# Multi-Armed Bandits

David I. Inouye

PURDUE UNIVERSITY

Elmore Family School of Electrical and Computer Engineering

# The **multi-armed bandit** problem is inspired by a row of slot machines

- A gambler is the **agent**
- The row of slot machines is the **environment**
- The agent can take an **action** by pulling a slot machine's "arm"
- The slot machine payout (or lack thereof) is the **reward** signal

https://medium.com/growth-book/guide-to-multi-arm-bandits-what-is-it-and-why-you-probably-shouldnt-use-it-ecc9bb2e5a84

# Interactive multi-armed bandit demo

# **Multi-armed bandits** are a simplification of RL yet they retain core RL-specific ideas

- The environment only has a **single state**
    - “Observing” the environment state is not necessary since it’s always the same

- The environment **does not change** (in the vanilla bandit problem)
    - The **distribution of rewards** does not change over time *or* due to actions
    - For example, the payout probabilities for each slot machine are fixed

- At every timestep, the agent can choose **any action**

- The only problem is **lack of knowledge**
    - If we knew which machine gave the highest average payout, we would just take that optimal action again and again.
    - If it was supervised learning, only one example of the “correct” action would be enough!
    - The **explore-exploit tradeoff** still exists because of **uncertainty**

# **Multi-armed bandits** are a simplification of RL yet they retain core RL-specific ideas

- Bandits isolate the unique feature of RL regarding "feedback"
- **Instructive feedback** provides the correct action no matter which action was already taken (e.g., supervised learning)
  - The optimal action $a_t^*$ (equivalently, ground truth label $y^*$) is the "feedback" given to a supervised learning system **regardless of the actual action $a_t$** (equivalently, system's prediction $\hat{y}$)
- **Evaluative feedback** provides a reward **depending on the action actually taken**
  - The reward signal is a function of the action actually taken $a_t$, i.e., $R(a_t)$.
  - Thus, the environment **evaluates the *actual* action/decision** made.

# How do we design a **policy** that maximizes the sum of rewards?

- We could just do a completely random policy that randomly chooses an action at every time step

$$A_t \sim \text{Uniform}(\{1,2,\dots,K\})$$

- This is good because it is **simple** and achieves an average reward over all choices
    - It chooses good and bad actions evenly
    - It completely ignores the past (i.e., ignores its experience)

- However, it will often take an action that gives **suboptimal reward**

# A better approach is to estimate the value of each action to determine optimal actions

- First, we will define the <u>**value**</u> of an action as the expected reward given this action:

$$q_*(a) \coloneqq \mathbb{E}[R_t | A_t = a]$$

  - $R_t$ represents the reward random variable at time $t$
  - $A_t$ represents the action random variable at time $t$
  - $a$ represents a specific action

- If we knew the $q_*$, then the problem would be trivial, just repeatedly take

$$A_t = \ a^* = \arg\max_a q_*(a)$$

- Obviously, we do not know $q_*$ but we can **approximate it** given our previous actions:

$$Q_t(a) \approx q_*(a)$$

# A **sample average** can be used to estimate the expectation

- We can estimate $q_*$ by using a sample average over the past actions and rewards:

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

- As an example, suppose the past rewards and actions are:

$$A = [1,2,2,1,2,2,2]$$
$$R = [0,1,1,1,0,1,1]$$

- If $t = 6$, then $Q_6(1) = \frac{1}{2}, Q_6(2) = \frac{4}{5}$
- What would it be for $t = 3$?

# Given an estimate of the action value $Q_t(a)$, how could we use this information?

- The **<u>greedy action</u>** optimizes the action value approximation $Q_t(a)$

$$A_t = \arg\max_a Q_t(a)$$

- This is good because it approximates the optimal action

$$a^* = \arg\max_a q_*(a)$$

  - Thus, it will tend to have better reward than the random policy
- Greedy algorithm
  - Initialize $\forall a\,,\quad Q_1(a) \leftarrow 0,\;\; n_a \leftarrow 0$
  - For $t = \{1,2, \dots, T\}$
    - Choose $A_t \leftarrow \arg\max_a Q_t(a)$
    - Receive reward $R_t \leftarrow \text{Environment}(A_t)$
    - Update $Q_{t+1}(A_t) \leftarrow \frac{\left(Q_t(A_t) \cdot n_{A_t}\right) + R_t}{n_{A_t}+1}$
    - Update $n_{A_t} \leftarrow n_{A_t} + 1$

# Greedy can be suboptimal if $Q_t$ is a a bad approximation

- However, greedy can be **bad**, if $Q_t(a)$ is **bad approximation**

$$\max_a Q_t(a) \neq \max_a q_*(a)$$

- Thus, the core explore-exploit tradeoff remains:
    - **Exploit** – Choose greedy action to maximize rewards.
    - **Explore** – Choose non-greedy action to improve estimate of $Q_t$
    - Note: The "explore" part is just about improving our understanding about the environment rather than finding new environment states because there are is **only one state** in bandits

- Can we do better than greedy?

# $\epsilon$-Greedy algorithm slightly modifies the greedy algorithm to improve exploration

- One simple idea is to randomly sample arms initially and then do greedy from then on
- A more common approach is to randomly choose between explore (via random algorithm) and exploit (via greedy algorithm)
- $\epsilon$-Greedy algorithm
  - Initialize $\forall a\,,\ Q_1(a) \leftarrow 0,\ \ n_a \leftarrow 0$
  - For $t = \{1,2, \dots, T\}$
    - With probability $\epsilon$, choose $A_t \leftarrow \text{RandomAction}()$
    - Otherwise, choose $A_t \leftarrow \arg\max_a Q_t(a)$
    - Receive reward $R_t \leftarrow \text{Environment}(A_t)$
    - Update $Q_{t+1}(A_t) \leftarrow \frac{\left(Q_t(A_t) \cdot n_{A_t}\right) + R_t}{n_{A_t} + 1}$
    - Update $n_{A_t} \leftarrow n_{A_t} + 1$

# Demo of bandit algorithms

# Non-stationary / dynamic bandits relax the assumption that the environment is only in one state

- The distribution of rewards changes over time
  - Though this doesn't necessarily mean that the actions affect the environment
- The optimal $q_*$ is dependent on time
- In practice, the estimate $Q_t$ can be updated using a gradient-like rule:

$$Q_{t+1}(A_t) \coloneqq Q_t(A_t) + \alpha[R_t - Q_t(A_t)]$$

- This turns out to be an decaying weighted average (i.e., more weight on the most recent rewards:

$$Q_{t+1}(A_t) = (1-\alpha)Q_1 + \sum_{i=1}^{t} \alpha(1-\alpha)^{t-i} R_i$$

# Contextual bandits relax the assumption that the agent can observe some clue about the environment state

- Suppose now that the environment changes (nonstationary) **AND** that the agent can observe some clue or **contextual information** about the environment
- Examples
  - Netflix images – The demographics or previous ratings of the user.
  - Best search result – The search query and user history
- Now the best action depends on this **<u>context</u>**, or more generally some observation of the **<u>environment state</u>**, denoted $S_t$
  - This is the "input" to the action-selection algorithm (like $x_i$ for supervised learning)
- One remaining assumption is that the actions do **NOT** affect the **next state**
  - Thus, there is still **no notion of planning** in contextual bandits
  - This is the last remaining assumption to relax to get the full RL problem

# Summary

- Multi-armed bandits are a simplification of RL
    - Single environment state
    - Lack of knowledge / uncertainty is the key challenge
- Bandit problems retain key unique aspects of RL including
    - **Evaluative feedback** rather than instructive feedback (as in supervised learning)
    - **Explore-exploit tradeoff** even though the environment does not change
- Bandit algorithms
    - Random
    - Greedy
    - $\epsilon$-Greedy
- Variants of bandit problems
    - **Nonstationary bandits** – Environment changes over time
    - **Contextual bandits** – Agent observes clues/context about the environment state
    - Both assume that actions do NOT affect future environment states

# Reference

- Based on the excellent RL book by Sutton and Barto
  - http://incompleteideas.net/book/the-book-2nd.html