CS686: Reinforcement Learning

Sung-Eui Yoon
(윤성의)

Course URL:
http://sgvr.kaist.ac.kr/~sungeui/MPA

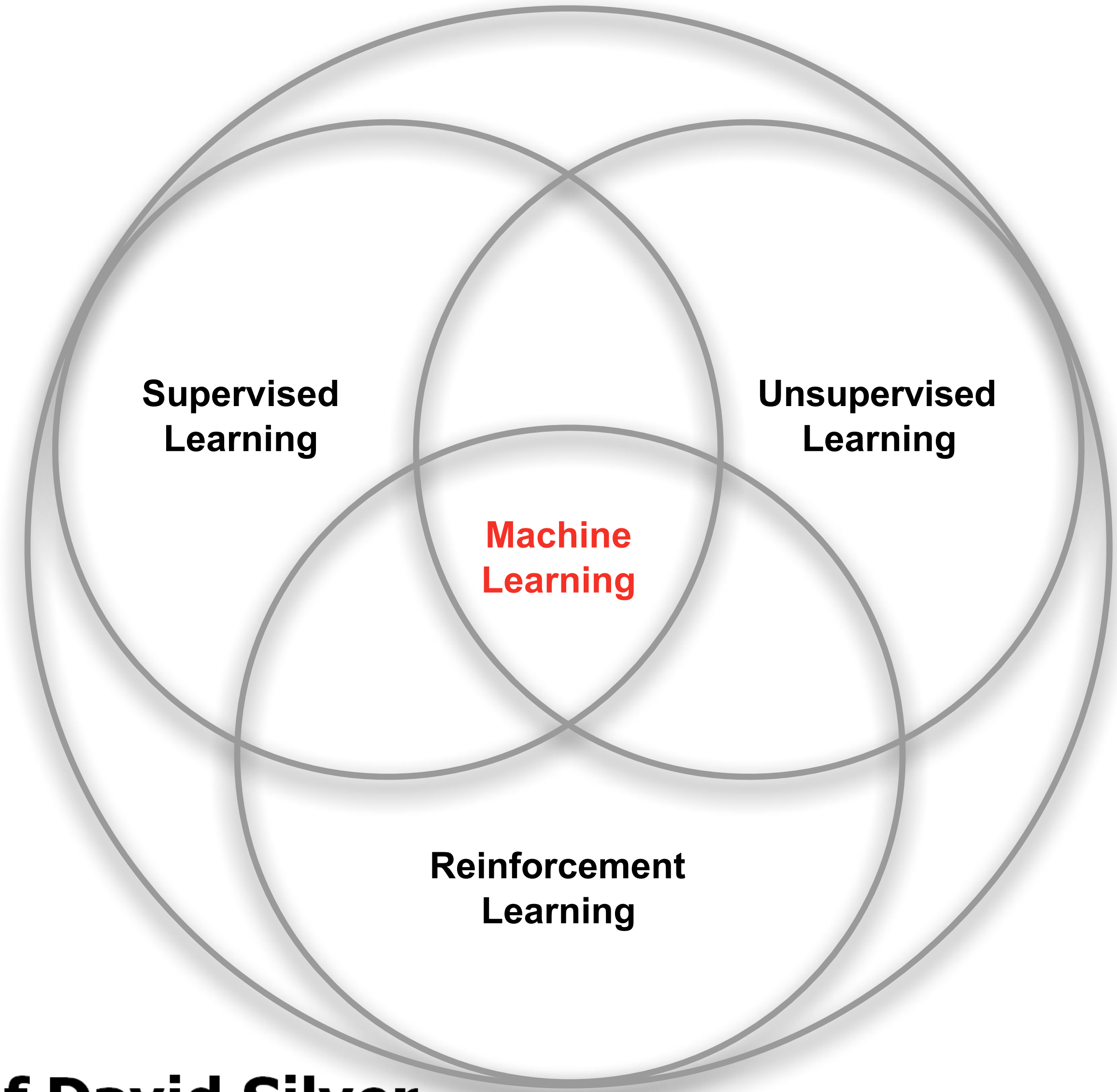
KAIST

Class Objectives

- Discuss basic concepts of reinforcement learning

- Last time:
  - RRT techniques and kinodynamic planner

Branches of Machine Learning



 Ack: slides of David Silver

Characteristics of Reinforcement Learning

- **What makes reinforcement learning different from other machine learning paradigms?**
  - **There is no supervisor, only a reward signal**
  - **Feedback is delayed, not instantaneous**
  - **Time really matters (sequential, non i.i.d data)**
  - **Agent's actions affect the subsequent data it receives**

Examples of Reinforcement Learning

- **Fly stunt maneuvers in a helicopter**
- **Make a humanoid robot walk**
- **Manage an investment portfolio**
- **Play many different Atari games better than humans**

Rewards

- **A reward R_t is a scalar feedback signal**
- **Indicates how well agent is doing at step t**
- **The agent's job is to maximize cumulative reward**

Reinforcement learning is based on the reward hypothesis

Definition (Reward Hypothesis)
All goals can be described by the maximization of expected cumulative reward

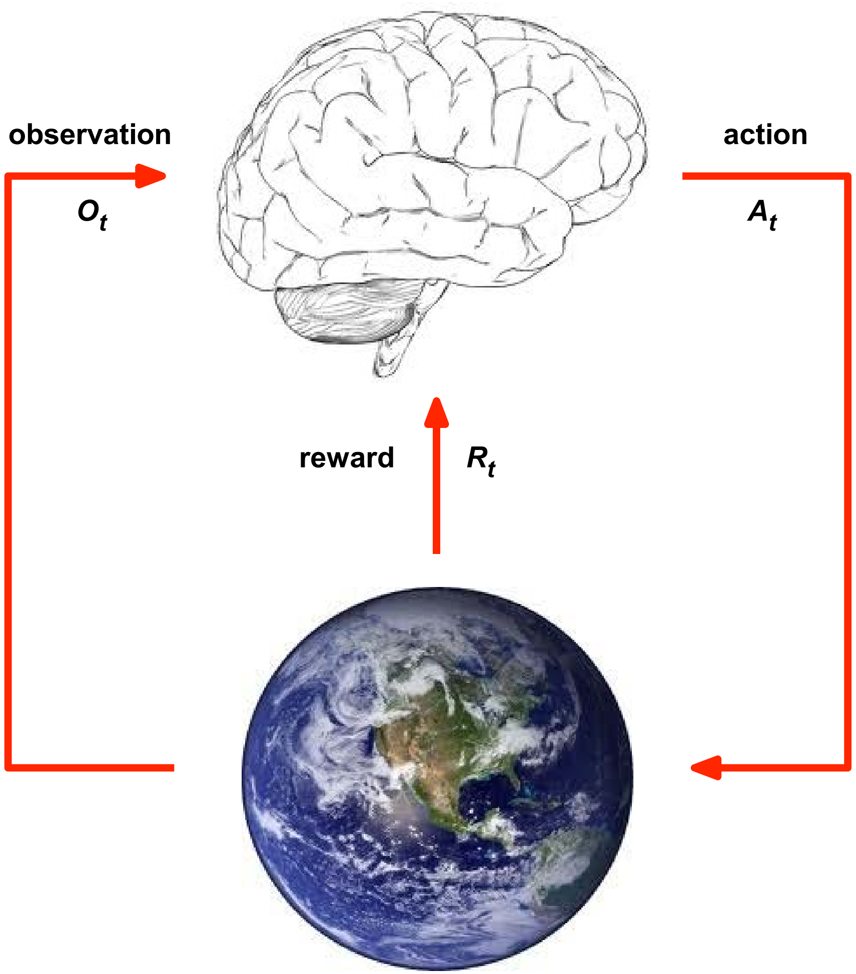
Examples of Rewards

- Fly stunt maneuvers in a helicopter
  - + reward for following desired trajectory
  - – reward for crashing
- Make a humanoid robot walk
  - + reward for forward motion
  - – reward for falling over
- Manage an investment portfolio
  - + reward for each $ in bank

Sequential Decision Making

- **Goal**
  - **Select actions to maximize total future reward**
- **Actions may have long term consequences**
  - **Reward may be delayed**
  - **It may be better to sacrifice immediate reward to gain more long-term reward**
- **Examples:**
  - **Refueling a helicopter (might prevent a crash in several hours)**
  - **Blocking opponent moves (might help winning chances many moves from now)**

Agent and Environment



- **At each step t, the agent:**
  - **Receives observation O_t**
  - **Receives scalar reward R_t**
  - **Executes action A_t**
- **The environment:**
  - **Receives action A_t**
  - **Emits observation O_{t+1}**
  - **Emits scalar reward R_{t+1}**
- **t increments at env. step**

History and State

- **The history is the sequence of observations, actions, rewards**

$$H_t = O_1, R_1, A_1, \ldots, A_{t-1}, O_t, R_t$$

- **What happens next depends on the history:**
  - **The agent selects actions**
  - **The environment selects observations/rewards**
- **State is the information used to determine what happens next**
- **Formally, state is a function of the history:**

$$S_t = f(H_t)$$

Information State

- **An information state (a.k.a. Markov state) contains all useful information from the history**

Definition

A state S_t is Markov if and only if

$$P[S_{t+1} \mid S_t] = P[S_{t+1} \mid S_1, ..., S_t]$$

  - **"The future is independent of the past given the present"**
  - **Once the state is known, the history may be thrown away**

Major Components of an RL Agent

- An RL agent may include one or more of these components:
  - Policy: agent's behavior function
  - Value function: how good is each state and/or action
  - Model: agent's representation of the environment

Policy

- **A policy is the agent's behavior**
  - **A map from state to action, e.g.**
- **Deterministic policy:** $a = \pi(s)$
- **Stochastic policy:** $\pi(a|s) = \mathbb{P}[A_t = a | S_t = s]$

Value Function

- **Value function is a prediction of future reward**
  - **Used to evaluate the goodness/badness of states, and thus to select between actions, e.g.**

$$v_\pi(s) = \mathbb{E}_\pi \left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots \mid S_t = s \right]$$

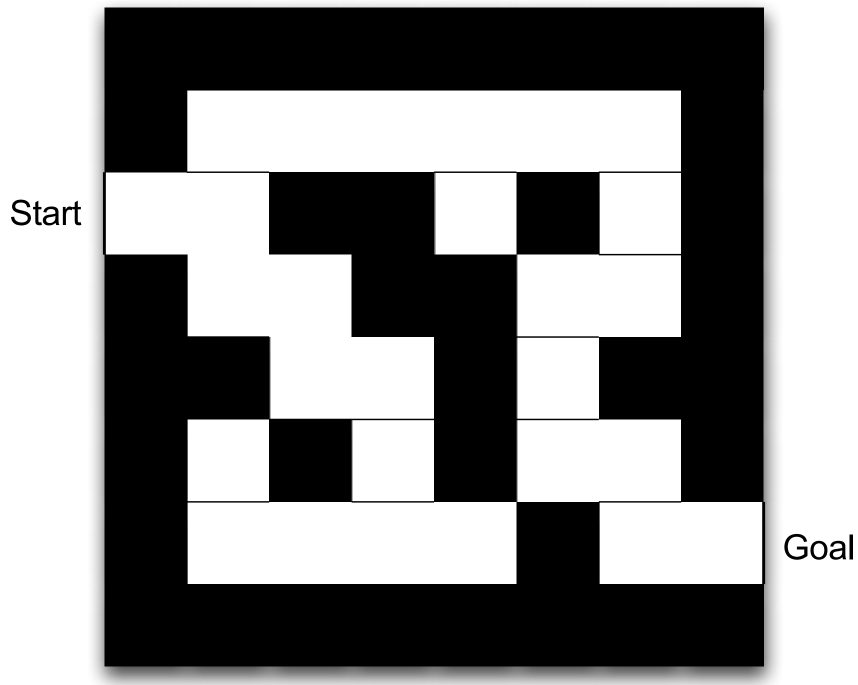
Playing Atari with Deep Reinforcement Learning

Model

- **A model predicts what the environment will do next**
  - **P predicts the next state**
  - **R predicts the next (immediate) reward, e.g.**

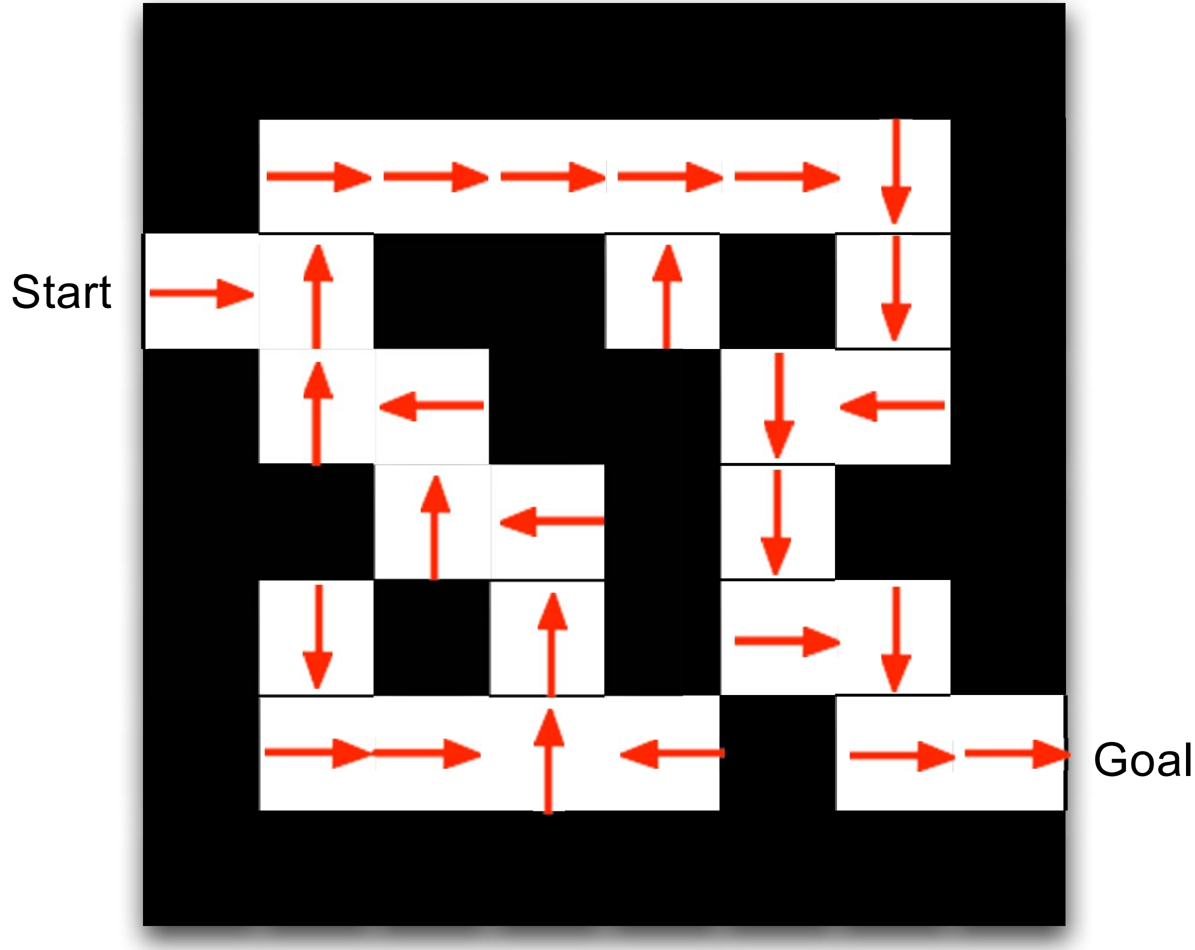
$$\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' \mid S_t = s, A_t = a]$$
$$\mathcal{R}_{s}^{a} = \mathbb{E}\left[R_{t+1} \mid S_t = s, A_t = a\right]$$

Maze Example



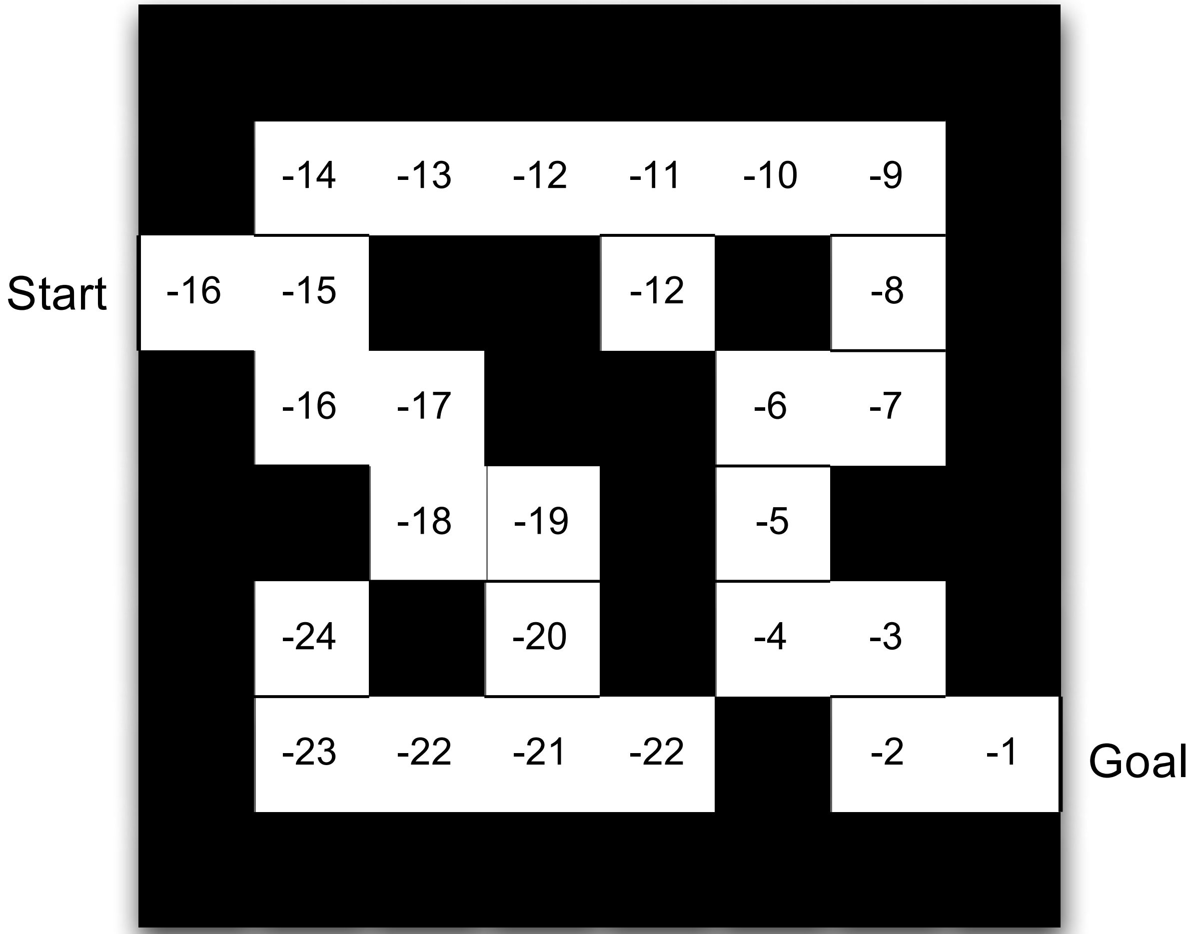
- **Rewards: -1 per time-step**
- **Actions: N, E, S, W**
- **States: Agent's location**

Maze Example: Policy



- **Arrows represent policy π(s) for each state s**

Maze Example: Value Function



- **Numbers represent value $v_\pi(s)$ of each state s**

Action-Value Function: Q-function

- **Expected return starting from state s, taking action A and then following policy with γ as the discounting factor.**

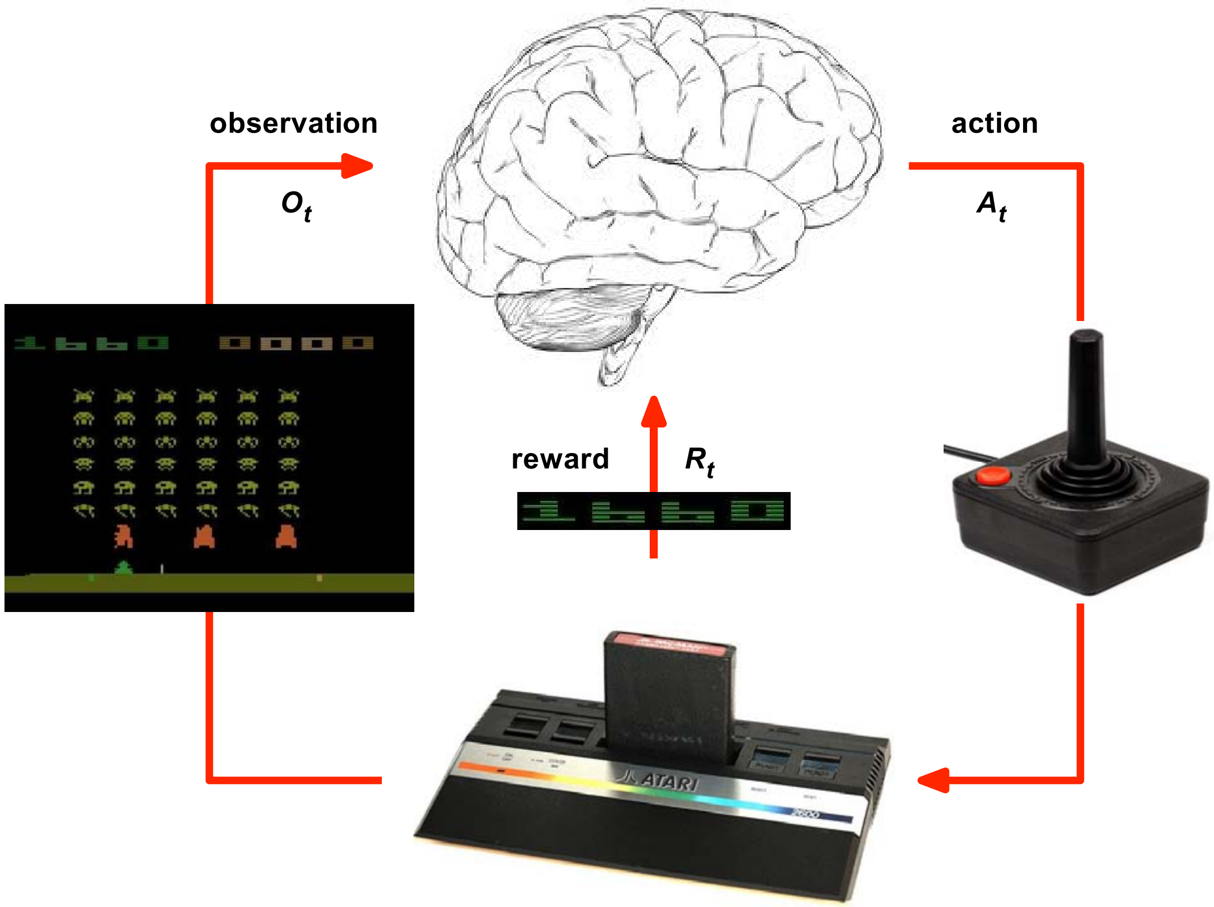
$$Q^{\pi}(s, a) = \mathbb{E}\left[r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots \mid s, a\right]$$

- **Goodness of state given an action a**

Learning and Planning

- Two fundamental problems in sequential decision making:
- Reinforcement Learning:
  - The environment is initially unknown
  - The agent interacts with the environment
  - The agent improves its policy
- Planning:
  - A model of the environment is known
  - The agent performs computations with its model (without any external interaction)
  - The agent improves its policy
  - a.k.a. deliberation, reasoning, introspection, pondering, thought, search

Learning and Planning



- Rules of the game are unknown
- Learn directly from interactive game-play
- Pick actions on joystick, see pixels and scores

Exploration and Exploitation (1)

- **Reinforcement learning is like trial-and-error learning**
  - **The agent should discover a good policy from its experiences of the environment without losing too much reward along the way**

Exploration and Exploitation (2)

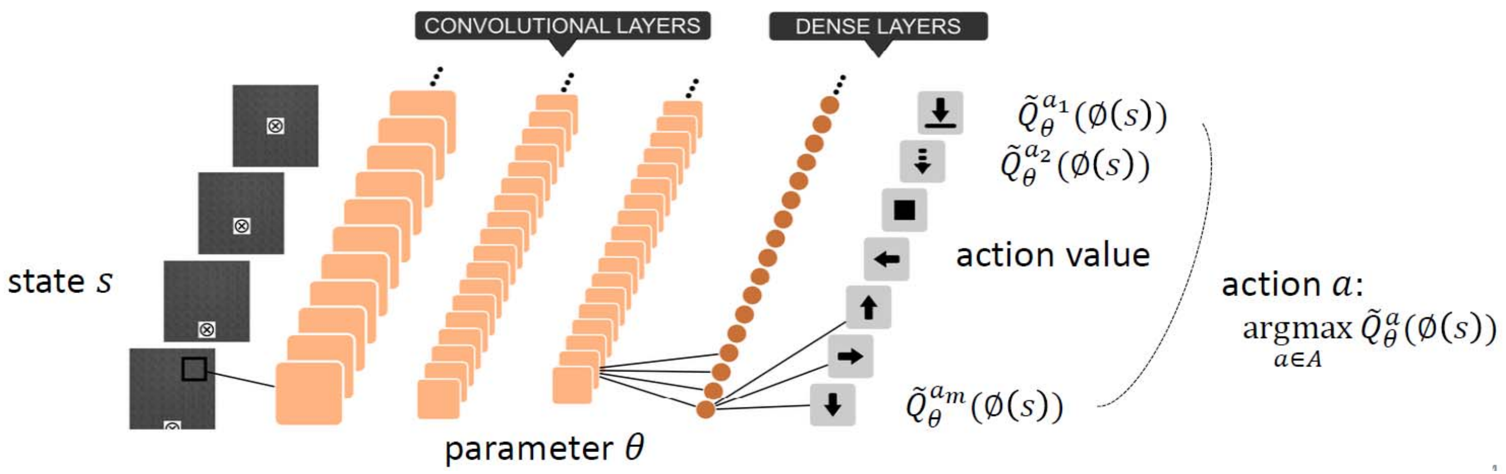
- Exploration finds more information about the environment
- Exploitation exploits known information to maximize reward
- It is usually important to explore as well as exploit

- Example of game playing
  - Exploitation: Play the move you believe is best
  - Exploration: Play an experimental move

DQN: Deep Q-Network

- DQN = Q-learning + Deep Network
  - Stabilize training with experience replay: store experience in a buffer and randomly sample them, to break the correlation between consecutive samples
  - End-to-end RL approach, flexible

$$Q(s,a) \approx \tilde{Q}^{a}_{\theta}(\emptyset(s)), \forall s \in S, \forall a \in A$$



video

Class Objectives were:

- **Discuss basic concepts of reinforcement learning**

- **Detailed lectures on the topic:**
  - **https://www.davidsilver.uk/teaching/**

No More HWs on:

- Paper summary and questions submissions

- Instead:
  - Focus on your paper presentation and project progress!