha \) is a constant, step-size parameter, \( 0 < \alpha \leq 1 \)

• There is bias due to \( Q_1 \) that becomes smaller over time
Standard stochastic approximation convergence conditions

• To assure convergence with probability 1:

\[ \sum_{n=1}^{\infty} \alpha_n(a) = \infty \quad \text{and} \quad \sum_{n=1}^{\infty} \alpha_n^2(a) < \infty \]

• e.g., \( \alpha_n = \frac{1}{n} \)

• not \( \alpha_n = \frac{1}{n^2} \)

if \( \alpha_n = n^{-p}, \ p \in (0, 1) \) then convergence is at the optimal rate:

\[ O(1/\sqrt{n}) \]
Optimistic Initial Values

• All methods so far depend on $Q_1(a)$, i.e., they are biased. So far we have used $Q_1(a) = 0$

• Suppose we initialize the action values \textit{optimistically} ($Q_1(a) = 5$), e.g., on the 10-armed testbed (with alpha= 0.1)
Upper Confidence Bound (UCB) action selection

- A clever way of reducing exploration over time
- Focus on actions whose estimate has large degree of uncertainty
- Estimate an upper bound on the true action values
- Select the action with the largest (estimated) upper bound

\[ A_t = \arg\max_a \left[ Q_t(a) + c \sqrt{\frac{\log t}{N_t(a)}} \right] \]
Complexity of UCB Algorithm

Theorem

The UCB algorithm achieves logarithmic asymptotic total regret

\[
\lim_{t \to \infty} L_t \leq 8 \log t \quad \Delta_a \\
\quad a | \Delta_a > 0
\]
Gradient-Bandit Algorithms

- Let $H_t(a)$ be a learned preference for taking action $a$

\[ \Pr\{A_t=a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a) \]

\[ H_{t+1}(A_t) = H_t(A_t) + \alpha (R_t - \bar{R}_t) (1 - \pi_t(A_t)), \quad \text{and} \]

\[ H_{t+1}(a) = H_t(a) - \alpha (R_t - \bar{R}_t) \pi_t(a), \quad \text{for all } a \neq A_t, \]  

(2.10)

\[ \bar{R}_t = \frac{1}{t} \sum_{i=1}^{t} R_i \]

Optimal action
Derivation of gradient-bandit algorithm

In exact gradient ascent:

$$H_{t+1}(a) = H_t(a) + \alpha \frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)},$$  \hspace{1cm} (1)$$

where:

$$\mathbb{E}[R_t] = \sum_b \pi_t(b)q_*(b),$$

$$\frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)} = \frac{\partial}{\partial H_t(a)} \left[ \sum_b \pi_t(b)q_*(b) \right]$$

$$= \sum_b q_*(b) \frac{\partial \pi_t(b)}{\partial H_t(a)}$$

$$= \sum_b (q_*(b) - X_t) \frac{\partial \pi_t(b)}{\partial H_t(a)},$$

where $X_t$ does not depend on $b$, because $\sum_b \frac{\partial \pi_t(b)}{\partial H_t(a)} = 0.$
\[
\frac{\partial \mathbb{E}[R_t]}{\partial H_t(a)} = \sum_b (q_*(b) - X_t) \frac{\partial \pi_t(b)}{\partial H_t(a)} \\
= \sum_b \pi_t(b)(q_*(b) - X_t) \frac{\partial \pi_t(b)}{\partial H_t(a)}/\pi_t(b) \\
= \mathbb{E}\left[(q_*(A_t) - X_t) \frac{\partial \pi_t(A_t)}{\partial H_t(a)}/\pi_t(A_t)\right] \\
= \mathbb{E}\left[(R_t - \bar{R}_t) \frac{\partial \pi_t(A_t)}{\partial H_t(a)}/\pi_t(A_t)\right],
\]

where here we have chosen \( X_t = \bar{R}_t \) and substituted \( R_t \) for \( q_*(A_t) \), which is permitted because \( \mathbb{E}[R_t|A_t] = q_*(A_t) \).

For now assume: \( \frac{\partial \pi_t(b)}{\partial H_t(a)} = \pi_t(b)(1_{a=b} - \pi_t(a)) \). Then:

\[
= \mathbb{E}\left[(R_t - \bar{R}_t)\pi_t(A_t)(1_{a=A_t} - \pi_t(a))/\pi_t(A_t)\right] \\
= \mathbb{E}\left[(R_t - \bar{R}_t)(1_{a=A_t} - \pi_t(a))\right].
\]

\( H_{t+1}(a) = H_t(a) + \alpha(R_t - \bar{R}_t)(1_{a=A_t} - \pi_t(a)), \) (from (1), QED)
Thus it remains only to show that
\[
\frac{\partial \pi_t(b)}{\partial H_t(a)} = \pi_t(b)(\mathbf{1}_{a=b} - \pi_t(a)).
\]

Recall the standard quotient rule for derivatives:
\[
\frac{\partial}{\partial x} \left[ \frac{f(x)}{g(x)} \right] = \frac{\frac{\partial f(x)}{\partial x} g(x) - f(x) \frac{\partial g(x)}{\partial x}}{g(x)^2}.
\]

Using this, we can write...
Quotient Rule: \[
\frac{\partial}{\partial x} \left[ \frac{f(x)}{g(x)} \right] = \frac{\frac{\partial f(x)}{\partial x} g(x) - f(x) \frac{\partial g(x)}{\partial x}}{g(x)^2}
\]

\[
\frac{\partial \pi_t(b)}{\partial H_t(a)} = \frac{\partial}{\partial H_t(a)} \pi_t(b)
\]

\[
= \frac{\partial}{\partial H_t(a)} \left[ \frac{e^{h_t(b)}}{\sum_{c=1}^{k} e^{h_t(c)}} \right]
\]

\[
= \frac{\frac{\partial e^{h_t(b)}}{\partial H_t(a)} \sum_{c=1}^{k} e^{h_t(c)} - e^{h_t(b)} \frac{\partial}{\partial H_t(a)} \sum_{c=1}^{k} e^{h_t(c)}}{\left( \sum_{c=1}^{k} e^{h_t(c)} \right)^2}
\]

\[
= \frac{1_{a=b} e^{h_t(a)} \sum_{c=1}^{k} e^{h_t(c)} - e^{h_t(b)} e^{h_t(a)}}{\left( \sum_{c=1}^{k} e^{h_t(c)} \right)^2}
\]

\[
= \frac{1_{a=b} e^{h_t(b)}}{\sum_{c=1}^{k} e^{h_t(c)}} - \frac{e^{h_t(b)} e^{h_t(a)}}{\left( \sum_{c=1}^{k} e^{h_t(c)} \right)^2}
\]

\[
= 1_{a=b} \pi_t(b) - \pi_t(b) \pi_t(a)
\]

\[
= \pi_t(b) (1_{a=b} - \pi_t(a)).
\]

(Q.E.D.)
Summary Comparison of Bandit Algorithms

The diagram illustrates the performance of different bandit algorithms: ε-greedy, UCB, greedy with optimistic initialization (α = 0.1), and gradient bandit. The x-axis represents the parameter α/c/Q₀, and the y-axis shows the average reward over the first 1000 steps.
Conclusions

- These are all simple methods
  - but they are complicated enough—we will build on them
  - we should understand them completely
  - there are still open questions

- Our first algorithms that learn from evaluative feedback
  - and thus must balance exploration and exploitation

- Our first algorithms that appear to have a goal
  —that learn to maximize reward by trial and error