Karlsruhe Institute of Technology

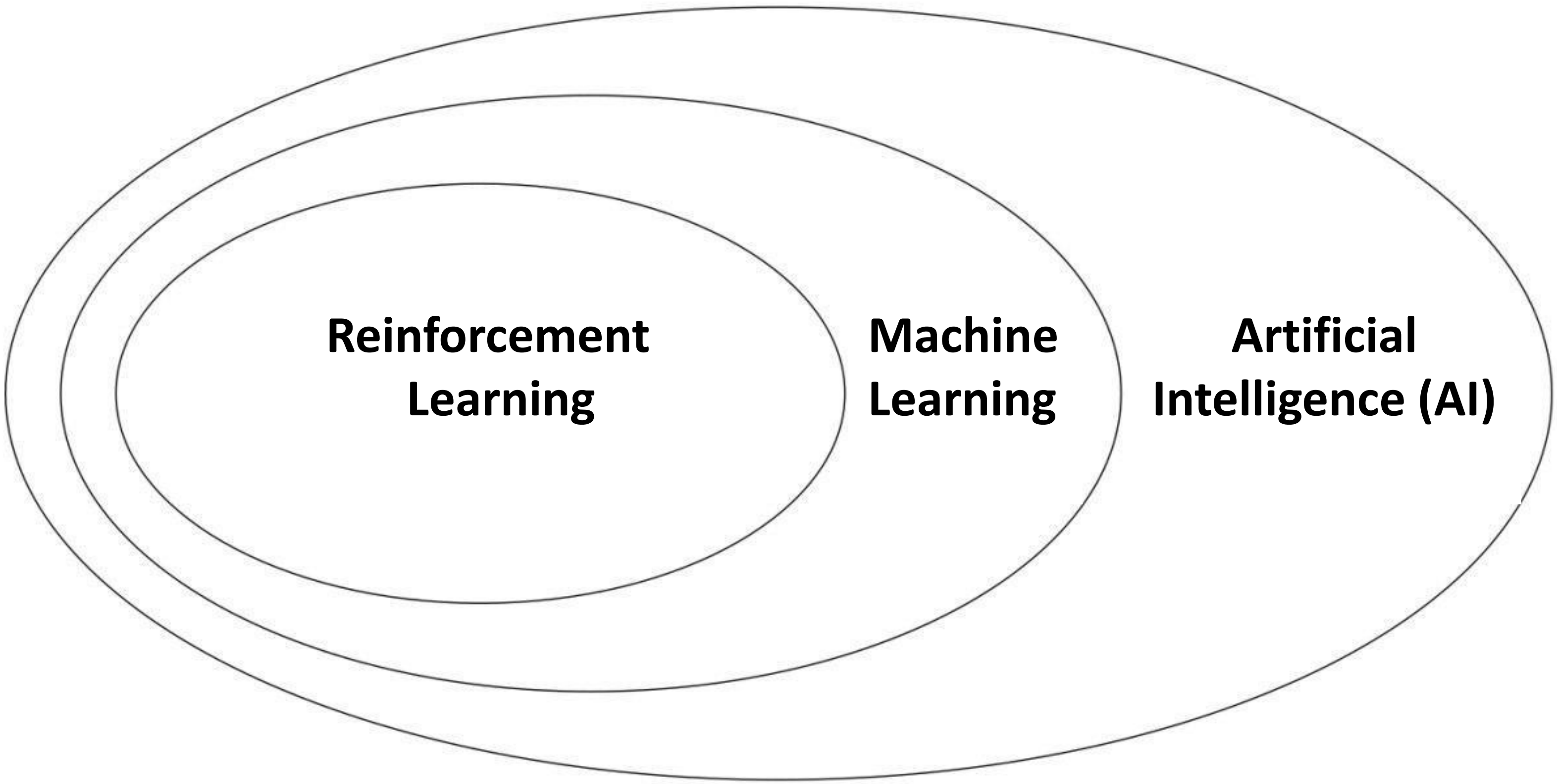
Introduction to Reinforcement Learning

Anis Farshian Abbasi

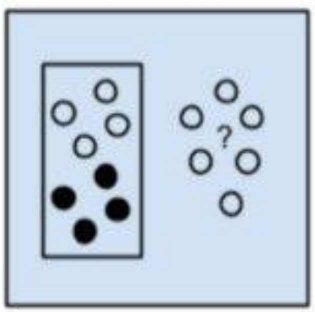
January 2020

Karlsruhe Institute of Technology , SCC

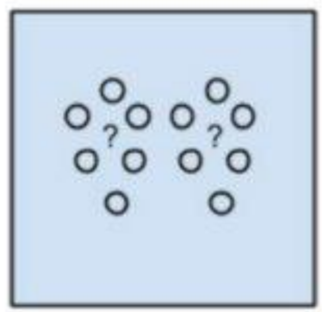
Terminology



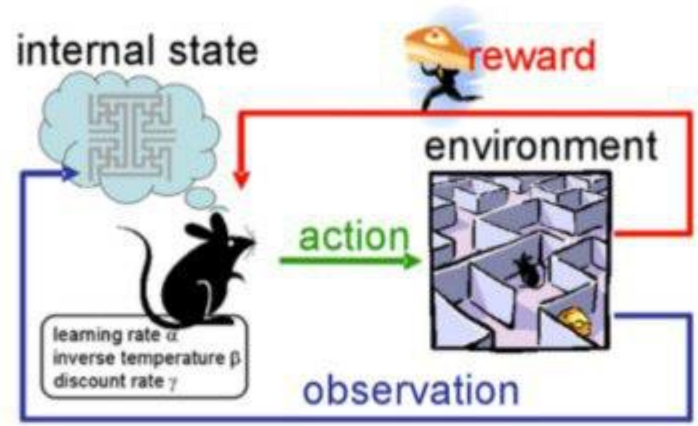
Learning Approaches



Supervised Learning



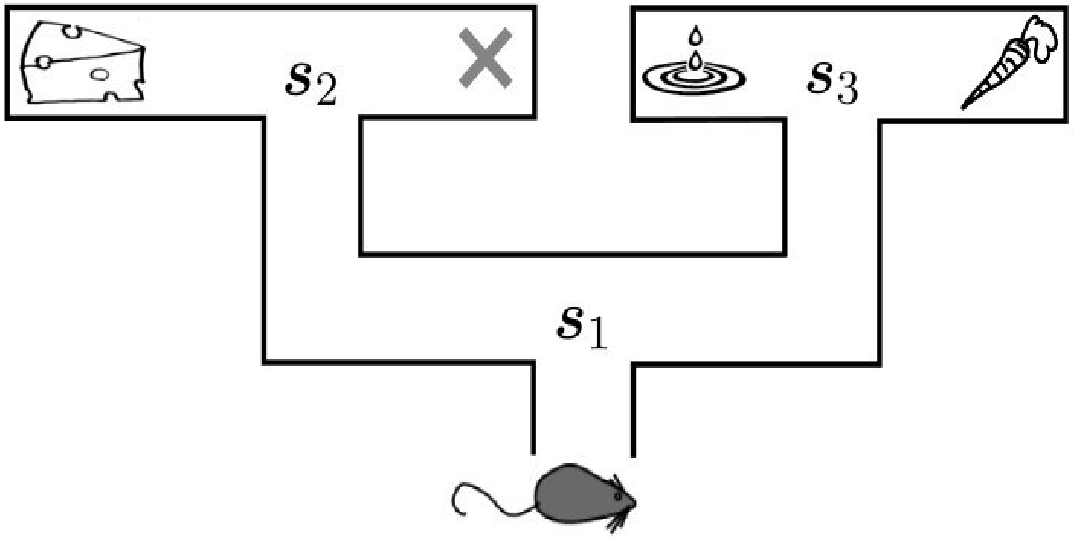
Unsupervised Learning



Reinforcement Learning

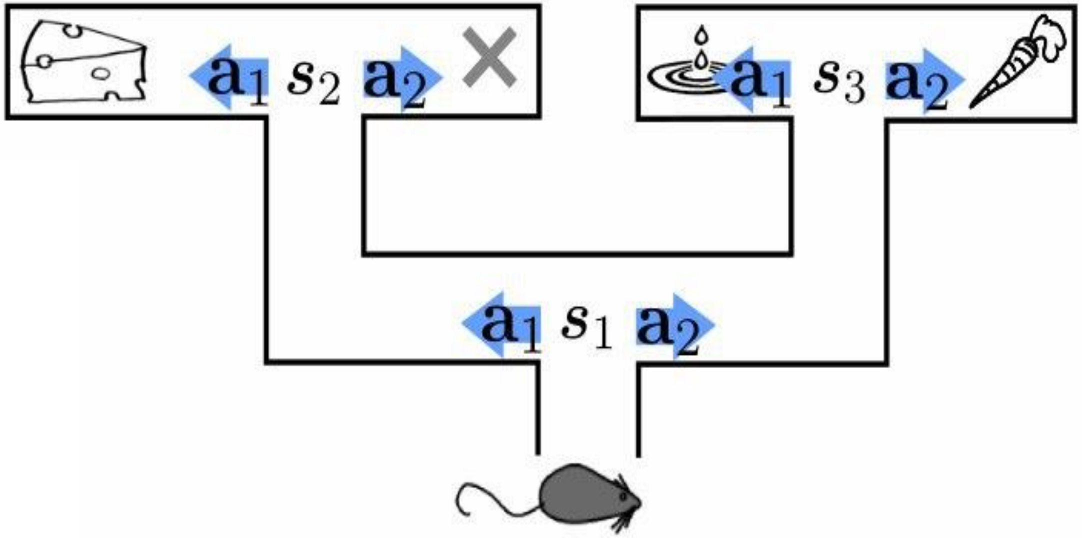
What is RL?

- RL setup: agent interacts with the environment
  - states $s \in \mathcal{S}$
  - actions $a \in \mathcal{A}$
  - rewards $r(s, a) \in \mathbb{R}$



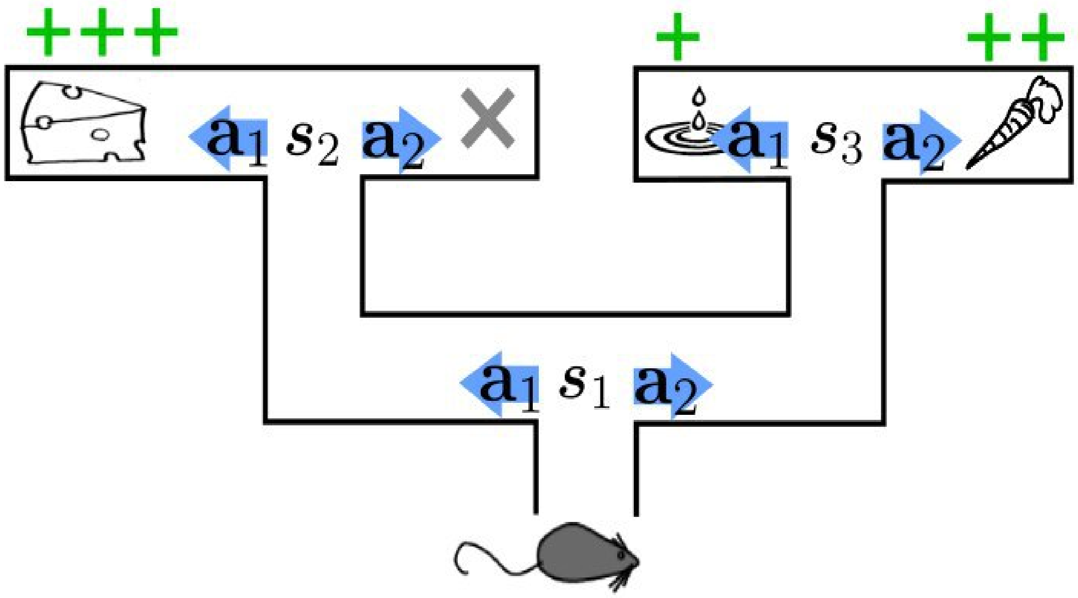
What is RL?

- RL setup: agent interacts with the environment
  - states $s \in \mathcal{S}$
  - actions $a \in \mathcal{A}$
  - rewards $r(s, a) \in \mathbb{R}$



What is RL?

- RL setup: agent interacts with the environment
  - states $s \in \mathcal{S}$
  - actions $a \in \mathcal{A}$
  - rewards $r(s, a) \in \mathbb{R}$



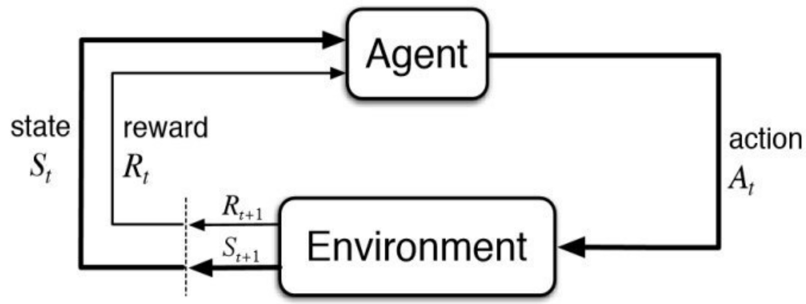
Environment and Actions

- Fully Observable (Chess) vs Partially Observable (Poker)
- Static (Chess) vs Dynamic (Driving)
- Single Agent (Atari) vs Multi Agent (Driving)
- Deterministic (Cart Pole) vs Stochastic (Driving)
- Discrete (Chess) vs Continuous (Carte Pole)

Markov Decision Processes

- MDP is the underlying basis of any RL process
  - Markov ⟶ Markov Property (Memoryless Property)

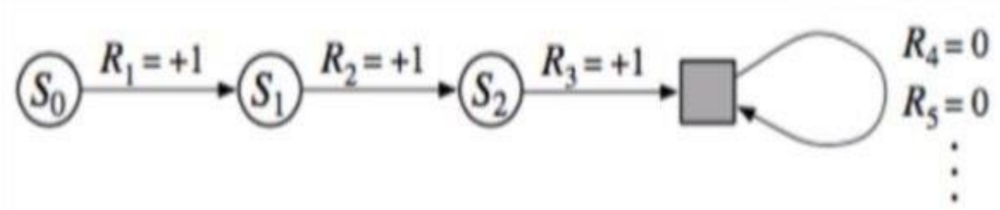
$$P(s', r \mid s, a) = P\{S_{t+1} = s', R_{t+1} = r \mid S_t = s, A_t = a\}$$



  - Decision Process ⟶ Policy (π)

The objective of MDP/RL agent ⟹ Maximizing the sum of all discounted rewards

Future Rewards



$$G(t) = \sum_{\tau=0}^{\infty} \gamma^{\tau} R(t+\tau+1)$$

$(Discount\ factor\colon\ 0 < \gamma < 1)$

- Why “discounted”?
  - Uncertainty about future due to environment stochasticity / partial observability
  - Math trick to help analyze convergence

Value Function

- The estimate of how good a state/state-action pair is!
- Almost all RL algorithms estimate value functions
- State Value Function (V(s)):

$$V_{\pi}(s) = E_{\pi}[G(t) \mid S_t = s]$$

$$V(s) = E[r + \gamma V(s')]$$

$$V_{\pi}(s) = \sum_{a} \pi(a \mid s) \sum_{s'} \sum_{r} p(s', r \mid s, a)\{r + \gamma V_{\pi}(s')\}$$

(Bellman Equation)

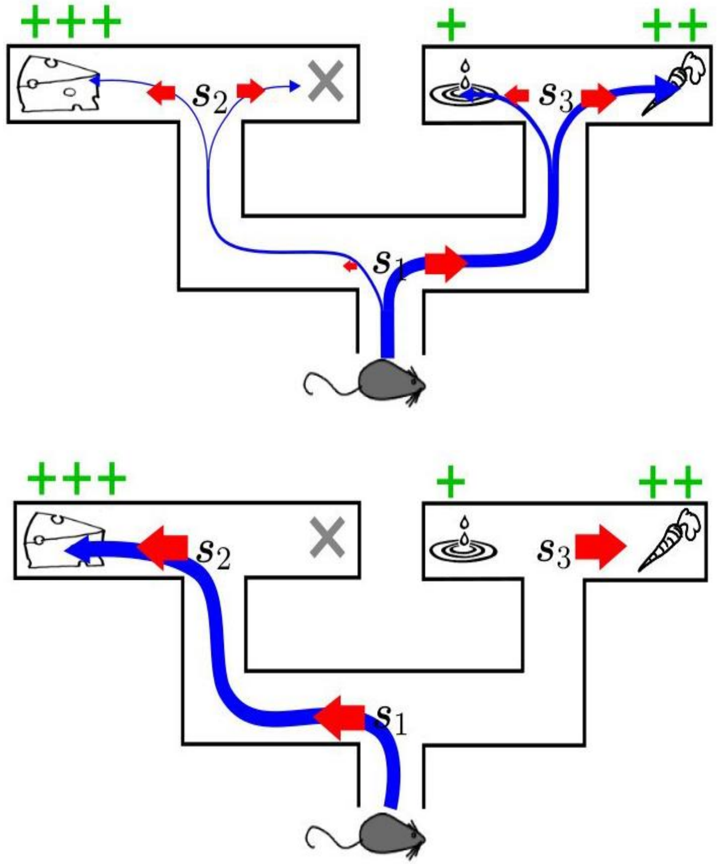
- State-action Value Function (Q(s,a)):

$$Q_{\pi}(s,a) = E_{\pi}[G(t) \mid S_t = s, A_t = a]$$

Optimal policy

$$\pi^* \quad = \quad \arg\max_{\pi} \mathbb{E}_{\pi}\Big[\sum_{t=0}^{\infty} \gamma^t r_t\Big]$$

- states $s \in \mathcal{S}$
- actions $a \in \mathcal{A}$
- rewards $r(s, a) \in \mathbb{R}$
- policy $\pi(a|s) \in \mathbb{P}(\mathcal{A})$



3 Types of Reinforcement Learning

Model-based

- Learn the model of the world, then plan using the model
- Update model often
- Re-plan often

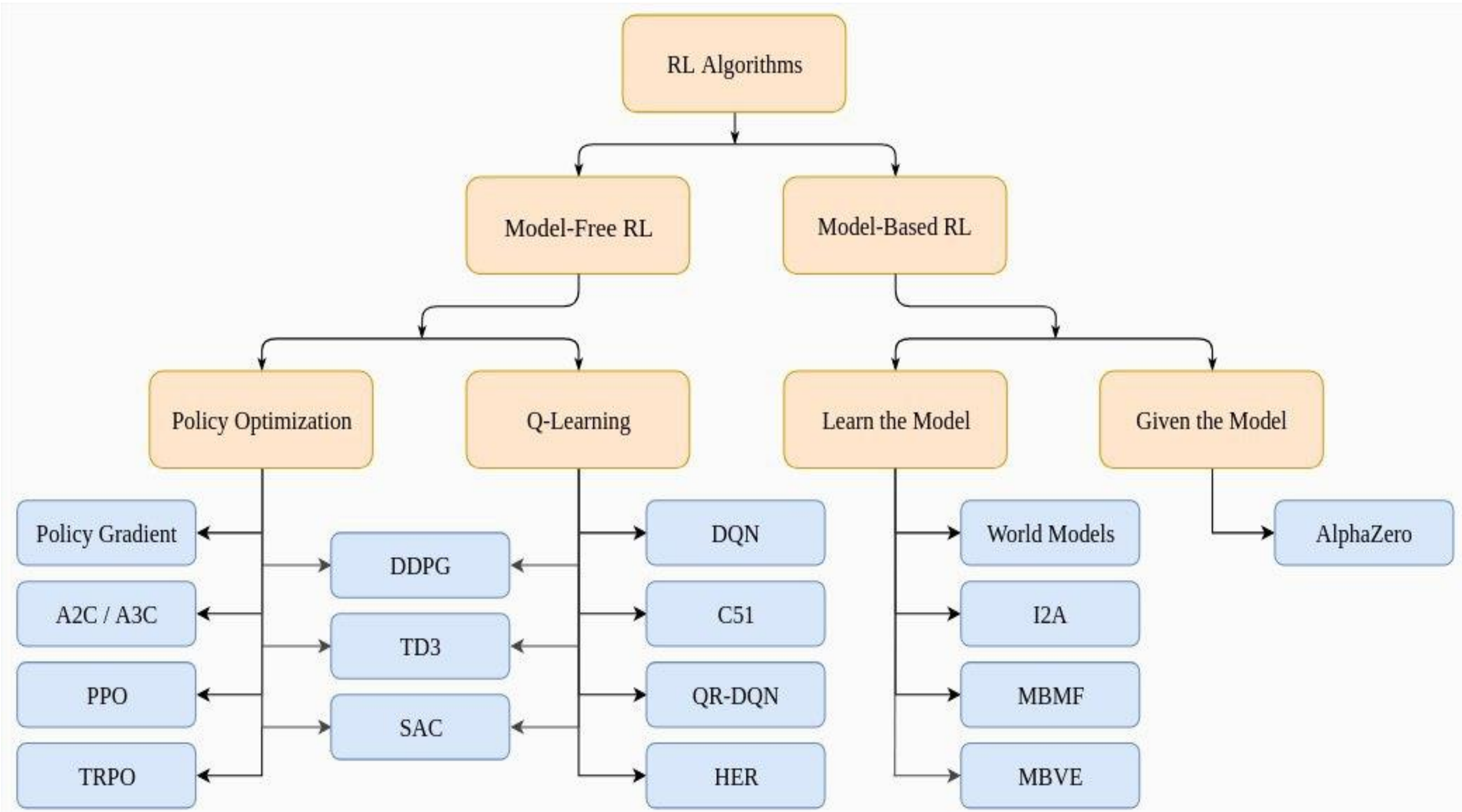
Value-based

- Learn the state or state-action value
- Act by choosing best action in state
- Exploration is a necessary add-on

Policy-based

- Learn the stochastic policy function that maps state to action
- Act by sampling policy
- Exploration is baked in

Taxonomy of RL Algorithms



Link: https://spinningup.openai.com/en/latest/spinningup/rl_intro2.html

Q-Learning

- State-action Value Function (Q(s,a))
- Off-policy and model-free method
- Use any policy to estimate Q that maximizes future reward
- Only requirement: Keep updating each (s,a) pair

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R(t+1) + \gamma max_{a'} Q(S_{t+1}, a') - Q(S_t, A_t)\right]$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha \left[r + \gamma max_{a'} Q(s', a') - Q(s,a)\right]$$

Exploration vs Exploitation

Deep Reinforcement Learning

- RL + Neural Networks
- Deep Q-Networks (DQN)
- Use a neural network to approximate the Q-function
- Loss Function (squared error):

$$L = \mathbb{E}[(\underbrace{r + \gamma \max_{a'} Q(s', a')}_{\text{target}} - \underbrace{Q(s,a)}_{\text{prediction}})^2]$$

Sources

- Reinforcement Learning: An Introduction

  https://web.stanford.edu/class/psych209/Readings/SuttonBartoIPRLBook2ndEd.pdf

- Deep Reinforcement Learning

  https://www.springer.com/gp/book/9789811382840

- https://deeplearning.mit.edu/

- https://whirl.cs.ox.ac.uk/