Learn with Google AI | Explore ML

Reinforcement Learning

Shweta Bhatt | Data Scientist | ML GDE

shweta_bhatt8

Image source

Agenda

- What is RL?
  - Introduction, applications
- Key Concepts in RL
  - Sequential decision making, state, action,
- Formulating an RL problem
  - Model, policy, value function
- Types of RL algorithms
  - Model based, model-free
- Q-learning
- Implementation platforms
- Summary
- Q & A

What is an Agent?

A system that is:

- situated in an environment
- is capable of perceiving its environment,
- is capable of acting in its environment

with the goal of satisfying its design objectives

source

What is an Environment?

Physical world in which the agent operates

Can be:

Fully observable or Partially observable

source

What is Reinforcement Learning?

Image source

What is Reinforcement Learning?

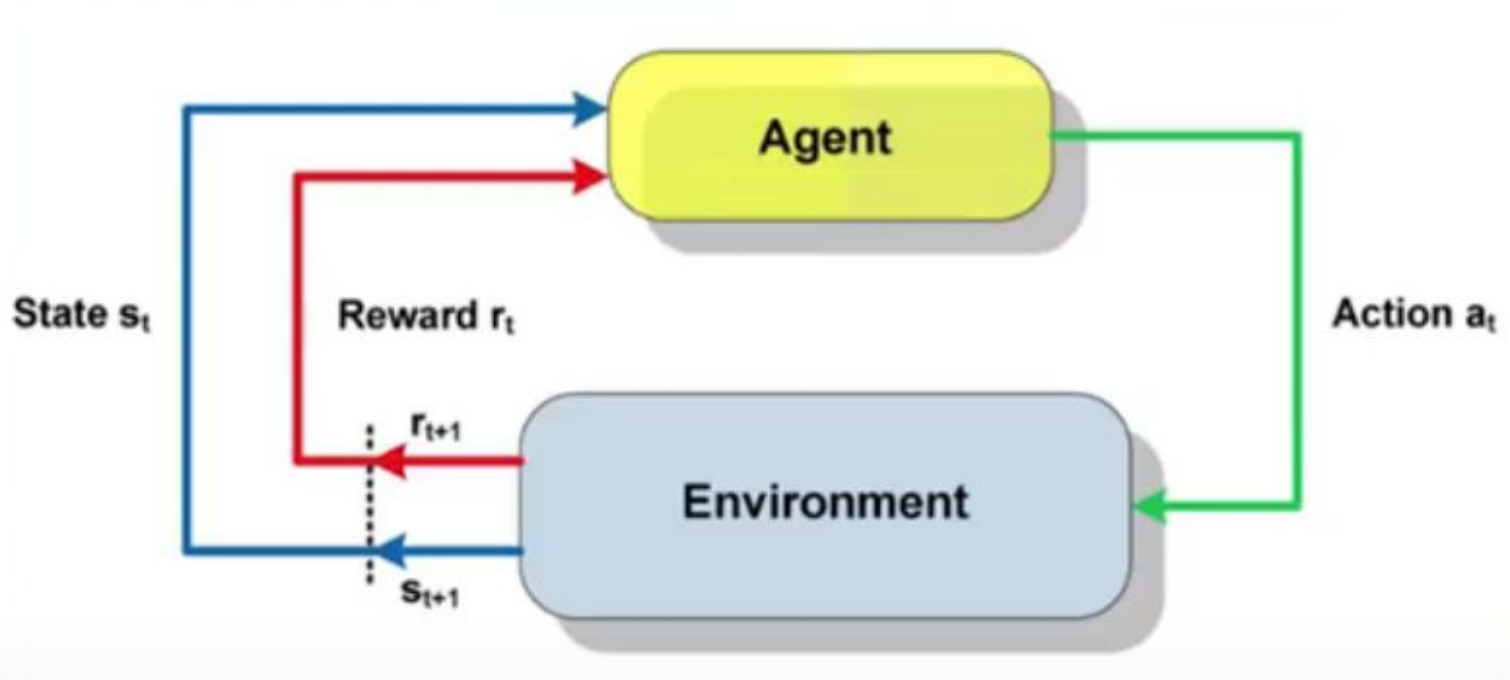


Image source

Key elements of an RL problem

- **Agent** → already defined
- **Environment** → already defined
- **State** → current situation of the agent
- **Reward** → feedback from the environment

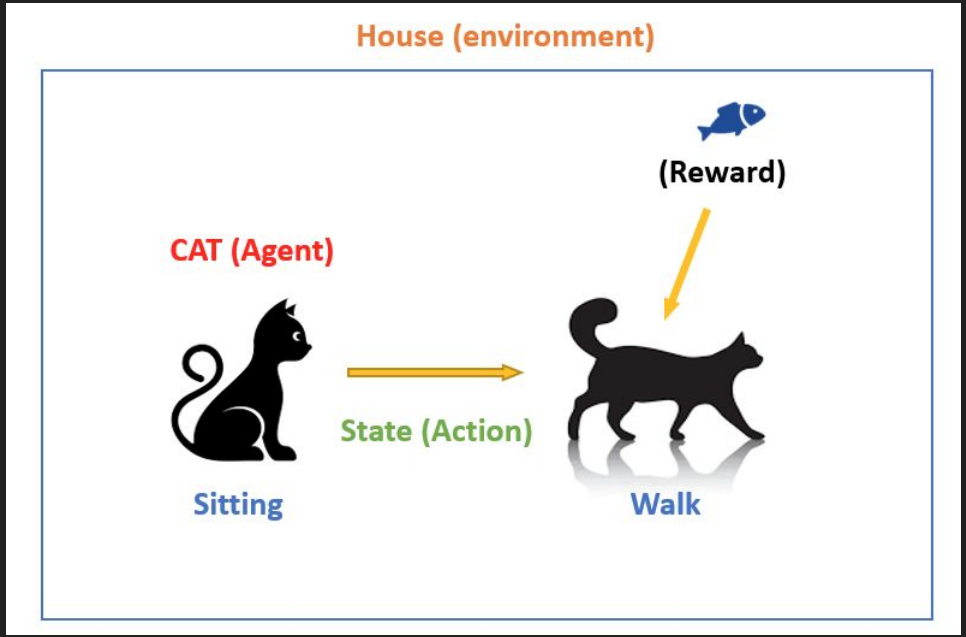


Image source

Example: PacMan

- Agent: PacMan
- Environment: The grid world or the maze
- Actions: Left, Right, Up, Down
- Rewards:
  - eating small food: 10 pts
  - eating big food: 50 pts
  - eating cherry:100 pts
  - eaten by ghost: game ends

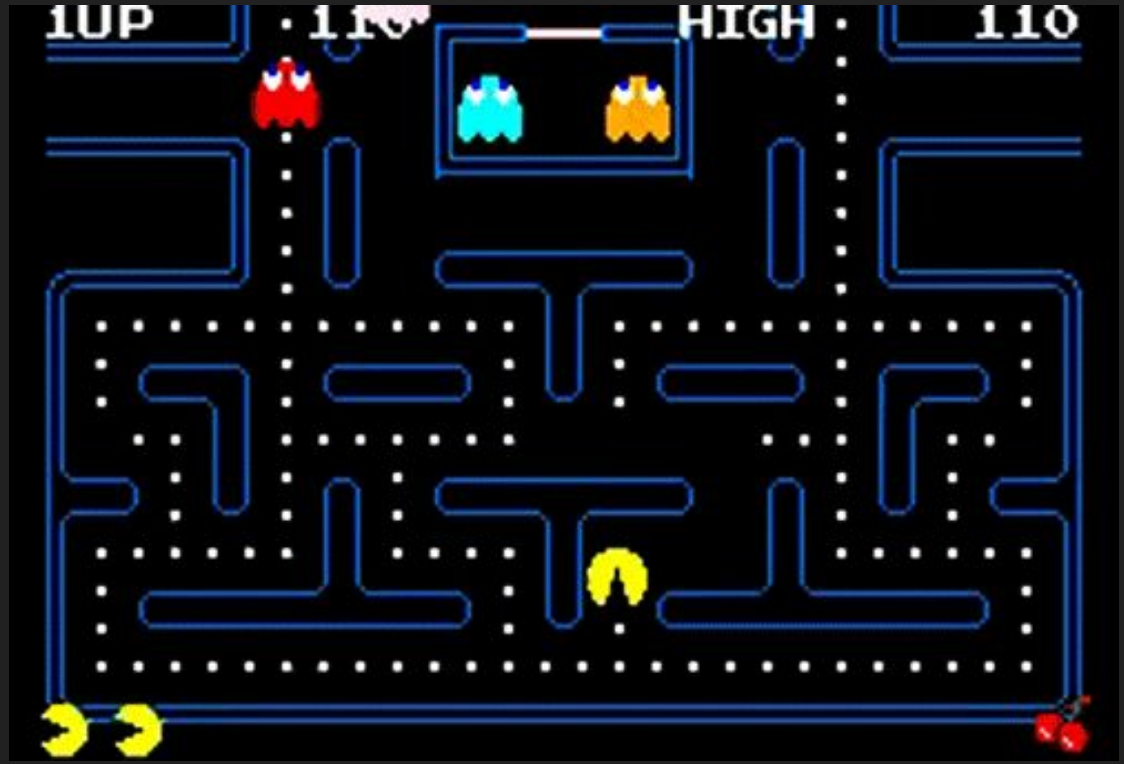


Image source

How is RL different from other types of ML?

Supervised

Data: (x,y)
X is data, y is label

Goal: Learn function to map

$x \rightarrow y$

Unsupervised

Data: x
X is data, no labels

Goal: Learn underlying structure

Reinforcement

Data: state-action pairs

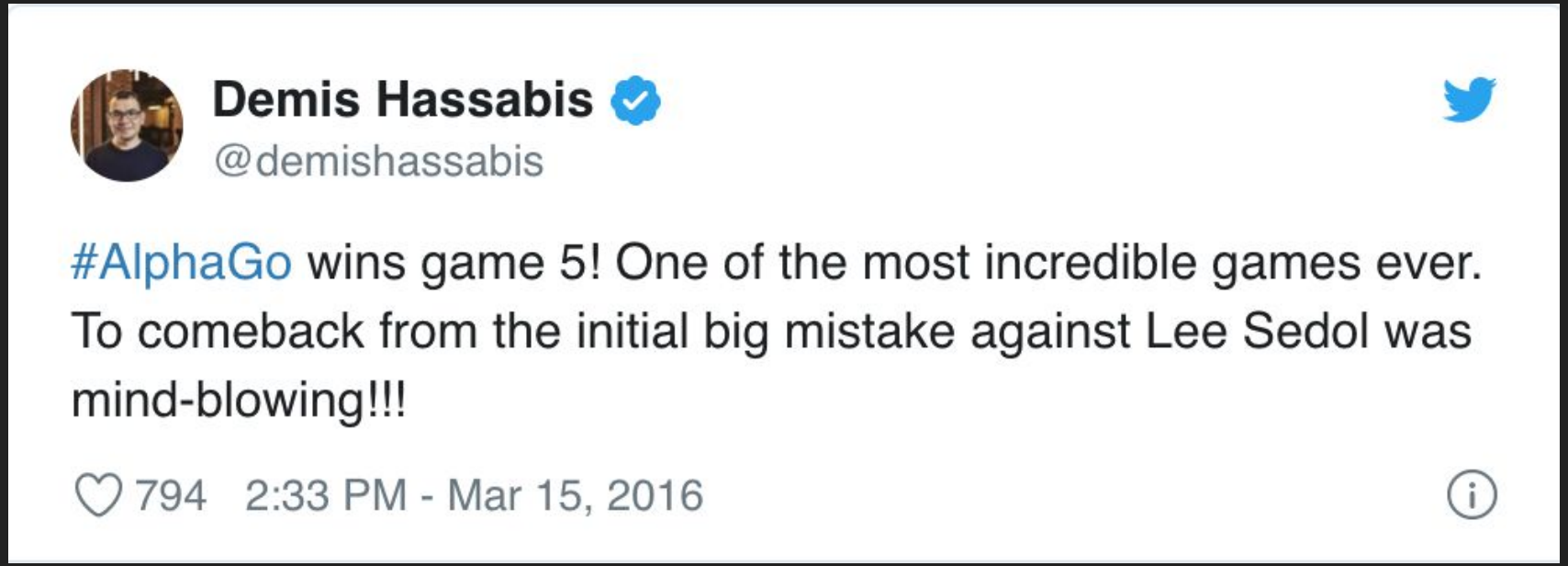
Goal: Maximize future rewards over many time steps

source

Why the hype?

AlphaGo vs Lee Sedol

Final Score: 4-1

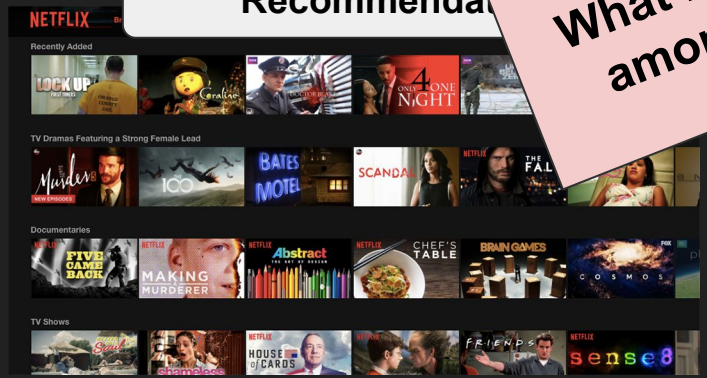


Real world applications

Robotics

Traffic Control System

Recommenda



What is common among these?

ck Trading

Sequential decision making under uncertainty

- **Goal**: Select actions to maximize expected cumulative future reward
- **Requires**: Balancing short term and long term rewards, Strategy to maximize rewards
- Example: Web Advertising

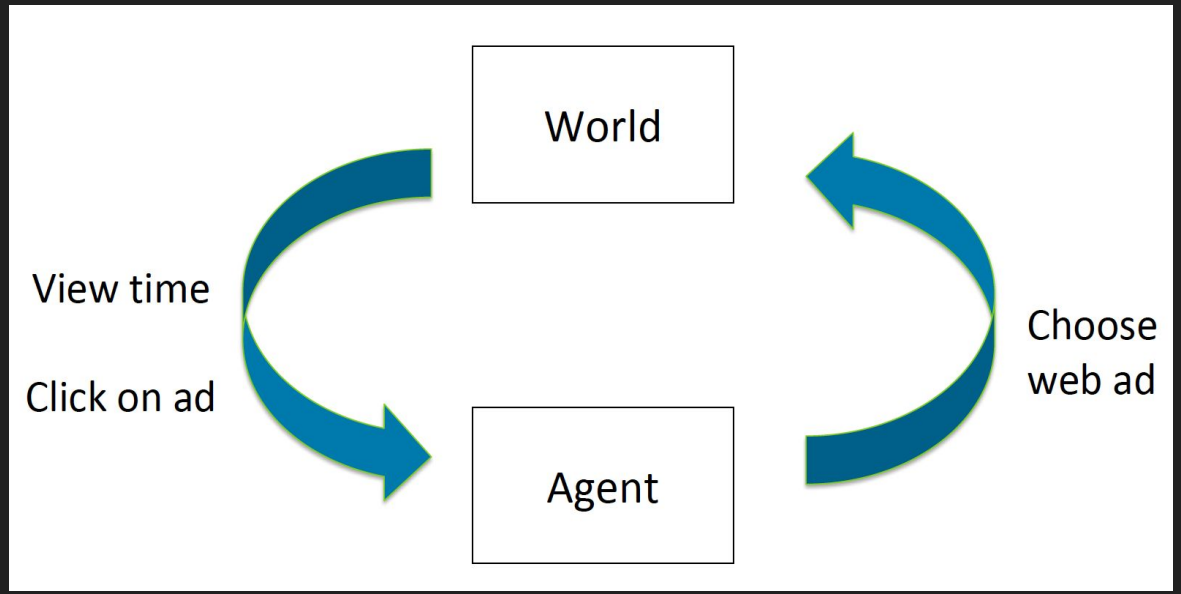


Image source

What else does RL involve?

- Optimization
- Exploration
- Delayed rewards
- Generalization

Types of sequential decision making

- Bandits
  - Agent’s state is fixed
  - Actions have no influence on next observations

- Markov Decision Processes (MDPs and POMDPs)
  - Agent’s state is dynamic
  - Actions influence future observations

Multi-arm bandits

Task: Choose repeatedly from one of n actions (play)

Objective: optimize long term cumulative reward

Image source

Exploration vs Exploitation

Exploration: Trying a new cuisine (Vietnamese)

Exploitation: Having your favorite cuisine (Indian)

Let’s formulate this problem

After each action(play) a_t at time step t you get a reward r_t, where:

$$E < r_t \mid a_t > = Q^*(a_t)$$

- Unknown action values
- Distribution of rewards (r_t) depends only on actions (a_t)

Greedy (best) action selection:

$$a_t = a_t^* = \arg\max_a Q_t(a)$$

ε-greedy action selection:

$$a_t = \begin{cases} a_t^* & \text{with probability } 1 - \varepsilon \\ \text{random action} & \text{with probability } \varepsilon \end{cases}$$

Contextual Bandits

Context ⇒ extra information that can be used for making better decision when choosing amongst all actions

Example, user history, preferences, etc

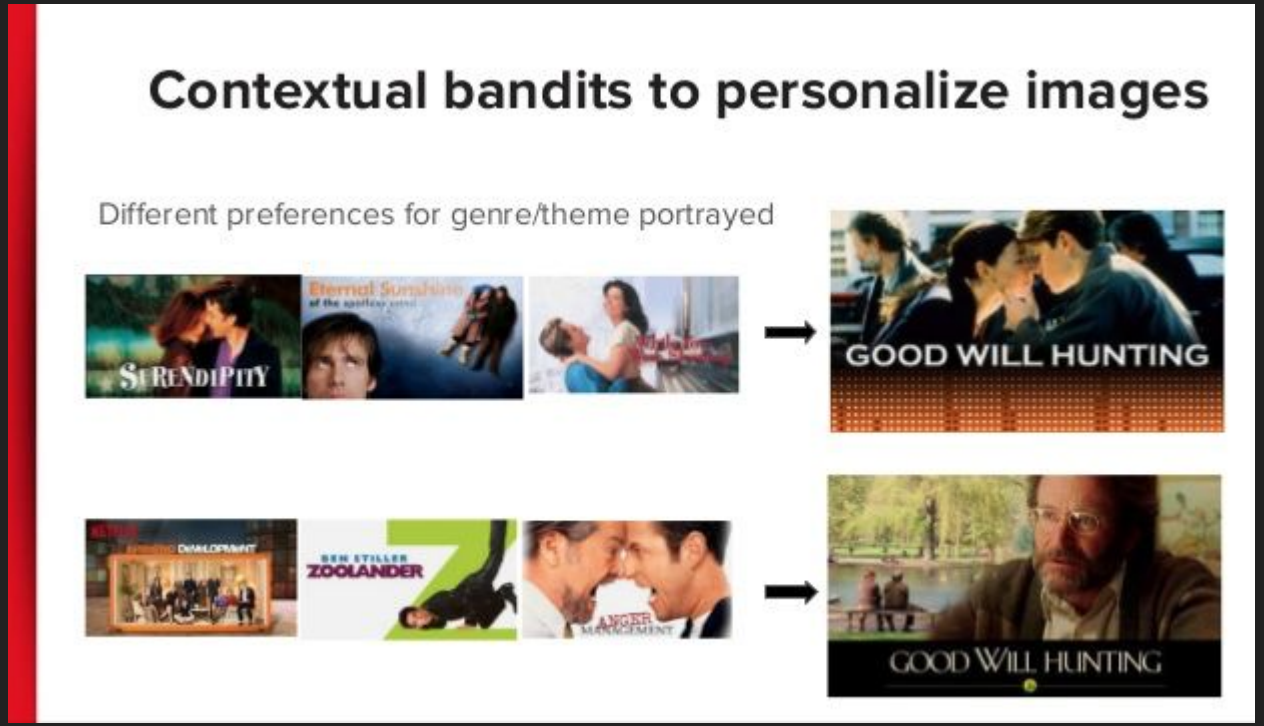


Image source

MDPs and POMDPs

What is Markov property or Markovian principle?

> Future is independent of the past given the present

Mathematically,

$$p(s_{t+1} \mid s_t , a_t) = p(s_{t+1} \mid h_t , a_t)$$

Why is it widely used?

In practice, most recent observation = sufficient statistic of history

RL Agent components

Any RL agent may include one or more:

- **Model** ⇒ representation of how the world changes in response to agent's actions
- **Policy** ⇒ Function mapping agent's states to actions
- **Value Function** ⇒ future rewards that the agent would receive by taking an action in a particular state

How to solve an MDP?

Problem Space:

Given,

- Set of states, with an initial state
- Set of actions in each state
- A transition model
- A reward function

Solution: Optimal policy ⇒ choice of action for each states in order to maximize the long term cumulative reward

Bellman Equation

We first compute utility for each state,

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} \Pr(s'|s,a) U(s')$$

Discount factor weighs immediate versus future rewards

Value Iteration

- Iterative process
- Start with arbitrary values of states and apply Bellman Equation update simultaneously to all the states:
-

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s'|s,a) U_i(s')$$

- Continue until the values of states do not change i.e. converge on the optimal values

Types of RL Agents

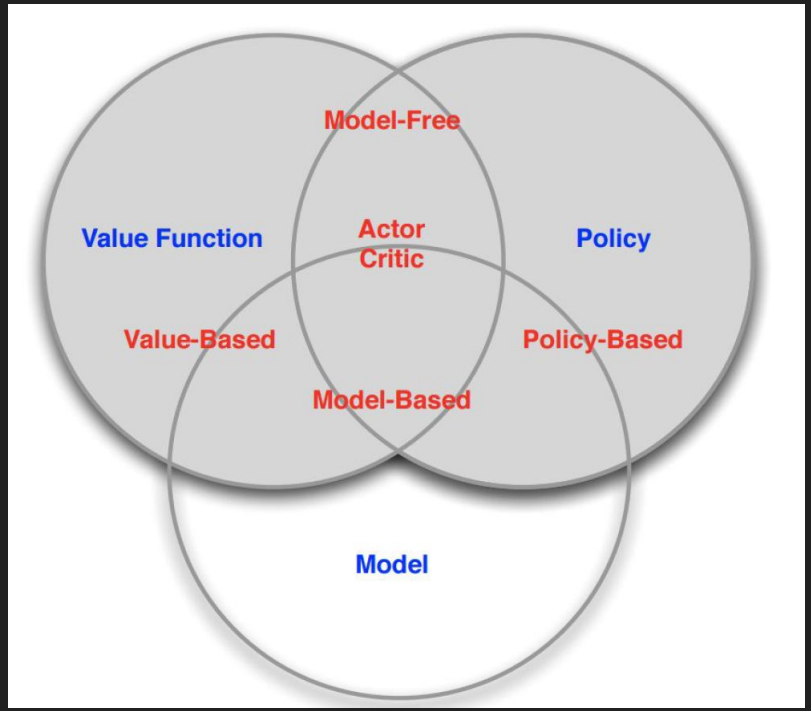


image source

Types of RL algorithms

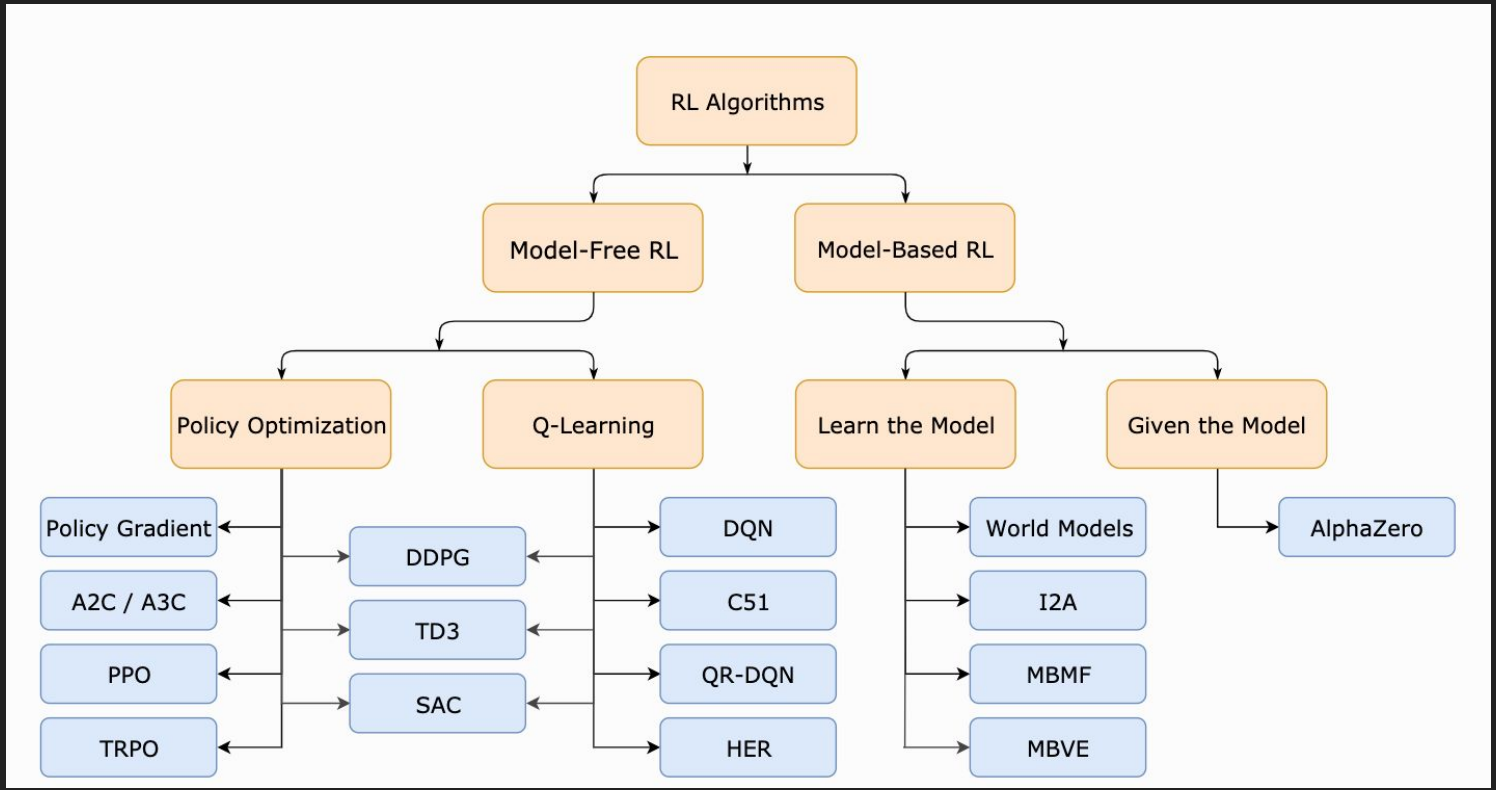


Image source

Q-learning

- Model-free approach
- Revolves around the notion of Q(s,a) = value of taking action s in state s

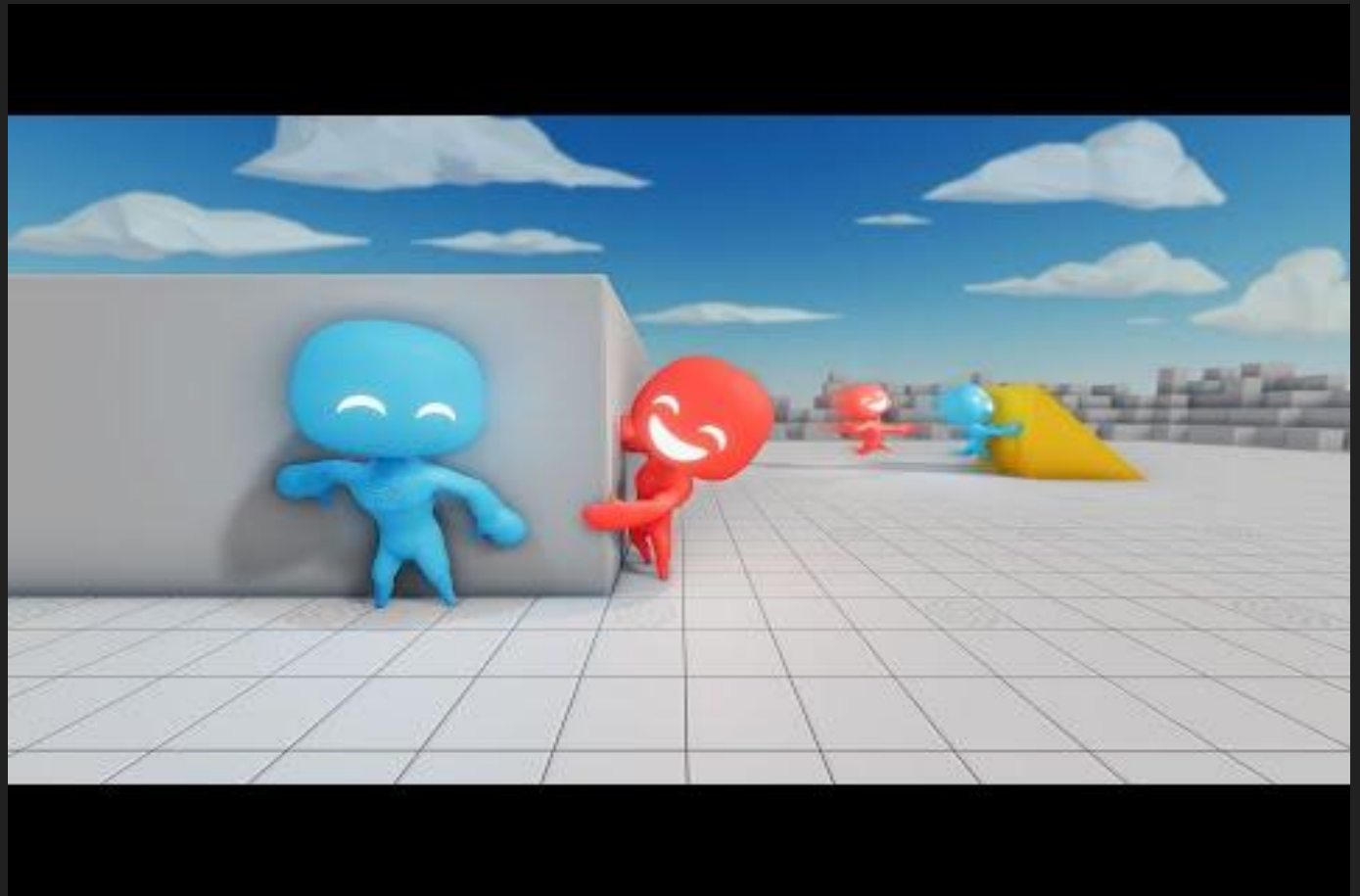
$$U(s) = \max_a Q(s, a)$$

- Update rule:

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

- Recalculate every time a is executed in s and takes agent to s’

Multi-agent RL (Hide and Seek)



Implementation platforms

- TF-Agents - Google
- OpenAI gym
- Project Malmo - Microsoft
- DeepMind Lab

Summary

- What is Reinforcement Learning?
- Applications of RL
- Sequential decision processes
- Bandit algorithms
- Markov Decision Processes
- Bellman Equation
- Types of RL algorithms
- Q-learning algorithm
- Implementation platforms available

Resources

- Videos:
  - CS 234 Reinforcement Learning - by Emma Brunskill, Stanford
  - Reinforcement Learning by David Silver
  - The Power of Self-Learning Systems -by Demis Hassabis(DeepMind)
- Books
  - Reinforcement Learning: An Introduction by Richard Sutton and Andrew Barto
  - Artificial Intelligence: A Modern Approach
- Practice Modelling Environment (research/project purposes)
  - Netlogo

Thank you!

Questions?