Reinforcement Learning 1 Q&A

Oana Cocarascu & Helen Yannakoudakis

Department of Informatics
King's College London

n-armed bandits

- *n*-armed bandit problem:
  - Evaluative feedback = how good was the action (does not say if this was the best (or worst!) action)
  - Non-associative = learns the one best output

n-armed bandits

- *n*-armed bandit problem: You can choose between n actions. After each choice you receive a reward. The goal is to maximise the reward in the long run.
- After each play a_t, you get a reward r_t: $E\langle r_t|a_t\rangle = Q^*(a)$
- $Q^*(a)$ = the value of action a (the expected reward given that a is selected).
- We don't know the action values with certainty, but we have estimates.
- Action value estimates: $Q_t(a)$ estimates $Q^*(a)$.
- $Q_t(a)$ = the estimated value of a at time step t.

n-armed bandits

- Action value estimates: $Q_t(a)$ which estimates $Q^*(a)$
- "Sample-average" method: have a list of all rewards received and average them for each action. Assuming by t-th play, a had been chosen k_a times, we have:

$$Q_t(a) = \frac{r_1 + r_2 + \ldots + r_{k_a}}{k_a}$$

- Given an estimate of a reward for each action, we can do greedy action selection (maximise immediate reward):

$$a_t = a_t^* = \arg\max_a Q_t(a)$$

Incremental implementation

- If Q_k is the average of the first k rewards and r_{k+1} is the $k+1$th reward, then:

$$Q_{k+1} = \frac{1}{k+1}\sum_{i=1}^{k+1} r_i$$

Incremental implementation

- If Q_k is the average of the first k rewards and r_{k+1} is the $k+1$th reward, then:

$$Q_{k+1} = \frac{1}{k+1}\sum_{i=1}^{k+1} r_i = \frac{1}{k+1}(r_1 + r_2 + \cdots + r_k + r_{k+1})$$

Incremental implementation

- If Q_k is the average of the first k rewards and r_{k+1} is the $k+1$th reward, then:

$$
\begin{aligned}
Q_{k+1} &= \frac{1}{k+1}\sum_{i=1}^{k+1} r_i = \frac{1}{k+1}(r_1 + r_2 + \cdots + r_k + r_{k+1}) \\
&= \frac{1}{k+1}\left(r_{k+1} + \sum_{i=1}^{k} r_i\right)
\end{aligned}
$$

Incremental implementation

- If Q_k is the average of the first k rewards and r_{k+1} is the $k+1$th reward, then:

$$
\begin{aligned}
Q_{k+1} &= \frac{1}{k+1}\sum_{i=1}^{k+1} r_i = \frac{1}{k+1}(r_1 + r_2 + \cdots + r_k + r_{k+1}) \\
&= \frac{1}{k+1}\left(r_{k+1} + \sum_{i=1}^{k} r_i\right)
\end{aligned}
$$

- We know that:

$$
Q_{k+1} = \frac{1}{k+1}\sum_{i=1}^{k+1} r_i \qquad \Rightarrow \qquad Q_k = \frac{1}{k}\sum_{i=1}^{k} r_i
$$

Incremental implementation

- If Q_k is the average of the first k rewards and r_{k+1} is the $k+1$th reward, then:

$$\begin{aligned} Q_{k+1} &= \frac{1}{k+1}\sum_{i=1}^{k+1} r_i = \frac{1}{k+1}(r_1 + r_2 + \cdots + r_k + r_{k+1}) \\ &= \frac{1}{k+1}\left(r_{k+1} + \sum_{i=1}^{k} r_i\right) \end{aligned}$$

- We know that:

$$Q_{k+1} = \frac{1}{k+1}\sum_{i=1}^{k+1} r_i \quad \Rightarrow \quad Q_k = \frac{1}{k}\sum_{i=1}^{k} r_i$$

- So we can write:

$$Q_{k+1} = \frac{1}{k+1}\left(r_{k+1} + kQ_k\right)$$

KING'S College LONDON

Incremental implementation

$$Q_{k+1} = \frac{1}{k+1}\left(r_{k+1} + kQ_k\right)$$

Incremental implementation

$$
\begin{aligned}
Q_{k+1} &= \frac{1}{k+1}\left(r_{k+1} + kQ_k\right) \\
&= \frac{1}{k+1}\left(r_{k+1} + kQ_k + Q_k - Q_k\right)
\end{aligned}
$$

Incremental implementation

$$
\begin{aligned}
Q_{k+1} &= \frac{1}{k+1}\left(r_{k+1} + kQ_k\right) \\
&= \frac{1}{k+1}\left(r_{k+1} + kQ_k + Q_k - Q_k\right) \\
&= \frac{1}{k+1}\left(r_{k+1} + (k+1)Q_k - Q_k\right)
\end{aligned}
$$

Incremental implementation

$$
\begin{aligned}
Q_{k+1} &= \frac{1}{k+1}\left(r_{k+1} + kQ_k\right) \\
&= \frac{1}{k+1}\left(r_{k+1} + kQ_k + Q_k - Q_k\right) \\
&= \frac{1}{k+1}\left(r_{k+1} + (k+1)Q_k - Q_k\right) \\
&= Q_k + \frac{1}{k+1}\left(r_{k+1} - Q_k\right)
\end{aligned}
$$

ϵ-greedy

- Use ϵ-greedy to balance exploration-exploitation:

$$a_t = \begin{cases} a_t^* & \text{with probability} \quad 1 - \epsilon \\ \textit{random action} & \text{with probability} \quad \epsilon, \qquad \epsilon \ll 1 \end{cases}$$

- ϵ-greedy makes a random choice among non-optimal actions. Instead, use softmax to select non-optimal actions based on their reward, with probability:

$$\frac{e^{\frac{Q_t(a)}{\tau}}}{\sum_{b=1}^{n} e^{\frac{Q_t(b)}{\tau}}}$$

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- An agent that is choosing between a_1, a_2 and a_3, with average rewards: $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$.
- What is the probability that each action will be selected if the agent uses ϵ-greedy action selection with $\epsilon = 0.1$?

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.

$$a_t = \begin{cases} a_t^* & \text{with probability} \quad 1 - \epsilon \\ \textit{random action} & \text{with probability} \quad \epsilon, \qquad \epsilon \ll 1 \end{cases}$$

- Q: Which action has the greatest expected reward?

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.

$$a_t = \begin{cases} a_t^* & \text{with probability} \quad 1 - \epsilon \\ \textit{random action} & \text{with probability} \quad \epsilon, \qquad \epsilon \ll 1 \end{cases}$$

- Q: Which action has the greatest expected reward?
- A: a_2

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.

$$a_t = \begin{cases} a_t^* & \text{with probability} \quad 1 - \epsilon \\ \textit{random action} & \text{with probability} \quad \epsilon, \qquad \epsilon \ll 1 \end{cases}$$

- Q: Which action has the greatest expected reward?
- A: a_2
- Q: a_2 will be selected with probability ...?

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.

$$a_t = \begin{cases} a_t^* & \text{with probability} \quad 1 - \epsilon \\ \textit{random action} & \text{with probability} \quad \epsilon, \qquad \epsilon \ll 1 \end{cases}$$

- Q: Which action has the greatest expected reward?
- A: a_2
- Q: a_2 will be selected with probability ...?
- A: a_2 will be selected with probability 0.9.

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.

$$a_t = \begin{cases} a_t^* & \text{with probability} \quad 1 - \epsilon \\ \textit{random action} & \text{with probability} \quad \epsilon, \qquad \epsilon \ll 1 \end{cases}$$

- From the definition, we will select a random action with probability 0.1.
- Q: a_1 will be selected with probability ...?

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.

$$a_t = \begin{cases} a_t^* & \text{with probability} \quad 1 - \epsilon \\ \textit{random action} & \text{with probability} \quad \epsilon, \qquad \epsilon \ll 1 \end{cases}$$

- From the definition, we will select a random action with probability 0.1.
- Q: a_1 will be selected with probability ...?
- A: a_1 will be selected with probability 0.033.

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.
- Softmax using the Gibbs distribution, $\tau = 0.1$:

$$P(a_i) = \frac{e^{\frac{Q(a_i)}{\tau}}}{\sum_{a_j \in A} e^{\frac{Q(a_j)}{\tau}}}$$

$$\begin{aligned} e_1 &= e^{\frac{Q(a_1)}{\tau}} = ? \\ e_2 &= e^{\frac{Q(a_2)}{\tau}} = ? \\ e_3 &= e^{\frac{Q(a_3)}{\tau}} = ? \end{aligned}$$

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.
- Softmax using the Gibbs distribution, $\tau = 0.1$:

$$P(a_i) = \frac{e^{\frac{Q(a_i)}{\tau}}}{\sum_{a_j \in A} e^{\frac{Q(a_j)}{\tau}}}$$

$$\begin{aligned} e_1 &= e^{\frac{Q(a_1)}{\tau}} = e^{\frac{5}{0.1}} = 5.18 \times 10^{21} \\ e_2 &= e^{\frac{Q(a_2)}{\tau}} = e^{\frac{7}{0.1}} = 2.5 \times 10^{30} \\ e_3 &= e^{\frac{Q(a_3)}{\tau}} = e^{\frac{4}{0.1}} = 2.35 \times 10^{17} \end{aligned}$$

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.
- Softmax using the Gibbs distribution, $\tau = 0.1$:

$$P(a_i) = \frac{e^{\frac{Q(a_i)}{\tau}}}{\sum_{a_j \in A} e^{\frac{Q(a_j)}{\tau}}}$$

$P(a_1) = ?$

$P(a_2) = ?$

$P(a_3) = ?$

ϵ-greedy / softmax exercise (Ex2 from Tutorial 7)

- $Q(a_1) = 5$, $Q(a_2) = 7$, $Q(a_3) = 4$ and ϵ-greedy, $\epsilon = 0.1$.
- Softmax using the Gibbs distribution, $\tau = 0.1$:

$$P(a_i) = \frac{e^{\frac{Q(a_i)}{\tau}}}{\sum_{a_j \in A} e^{\frac{Q(a_j)}{\tau}}}$$

$$\begin{aligned}
e_1 &= e^{\frac{Q(a_1)}{\tau}} = e^{\frac{5}{0.1}} = 5.18 \times 10^{21} \Rightarrow & P(a_1) &= \frac{e_1}{e_1+e_2+e_3} \approx 0 \\
e_2 &= e^{\frac{Q(a_2)}{\tau}} = e^{\frac{7}{0.1}} = 2.5 \times 10^{30} \Rightarrow & P(a_2) &\approx 1 \\
e_3 &= e^{\frac{Q(a_3)}{\tau}} = e^{\frac{4}{0.1}} = 2.35 \times 10^{17} \Rightarrow & P(a_3) &\approx 0
\end{aligned}$$

Scenario

- At the casino, there is a slot machine with n levers.

- Each lever has a different reward.
- The gambler does not know the probability distribution for the reward corresponding to each lever.
- The gambler wants to maximise the reward (£££).
- What can the gambler do?
- Exploration/exploitation.

10-armed testbed

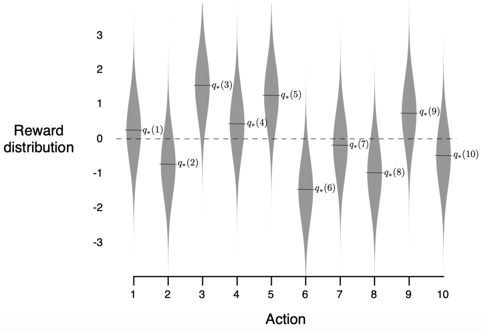
- Bandit with $n = 10$ possible actions.
- Each $Q^*(a)$ is chosen randomly from a normal distribution:

$$\mathcal{N}(\mu = 0, \sigma = 1)$$

- Each r_t is also chosen from a normal distribution (notice the t):

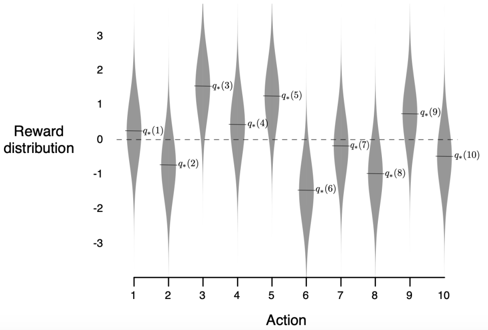
$$\mathcal{N}(\mu = Q^*(a_t), \sigma = 1)$$

- An example of bandit:



10-armed testbed

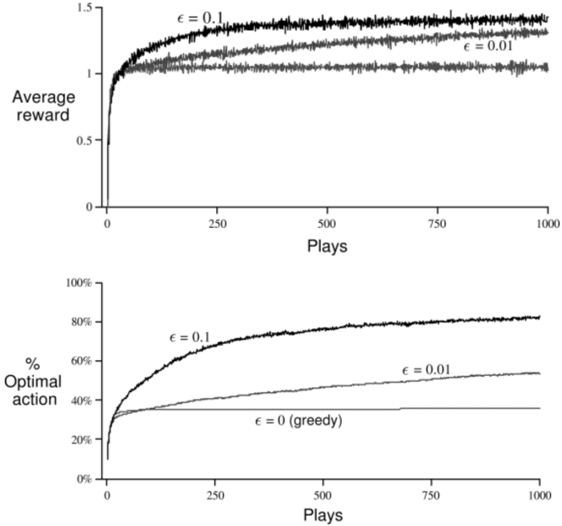
- An example of bandit:



- The 10-armed testbed: 1000 plays and 2000 randomly generated 10-armed bandit tasks.
    - One experiment: A bandit task, $Q^*(a)$ chosen as on previous slide, plot over 1000 plays (time steps).
    - Do 2000 experiments (each time, different bandit problem).
    - Average over these and plot performance.

10-armed testbed

- Greedy method compared with two epsilon-greedy methods.



Markov decision process (MDP)

- Captures any fully observable non-deterministic environment with a Markovian transition model and additive rewards.
- Mathematically we have: states, actions, a transition model, a reward function.
- The utility function is defined over sequences of states (an environment history) rather than on a single state.
- Utility is the long term total reward from s onwards whereas reward is the short term reward from s (in each state s the agent receives a reward).
- A solution to MDP is a policy.
- We would like the optimal policy π^* (i.e. the choice of action for every state that maximises overall cumulative reward).

Optimal policies

- To get the optimal policy, we calculate the utility for each state.
- If we have the values of states under an optimal policy, the agent just picks the action *a* that maximises the expected utility of the next state:

$$\pi^*(s) = \arg\max_{a \in A(s)} \sum_{s'} P(s'|s,a) U^{\pi^*}(s')$$

- To compute these values, we use value iteration.

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s'|s,a) U_i(s')$$

where $0 < \gamma \leqslant 1$ is the discount rate.
This update computes the utility of each state by doing a maximum expected utility calculation on the current utility of the states around it.

- Consider an agent which has established, using value iteration, the utility values shown in the figure. What policy should this agent adopt?

3	0.812	0.868	0.918	+1
2	0.762		0.660	–1
1	0.705	0.655	0.611	0.388
	1	2	3	4

Grid example - Ex3 Tutorial 7

- Consider an agent which has established, using value iteration, the utility values shown in the figure. What policy should this agent adopt?

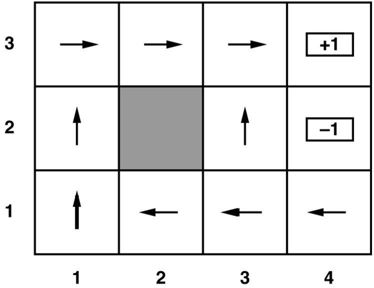
3	0.812	0.868	0.918	+1
2	0.762		0.660	–1
1	0.705	0.655	0.611	0.388
	1	2	3	4

- Calculate: $EU(Up_{(3,1)})$, $EU(Left_{(3,1)})$, $EU(Down_{(3,1)})$, $EU(Right_{(3,1)})$.
- In state (3,1), the optimal thing to do is to go left (and not up where the value is higher).

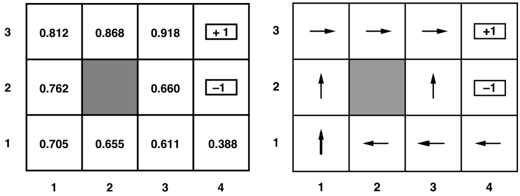
Grid example - Ex3 Tutorial 7

- In state (3,1), the optimal thing to do is to go left (and not up where the value is higher).

	1	2	3	4
3	0.812	0.868	0.918	+1
2	0.762		0.660	–1
1	0.705	0.655	0.611	0.388



Passive learning



- We can solve this as an MDP.
- What about learning? The agent does not know the transition model or the reward function $R(s)$. How does it learn $U^{\pi}(s)$?
- In passive reinforcement learning the agent's policy is fixed.
- Agent learns utility $U^{\pi}(s)$ by carrying out runs through the environment, following some policy π.

Passive vs Active reinforcement learning

- RL
  - The agent doesn't know the transition model or the reward function.
  - In each state, the agent receives a reward.
- Passive RL
  - Goal: evaluate how good a policy is.
  - Direct Utility Estimation
  - Adaptive Dynamic Programming(ADP): The utility of a state is the reward for being in that state plus the expected discounted reward of being in the next state.
  - Temporal difference (TD) learning: model-free (i.e., no transition model)
- Active RL
  - Goal: learn an optimal policy by choosing actions.
  - ADP with exploration function (learns a transition model).
  - Q-learning is a TD learning method (no transition model).
- More on this next week in RL2.

Grid example - Ex4 Tutorial 7

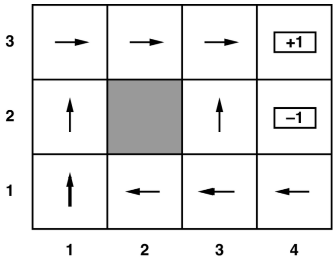
- Consider the following runs (actions are stochastic):

$$
\begin{aligned}
(1,1)_{-0.04} &\rightarrow (1,2)_{-0.04} \rightarrow (1,3)_{-0.04} \rightarrow (1,3)_{-0.04} \rightarrow (2,3)_{-0.04} \\
&\rightarrow (2,3)_{-0.04} \rightarrow (2,3)_{-0.04} \rightarrow (3,3)_{-0.04} \rightarrow (3,2)_{-0.04} \\
&\rightarrow (3,3)_{-0.04} \rightarrow (3,2)_{-0.04} \rightarrow (3,3)_{-0.04} \rightarrow (4,3)_{1}
\end{aligned}
$$

$$
\begin{aligned}
(1,1)_{-0.04} &\rightarrow (1,2)_{-0.04} \rightarrow (1,2)_{-0.04} \rightarrow (1,3)_{-0.04} \\
&\rightarrow (2,3)_{-0.04} \rightarrow (3,3)_{-0.04} \rightarrow (4,3)_{1}
\end{aligned}
$$

$$
\begin{aligned}
(1,1)_{-0.04} &\rightarrow (1,1)_{-0.04} \rightarrow (1,2)_{-0.04} \rightarrow (1,3)_{-0.04} \\
&\rightarrow (2,3)_{-0.04} \rightarrow (3,3)_{-0.04} \rightarrow (4,3)_{1}
\end{aligned}
$$

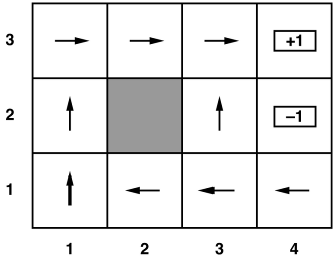
Grid example - Ex4 Tutorial 7



$$(1,1)_{-0.04} \to (1,2)_{-0.04} \to (1,3)_{-0.04} \to (1,3)_{-0.04} \to (2,3)_{-0.04}$$
$$\to (2,3)_{-0.04} \to (2,3)_{-0.04} \to (3,3)_{-0.04} \to (3,2)_{-0.04}$$
$$\to (3,3)_{-0.04} \to (3,2)_{-0.04} \to (3,3)_{-0.04} \to (4,3)_{1}$$

- Estimated utility: $U(1,1) = 1 - 12 \times 0.04 = 0.52$
- $U(3,2) = 0.84$ and $U(3,2) = 0.92 \Rightarrow U(3,2) = 0.88$

Grid example - Ex4 Tutorial 7



$$(1,1)_{-0.04} \rightarrow (1,2)_{-0.04} \rightarrow (1,3)_{-0.04} \rightarrow (1,3)_{-0.04} \rightarrow (2,3)_{-0.04}$$
$$\rightarrow (2,3)_{-0.04} \rightarrow (2,3)_{-0.04} \rightarrow (3,3)_{-0.04} \rightarrow (3,2)_{-0.04}$$
$$\rightarrow (3,3)_{-0.04} \rightarrow (3,2)_{-0.04} \rightarrow (3,3)_{-0.04} \rightarrow (4,3)_{1}$$

$$P((2,3)|(2,3), Right) = 2 \times 1/3 = 0.667$$
$$P((3,3)|(2,3), Right) = 1/3 = 0.333$$