Reinforcement Learning

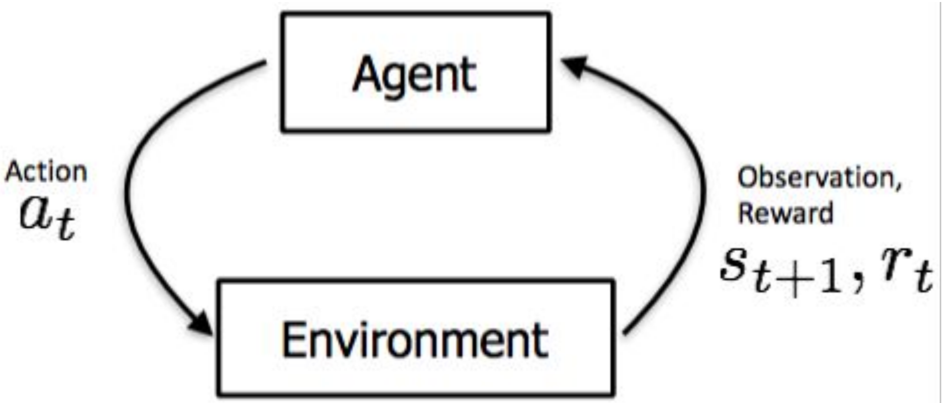
VS265 - Neural Computation, 2018

What we have covered

Passive Learning

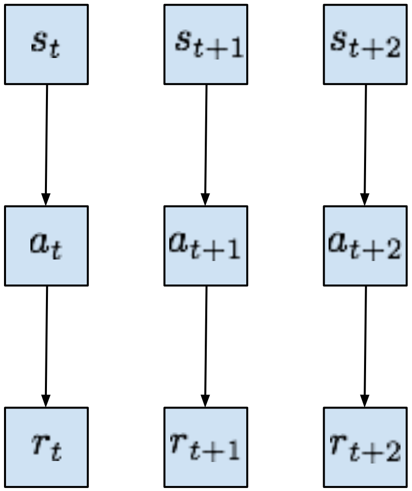
Today:Active Learning (RL)

What is Reinforcement Learning?

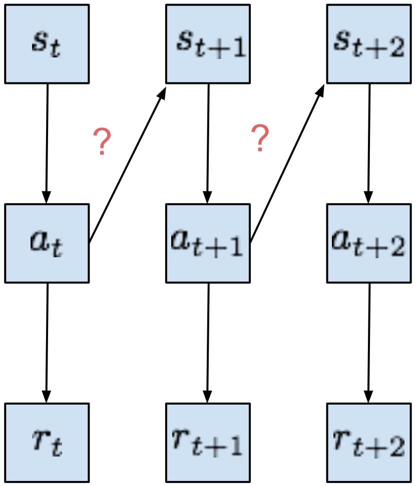


How is it different than other models?

Passive Learning



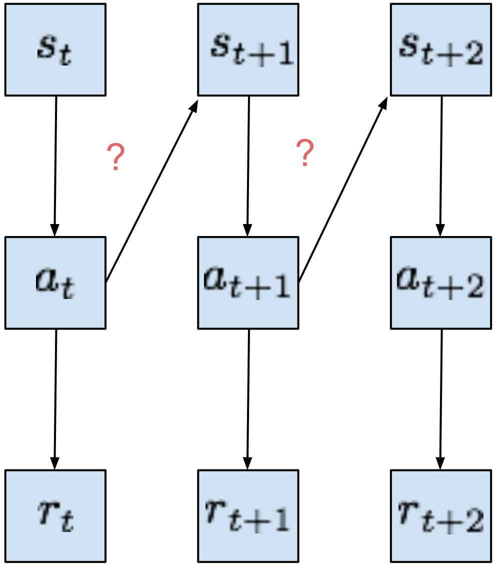
Active Learning (RL)



Why is this hard?

- Actions affect future data
- Rewards are sparse
- Feedback is delayed

Reinforcement Learning



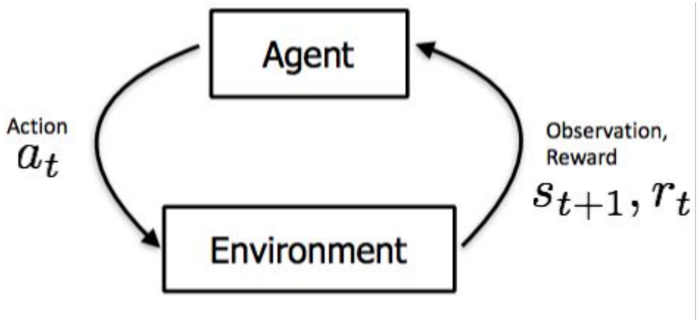
Outline

- Markov Decision Processes (MDPs)
- How to maximize reward (Q-Learning)
- Connection to neurons in the Ventral Tegmental Area (VTA)
- How to learn in large, unstructured**, environments
- Open Questions

Markov Decision Process

Markov Decision Process (MDP)

An MDP fully describes an Environment:



- S: State Space
- A: Action Space
- P: Transition Kernel - $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$
- R: Reward Function - $r_t = R(s_t, a_t)$

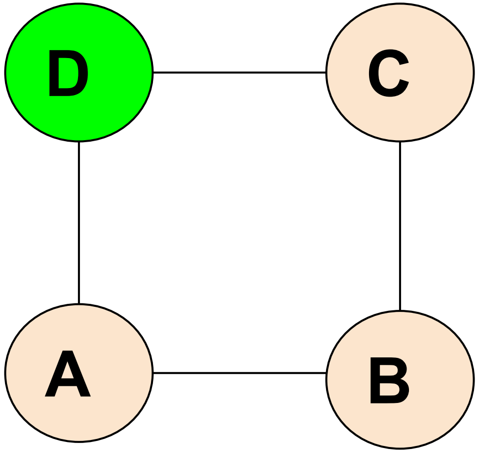
Markov Decision Process (MDP)

- Markov
  - $p(s_{t+1}|s_0 \ldots s_t, a_0 \ldots a_t) = p(s_{t+1}|s_t, a_t)$
- Decision
  - Decide on an action at each time point
  - $a_t \in A$
- Process
  - States evolve over time

    $s_{t+1} \sim p(s_{t+1}|s_t, a_t)$

Markov Decision Process (MDP)

$$S = \{A, B, C, D\}$$
$$A = \{Up, Down, Left, Right\}$$
$$R = \{0, 0, 0, 1\}$$



Q-Learning - Algorithm

Q-Learning - Algorithm

- Find a good **policy,** $\pi : S \rightarrow A$, that maximizes the expected sum of rewards over time:

$$a_t = \pi(s_t)$$

$$s_{t+1} \sim P(s_{t+1}|s_t, a_t)$$

$$r_t = R(s_t, a_t)$$

$$\pi^* = \arg\max_{\pi} \; \mathbb{E}\Big[\sum_{\tau=0}^{\infty} \gamma^{\tau} r_{t+\tau}\Big]$$

Q-Learning - Algorithm

- **Q(s,a)** is the total expected reward starting from state s, taking action a, and then following optimal policy

$$a_t = \pi(s_t) = \underset{a}{argmax} \;\; Q_\pi(s_t, a)$$

Q-Learning - Update Rule

Q-Learning - Update Rule

$$\pi^* = \arg\max_{\pi} \ \mathbb{E}\left[\sum_{\tau=0}^{\infty} \gamma^{\tau} r_{t+\tau}\right]$$

Q-Learning - Update Rule

State-Value Function:

$$V_\pi(s_t) = \mathbb{E}\Big[\sum_{\tau=0}^{\infty} \gamma^\tau r_{t+\tau}\Big]$$

$$V_\pi(s_t) = \mathbb{E}\Big[r_t + \sum_{\tau=1}^{\infty} \gamma^\tau r_{t+\tau}\Big]$$

$$V_\pi(s_t) = \mathbb{E}\Big[r_t\Big] + \mathbb{E}\Big[\sum_{\tau=1}^{\infty} \gamma^\tau r_{t+\tau}\Big]$$

$$V_\pi(s_t) = \mathbb{E}\Big[r_t\Big] + \gamma V_\pi(s_{t+1})$$

Q-Learning - Update Rule

Action-Value Function:

$$V_\pi(s_t) = \mathbb{E}\Big[r_t\Big] + \gamma V_\pi(s_{t+1}), \quad r_t = R(s_t, \pi(s_t))$$

$$Q_\pi(s_t, a_t) = \mathbb{E}\Big[r_t\Big] + \gamma V_\pi(s_{t+1}), \quad r_t = R(s_t, a_t)$$

$$a_t = \pi(s_t) = \underset{a}{argmax} \;\; Q_\pi(s_t, a)$$

$$V_\pi(s_t) = \underset{a}{max} \;\; Q(s_t, a)$$

$$Q_\pi(s_t, a_t) = \mathbb{E}\Big[r_t\Big] + \gamma \underset{a}{max} \;\; Q_\pi(s_{t+1}, a), \quad r_t = R(s_t, a_t)$$

Q-Learning - Update Rule

$$Q_\pi(s_t, a_t) = \mathbb{E}\Big[r_t\Big] + \gamma \underset{a}{max} \;\; Q_\pi(s_{t+1}, a), \quad r_t = R(s_t, a_t)$$

$$a_t = \arg\max_a Q(s_t, a)$$

Q-Learning - Update Rule

$$\boxed{Q_\pi(s_t, a_t) = \mathbb{E}\Big[r_t\Big] + \gamma \max_a \; Q_\pi(s_{t+1}, a), \quad r_t = R(s_t, a_t)}$$

$Q(s_t, a_t) = 0$ // Initialize action-value beliefs to 0

Iterate:

$$a_t = \underset{a}{argmax} \; Q_\pi(s_t, a) \qquad r_t = R(s_t, a_t) \qquad s_{t+1} \sim p(s_{t+1} | s_t, a_t)$$

$$\Delta Q_\pi(s_t, a_t) = (\text{New Belief}) - (\text{Old Belief})$$

$$\underbrace{\Delta Q_\pi(s_t, a_t)}_{\text{Temporal Difference}} = \underbrace{r_t + \gamma \max_a Q_\pi(s_{t+1}, a)}_{\text{Critic (New Belief)}} - \underbrace{Q_\pi(s_t, a_t)}_{\text{Belief}}$$

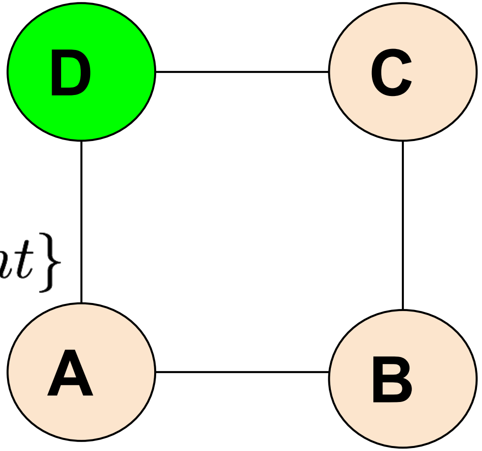
Q-Learning - Exercise

Q-Learning (Exercise)

$$\underbrace{\Delta Q(s_t, a_t)}_{\text{Temporal Difference}} = [r_t] + \max_a Q(s_{t+1}, a) - Q(s_t, a_t)$$

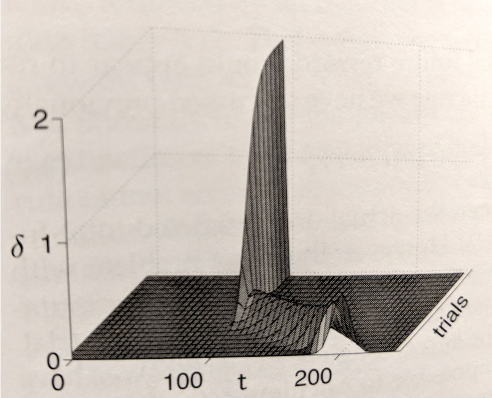
$$Q_\pi(s_t, a_t) \mathrel{+}= \eta \Delta Q_\pi(s_t, a_t)$$

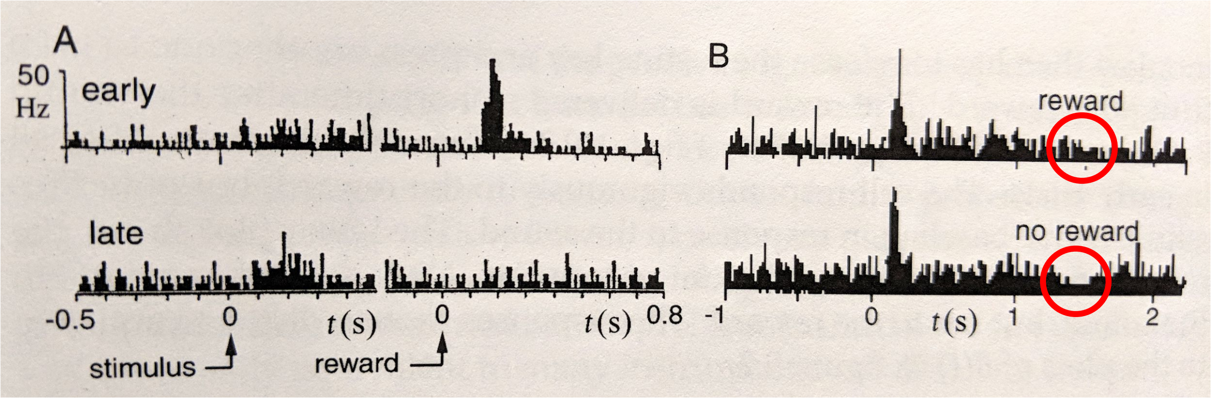
$$S = \{A, B, C, D\}$$
$$A = \{Up, Down, Left, Right\}$$
$$R = \{0, 0, 0, 1\}$$



Connection to VTA

Connection to VTA



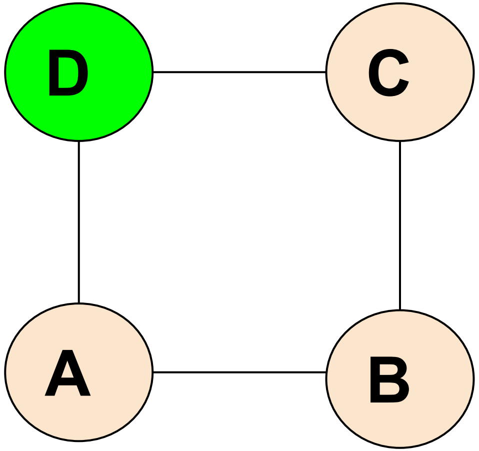


Theoretical Neuroscience, ch.9 (Dayan & Abbot)
(Adapted from Mirenowicz & Schultz, ‘94 & Schultz ‘98)

Q-Learning in large environments

Q-Learning in large environments

$$|S| \geq 2^{100}$$



$$|S| = 4$$

Q-Learning in large environments

- Deep Q-Networks (DQN): Estimate Q using a neural network

$$Q_\theta(s, a) \approx Q(s, a)$$

Q-Learning in large environments

- Objective Function: Use the temporal difference signal

$$E = [r_t + \gamma \max_a Q_\theta(s_{t+1}, a)] - Q_\theta(s_t, a_t)$$

$$\theta = \theta + \eta \frac{\partial E}{\partial \theta}$$

Deep-Q-Network

- Use a Convolutional Neural Network (CNN) as the function approximator
- **Experience Replay -** Store experiences in a data-set and randomly sample them during learning

$$e_t = (s_t, a_t, r_t, s_{t+1})$$
$$\mathbb{D} = e_1, ..., e_N$$

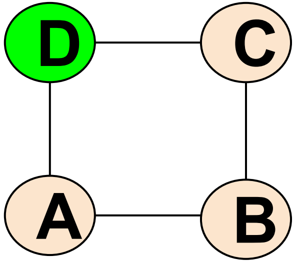
Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).

Open Questions

Open Questions

- Credit assignment in worlds with sparse rewards
- Exploration vs. Exploitation
- Generalization to the real world
- Continual Learning

Q-Learning in even more complex worlds



Resources

- David Silver's Lectures
  - http://www0.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html
- CS294 - Deep Reinforcement Learning
  - http://rll.berkeley.edu/deeprlcourse/

Q-Learning

$$\underbrace{\Delta Q(s_t, a_t)}_{\text{Temporal Difference}} = [r_t] + \max_a Q(s_{t+1}, a) - Q(s_t, a_t)$$

$$Q_\pi(s_t, a_t) \mathrel{+}= \eta \Delta Q_\pi(s_t, a_t)$$

$$S = \{A, B, C, D\}$$

$$A = \{Up, Down, Left, Right\}$$

$$R = \{0, 0, 0, 1\}$$