CS4910: Deep Learning for Robotics

David Klee
klee.d@northeastern.edu

T/F, 3:25-5:05pm
Behrakis Room 204

https://www.ccs.neu.edu/home/dmklee/cs4910_s22/index.html

https://piazza.com/northeastern/spring2022/cs4910a/home

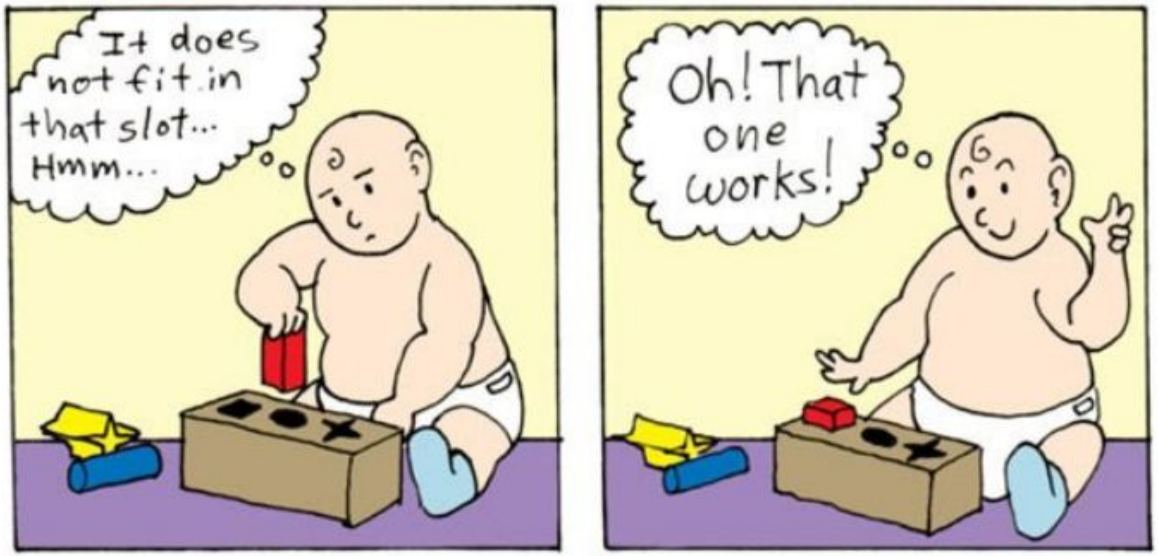
Reinforcement Learning

Some slide material taken from CS7180 F18
For more details see: Sutton & Barto 2018

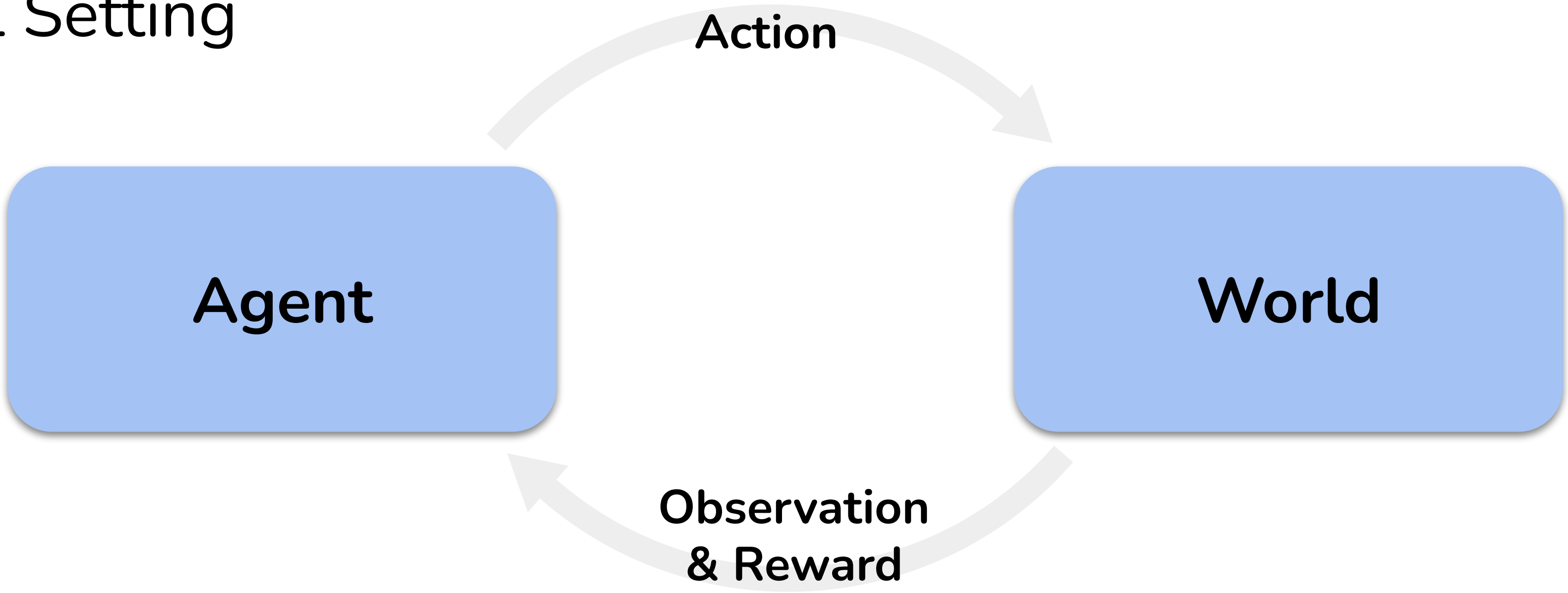
Today’s Agenda

1. HW2 Questions & Dataset
2. What is RL?
3. Bandit Problem
4. Understanding Value functions
5. Nuro arm

Reinforcement Learning (RL) is learning through trial-and-error *without a model of the world*



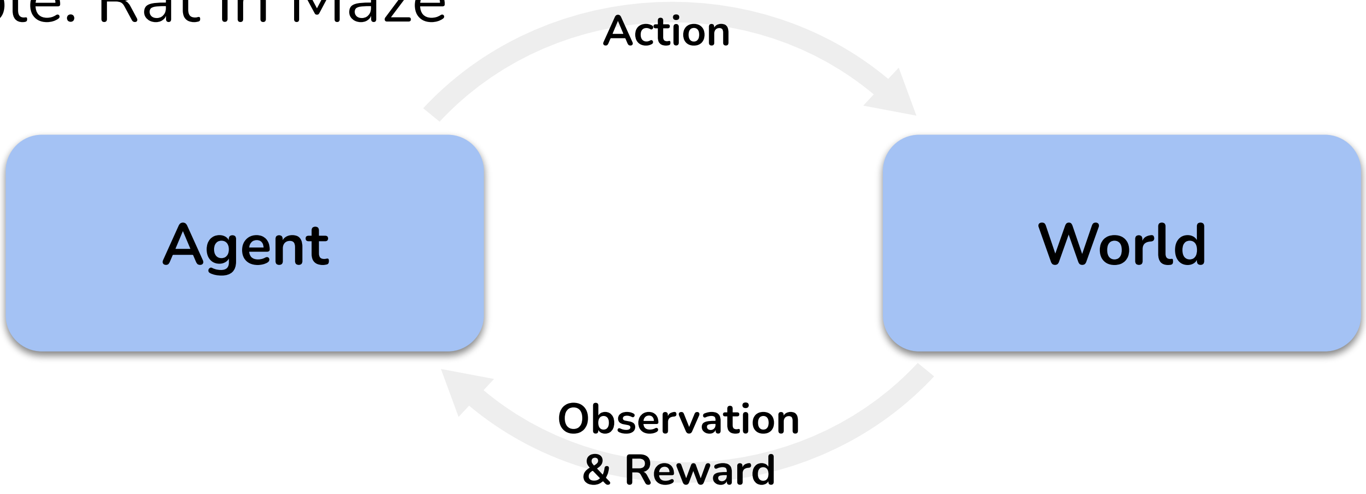
RL Setting



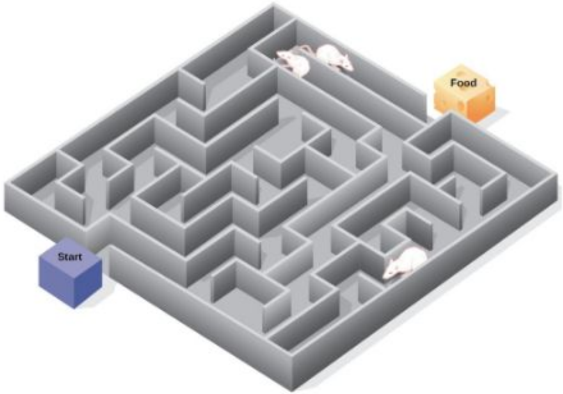
At every time step, an agent makes an observation of the world, selects action to execute, and makes note of the reward

Goal: select actions that maximize cumulative reward over time

Example: Rat in Maze



How would you instantiate the rat maze in an RL setting?



Example: Rat in Maze

Action

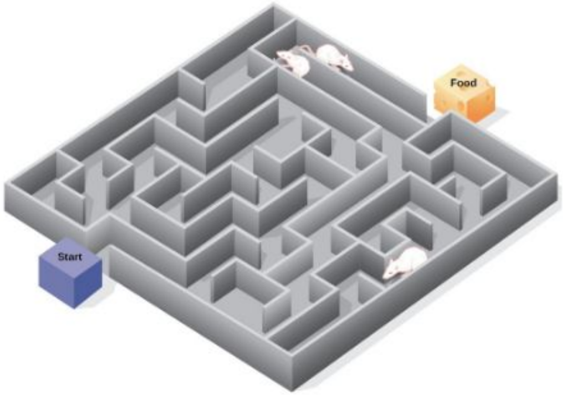
Agent

World

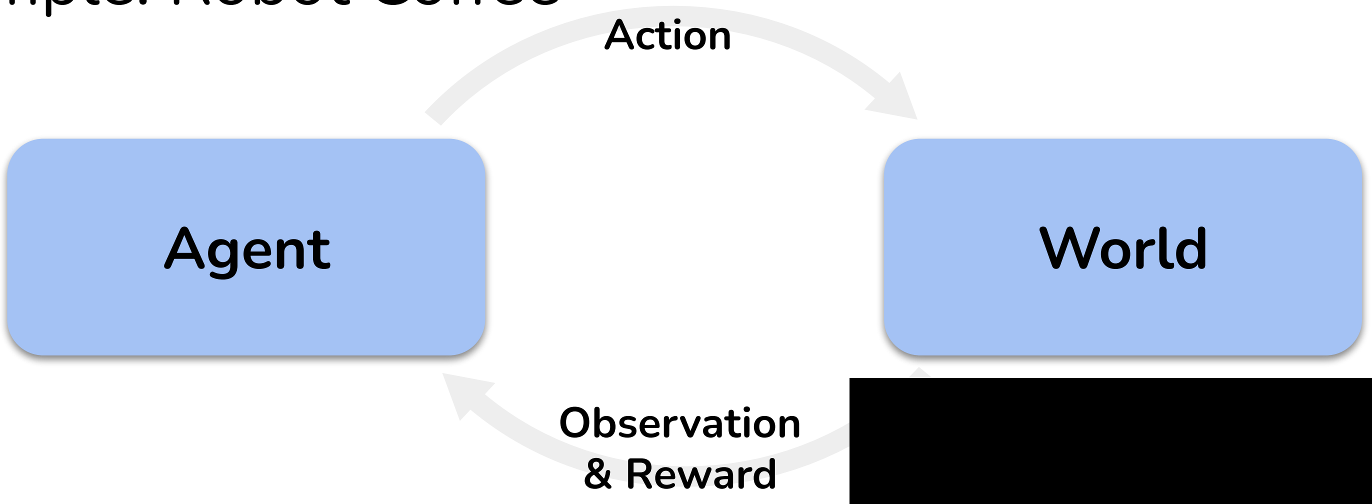
Observation
& Reward

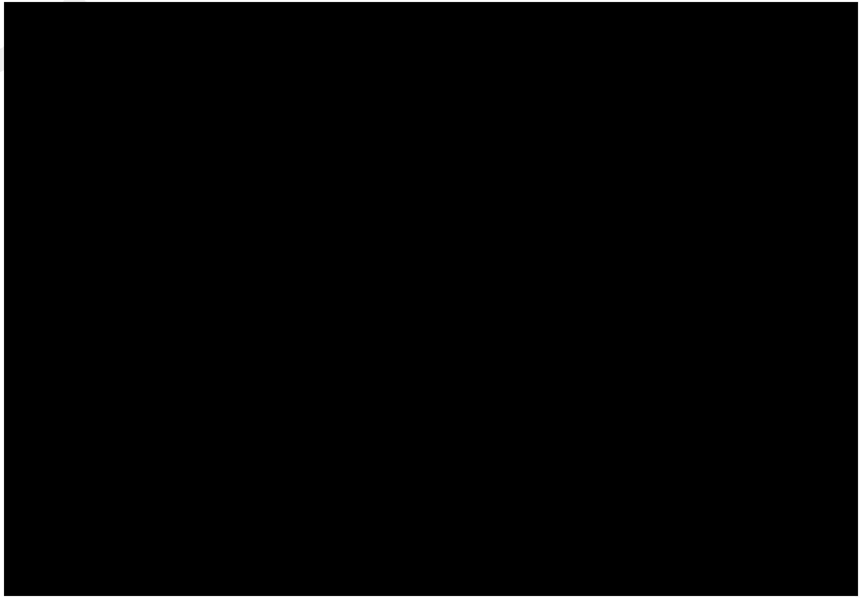
Action: move up/down/left/right
Observation: position/view/smells...
Reward: +1 if you get cheese

Goal: maximize cheese eaten



Example: Robot Coffee





How would you instantiate the robot coffee making in an RL setting?

Example: Robot Coffee

Action

Agent

World

Observation & Reward

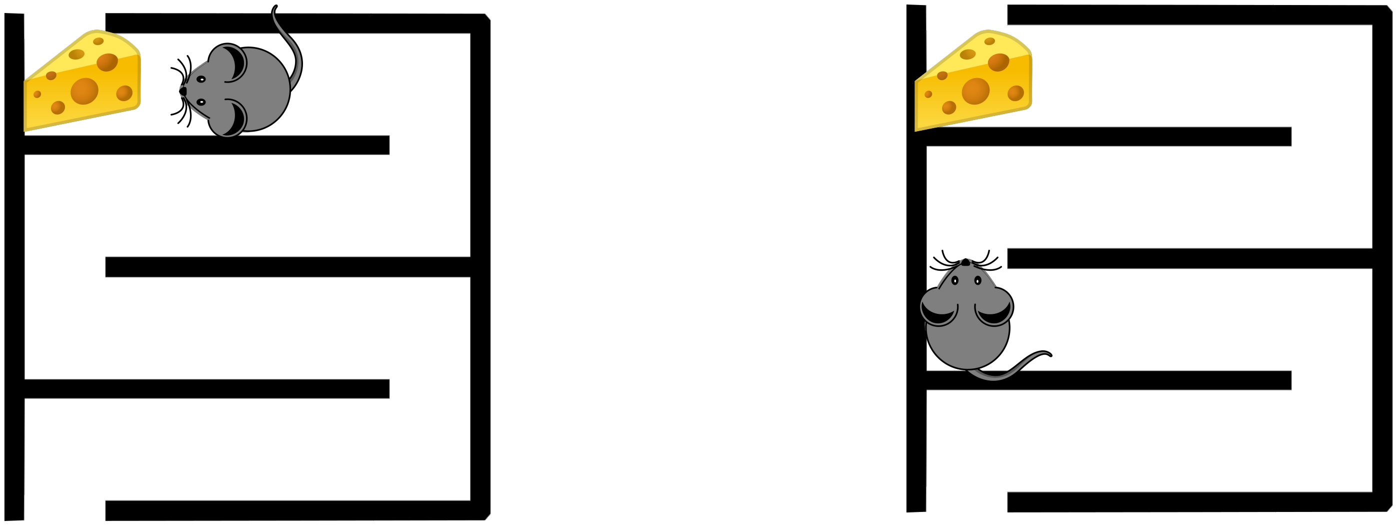
Action: move hand to position
Observation: camera image
Reward: +1 coffee made (-1 for every second coffee is not ready)

Goal: maximize coffee output

There is not a single ***best*** description of the setting. You want to consider several options, and hypothesize which will result in the preferred behavior.

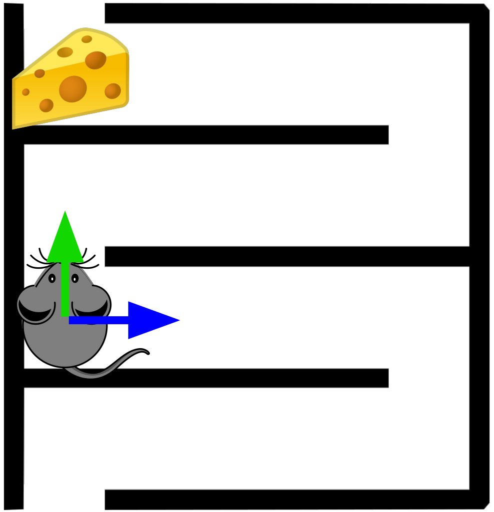
What state of the rat maze is better?

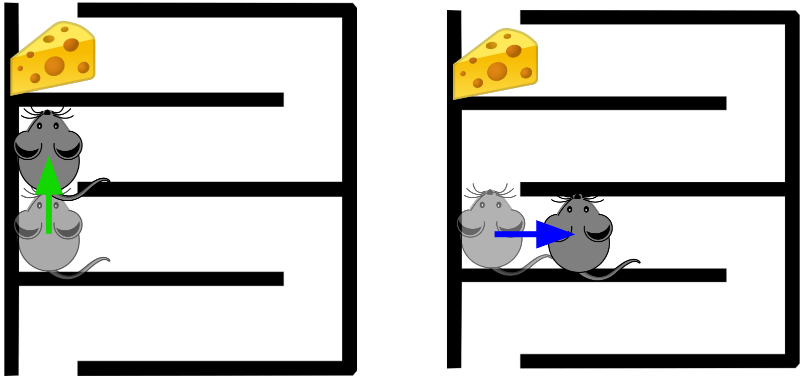
observation == state



In the language of RL, the value of the left state is greater than the value of the right state, *given that the goal is to collect cheese*

What action is better?





We can evaluate which action is better by keeping track of the value of the state that we transition to.

1-Armed Bandit Problem

State:

Action:

Goal:

25¢ to pull the lever

1% chance to win $6

20% chance to win $1

1-Armed Bandit Problem

Reward:

25¢ to pull the lever

1% chance to win $6

20% chance to win $1

What is the expected payout (i.e. value) of pulling the lever?

$$\mathbb{E}[\text{payout}] = \sum_{r \in R} r \cdot p(r)$$

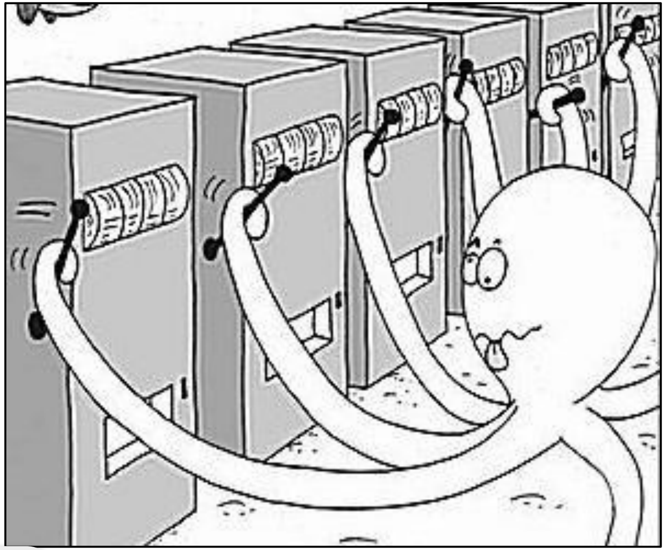
$$\mathbb{E}[\text{payout}] = 1.0 \times -0.25 + 0.01 \times 6 + 0.2 \times 1 = 0.01$$

k-Armed Bandit

State: sitting in front of slot machines

Action: pull one of the *k* levers

Reward: payout of the *selected* machine

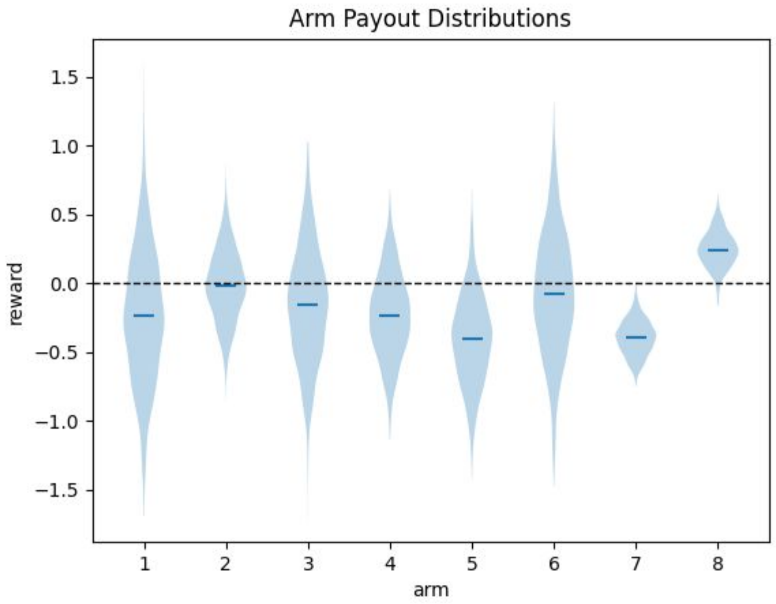


What action will result in the most reward over time?

Problem: expected payout for each machine is unknown

k-Armed Bandit

python examples/k_armed_bandit.py



Determining the value of each action

$$
\begin{aligned}
Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\Big(R_n + (n-1)Q_n\Big) \\
&= \frac{1}{n}\Big(R_n + nQ_n - Q_n\Big) \\
&= Q_n + \frac{1}{n}\Big[R_n - Q_n\Big],
\end{aligned}
$$

SB, eq 2.3

Implement this in Gambler.update_value

How should we select actions?

Exploration **vs.** **Exploitation**

How would we implement pure exploration or pure exploitation?

Epsilon-greedy (ε-greedy) action selection

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Repeat forever:

$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$

$R \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

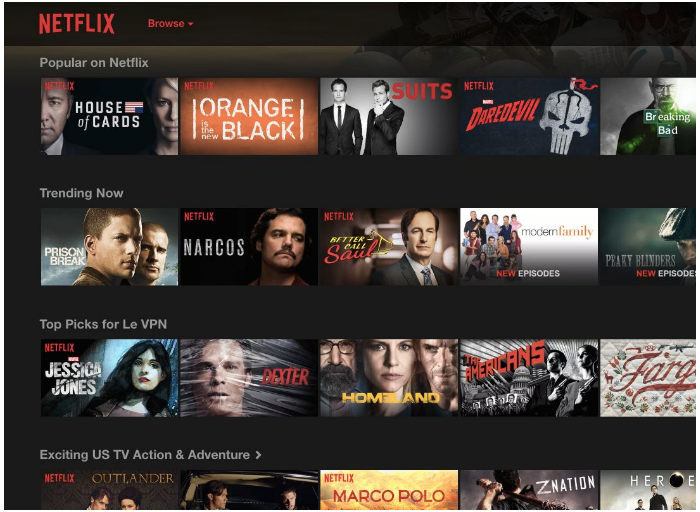
Other approaches to action selection

Optimistic Initialization 🙂 : $Q_o(a) = +q_{init}$

Upper-confidence bound (UCB)

$$A_t \doteq \underset{a}{\arg\max} \left[Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right]$$

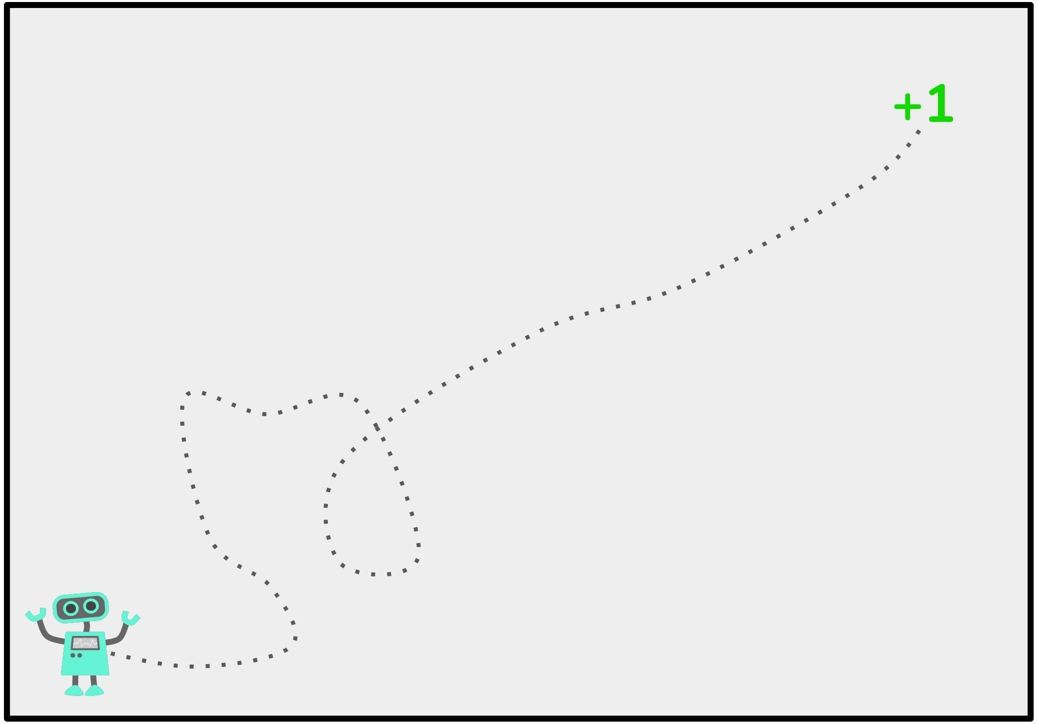
K-Armed bandit: wider applications



Recommendation algorithms

Grasping (think of each pixel as an arm of the bandit, learn which pixel results in grasp)

The credit assignment problem: how much is an action responsible for future rewards



Next Class

TD-Learning: how to update values based on multi-timestep trajectories

Q-learning: how to learn the value of actions on multi-timestep trajectories

How to extend ideas to context of deep learning…

Survey to provide feedback

https://forms.gle/fRhsy2JMUrThc4jN6

Do you feel ready to begin working on a project?