Introduction to Reinforcement Learning

Mickael Song

November 2, 2023

TOULOUSE INP N7

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Contents

Current Section

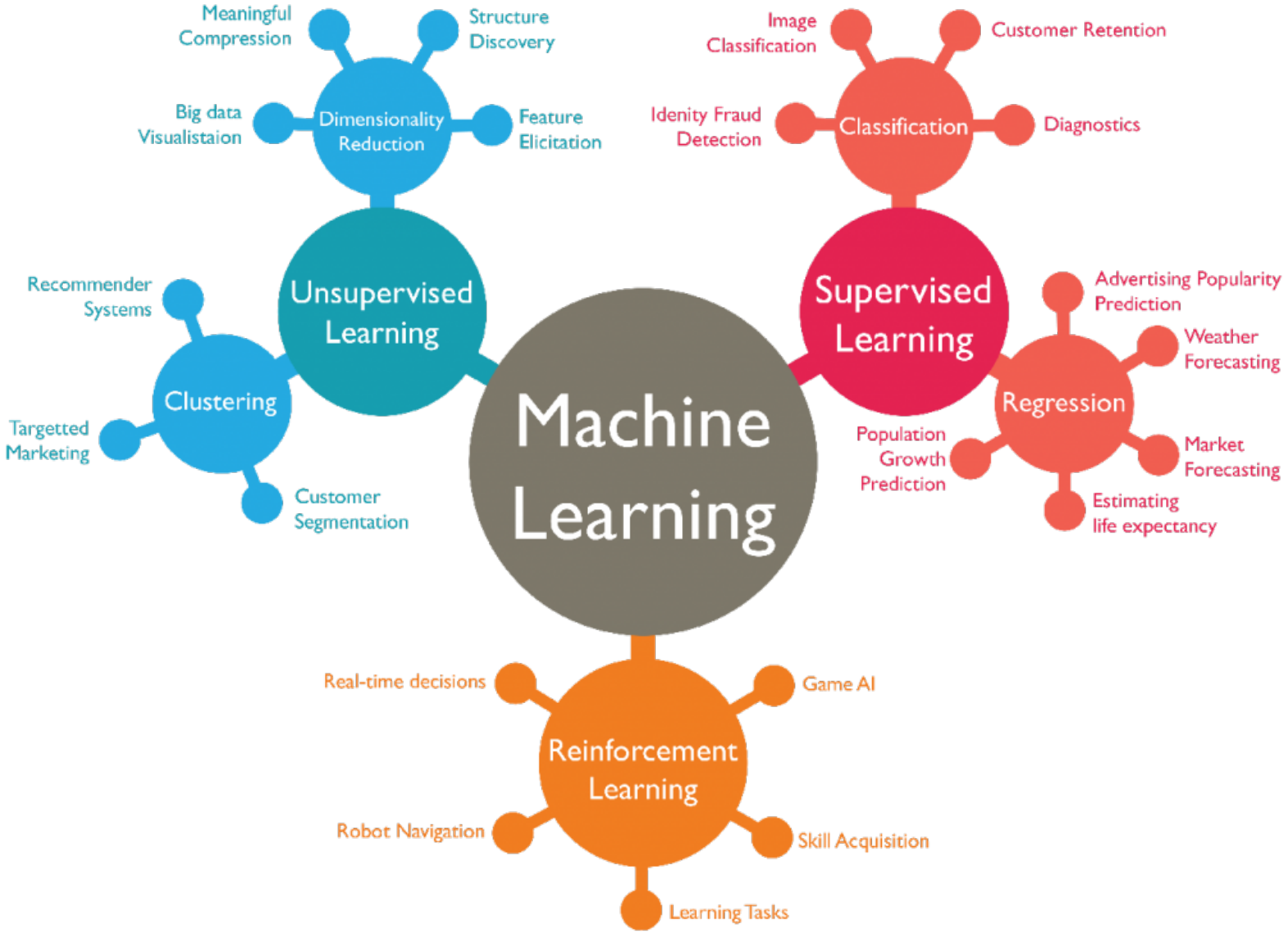


Figure 1: The 3 paradigms in Machine Learning

Introduction

Motivation

Learn and make optimal decisions in complex, uncertain, and dynamic environments

Examples

- First breakout of reinforcement learning concerned ATARI arcade games
- The champion of go Lee Sedol lost 4-1 ag. AlphaGo developped by DeepMind using RL

Introduction

Figure 2: A screenshot of an ATARI game

Figure 3: The champion of go Lee Sedol plays a match against AlphaGo

Current Section

Bandit problem & algorithms

Motivation

A much simplier problem than what RL faces yet it hightlights the importance to combine exploitation with exploration

Toy example : playing in a casino

- Imagine we are given 1000 euros that we can use on 10 different slot machines (or one-armed bandits), 1 euro each
- The average reward may vary from one slot machine to another. We initially do not know which machine is optimal
- What is the best strategy to optimize our cumulative reward after 1000 rounds?
- We should both try all machines (exploration) while playing an empirically good machine sufficiently often (exploitation)

Bandit problem & algorithms

Definition

A problem that models sequential decision tasks where the learner must simultaneously exploit their knowledge and explore unknown actions to gain knowledge for the future (exploration-exploitation tradeoff)

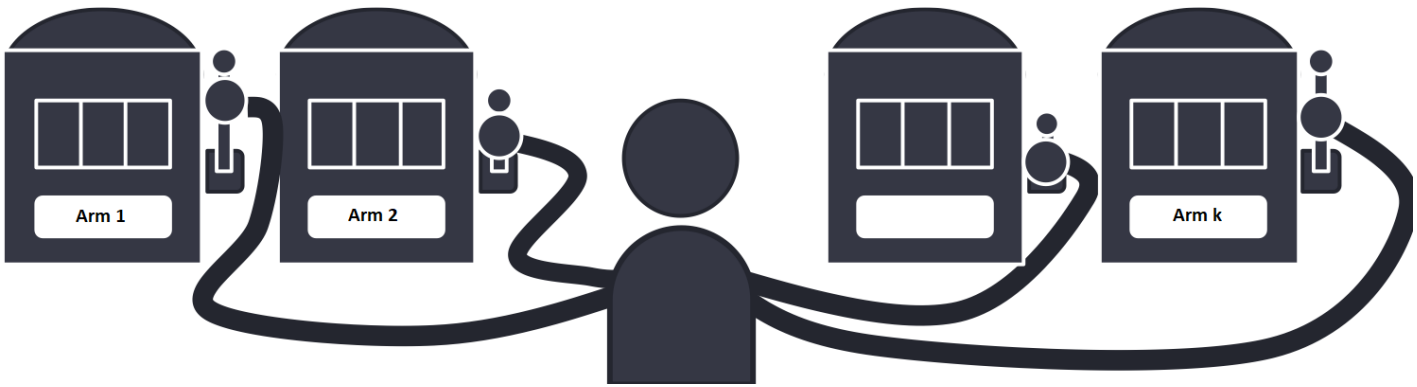


Figure 4: Multi Armed Bandit problem

Bandit problem & algorithms

Explore-Then-Commit (ETC) algorithm

- Parameter : number m of initial draws for each arm
- Exploration Phase : the algorithm explores by pulling each arm m times. This exploration phase is used to estimate the unknown reward distribution for each arm
- Commit Phase : at each round t, the algorithm choose the action that was empirically best after the first phase

Bandit problem & algorithms

ε-greedy algorithm

- Parameters : number k of arms and the probability of exploration ε
- Action Selection: choose a random number p and if p < ε, select a random arm to explore, else select the arm with the highest estimated reward
- Update Estimates: observe the reward and update the estimated reward for that arm

Next ?

- Other more powerful algorithms : Upper Confidence Bound (UCB), ...
- More complex bandit problems : combinatorial bandits, continuum-armed bandits, contextual bandits, completely arbitrary rewards...

Bandit problem & algorithms

Summary

- Bandit problem are sequential decision models where the learner must simultaneously exploit their current knowledge and explore unknown actions to gain knowledge for the future
- Yet, making an action does not change the state of the environment => Markov Decision Process

Markov Decision Process

Motivation

- More general models than MAB that include a state that can evolve over time, based on the actions of the learner
- The mathematical framework of MDPs is used for the formulation and modelisation of reinforcement learning methods

Markov Decision Process

Definition

- A mathematical framework used to model decision-making problems involving sequential interactions between an agent and an environment
- At time step t :
  - State $S_t \in \mathscr{S}$ and new state S_{t+1} : environment
  - Action $A_t \in \mathscr{A}(S_t)$: chosen action
  - Reward $R_{t+1} \in \mathscr{R}$: immediate reward
- Transition probability $\mathbb{P}(S_t = s', R_t = r | S_{t-1} = s, A_{t-1} = a)$

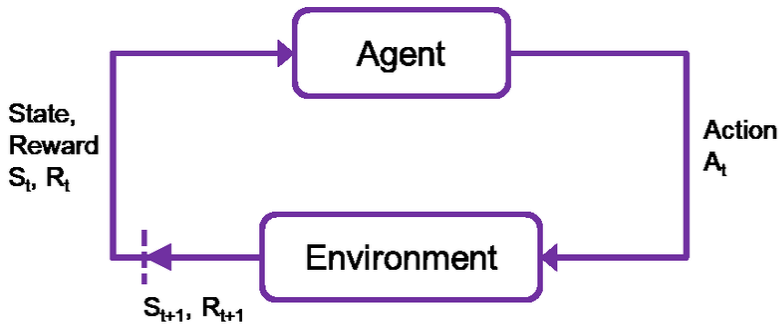


Figure 5: The agent-environment interaction in a MDP

Markov Decision Process

Definitions

- $\mathscr{S}, \mathscr{A}, \mathscr{R}$ are often finite
- Policy (determinist case) $\pi : \mathscr{S} \rightarrow \mathscr{A}$: Describes the choices of actions to be taken by the agent in each state
- Policy (stochastic case) $\pi : \mathscr{S} x \mathscr{A} \rightarrow [0, 1]$: Maps each state to a probability distribution over actions

Dynamic Programming (DP)

- DP is a general optimization technique used to find optimal policies and value functions by recursively solving subproblems
- The Bellman equation is a key equation in DP that expresses the relationship between the value of a state and the values of its successor states

Markov Decision Process

Summary

- MDP provides the formal framework for modeling decision-making problems
- RL is the field of study concerned with learning optimal decision-making policies within these framework
- RL extends the MDP framework to scenarios where the agent learns from interactions and may not have complete knowledge of the MDP's parameters (transitions, rewards ...)

Current Section

RL definition

RL Settings

- Environment: provides a reward and a new state for any action of an agent
- Agent policy π: choice of an action A_t from the state S_t
- Total reward: (discounted) sum of the rewards.

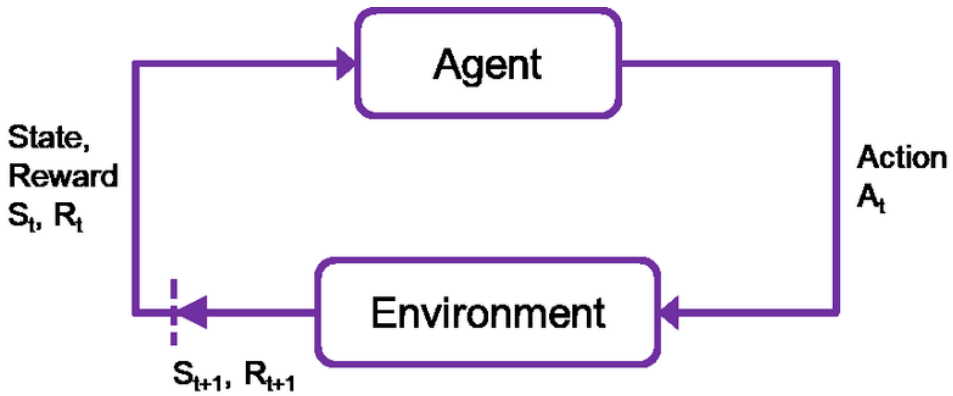
Questions

- Policy evaluation: how to evaluate the expected reward of a policy knowing the environment?
- Planning: how to find the best policy knowing the environment?
- Reinforcement Learning: how to find the best policy (maximize the cumulutive reward) without knowing the environment?

RL definition

Reinforcement Learning - Sutton (1998)

An agent takes actions in a sequential way, receives rewards from the environment and tries to maximize his long-term (cumulative) reward



RL definition

Two approaches in RL

- Model-based Learning approach : the agent try to learn a model of how the environment works by estimating the transition and reward function from observations, it can then use a planning algorithm with its learned model to find a policy
- Model-free Learning approach : the agent will find an optimal policy diretly with its interaction with the environment without having a concrete model

RL definition

Summary

- In RL, the agent does not know how the world will change in response to its actions (the transition function), nor what immediate reward it will receive for doing so (the reward function)
- The goal of RL is to find the best policy by maximizing the cumulutive reward

Q-learning

A model-free algorithm : Q-Learning

- Based on the concept of the Q-function (also known as the action-value function)
- The Q function of a policy π, Q^{π}(s,a) measures the expected return or discounted sum of rewards obtained from a state by taking a specific action first and following the policy thereafter
- The fundamental idea is to use the Bellman optimality equation as an iterative update

$$Q_{i+1}(s,a) \leftarrow \mathbb{E}[r + \gamma max_{a'} Q_i(s',a')]$$

- In practice, the q function is implemented using a table with state as rows and action as columns

Q-learning

Deep Q-Learning (DQN)

- For most problems, it is impossible to represent the Q-function as a table containing values for every combination of state and action
- We approximate the Q-value function using Deep Neural Networks (DNN)
- The DNN takes as an input a state and outputs the Q-values of all possible actions for that state

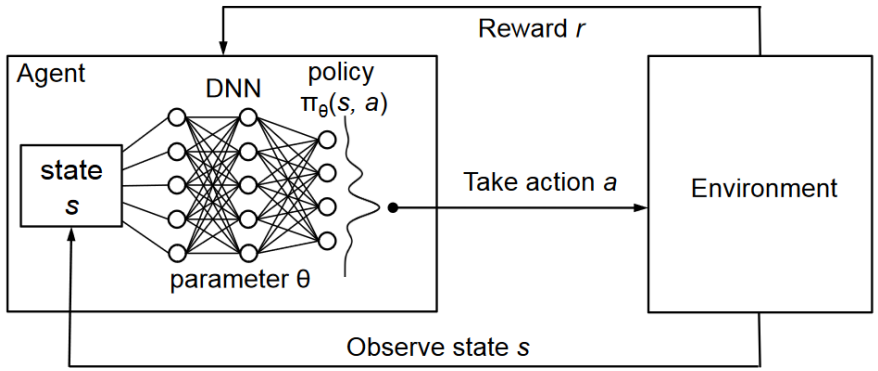


Figure 6: Reinforcement Learning with policy represented via DNN

Q-learning

Deep Q-Network: Replayed Memory

- The agent stores past experiences (state, action, reward, next state) in a memory buffer
- During training, batches of experiences are randomly sampled from the buffer to decorrelate and reuse experiences for stable learning
- During the training, we use as a loss function the Temporal Difference error function (TD function), which is the difference between the Q-value of a state-action pair and its Q-Target

Off-policy algorithm

DQN is an off-policy model-based algorithm : the agent uses a learning policy (he takes the highest Q-value as the best actions) different from the exploration policy (here ε-greedy)

Q-learning

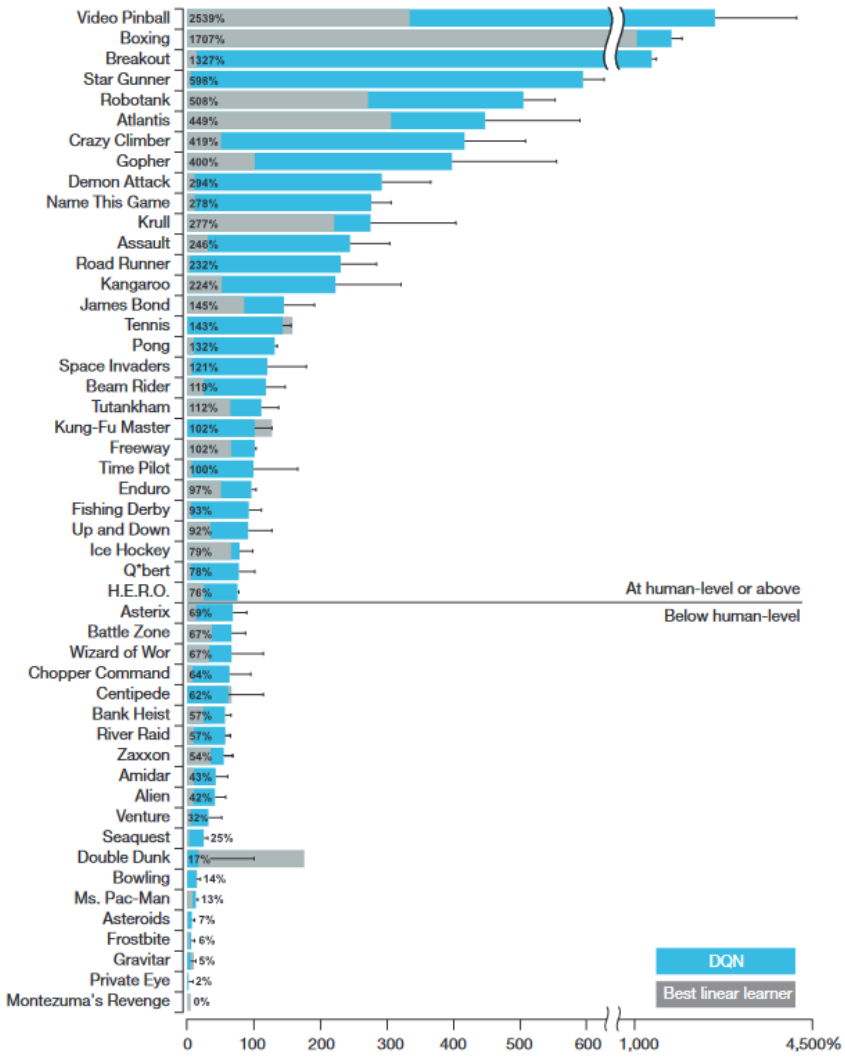


Figure 7: Performances of DQN on ATARI arcades games

Q-learning

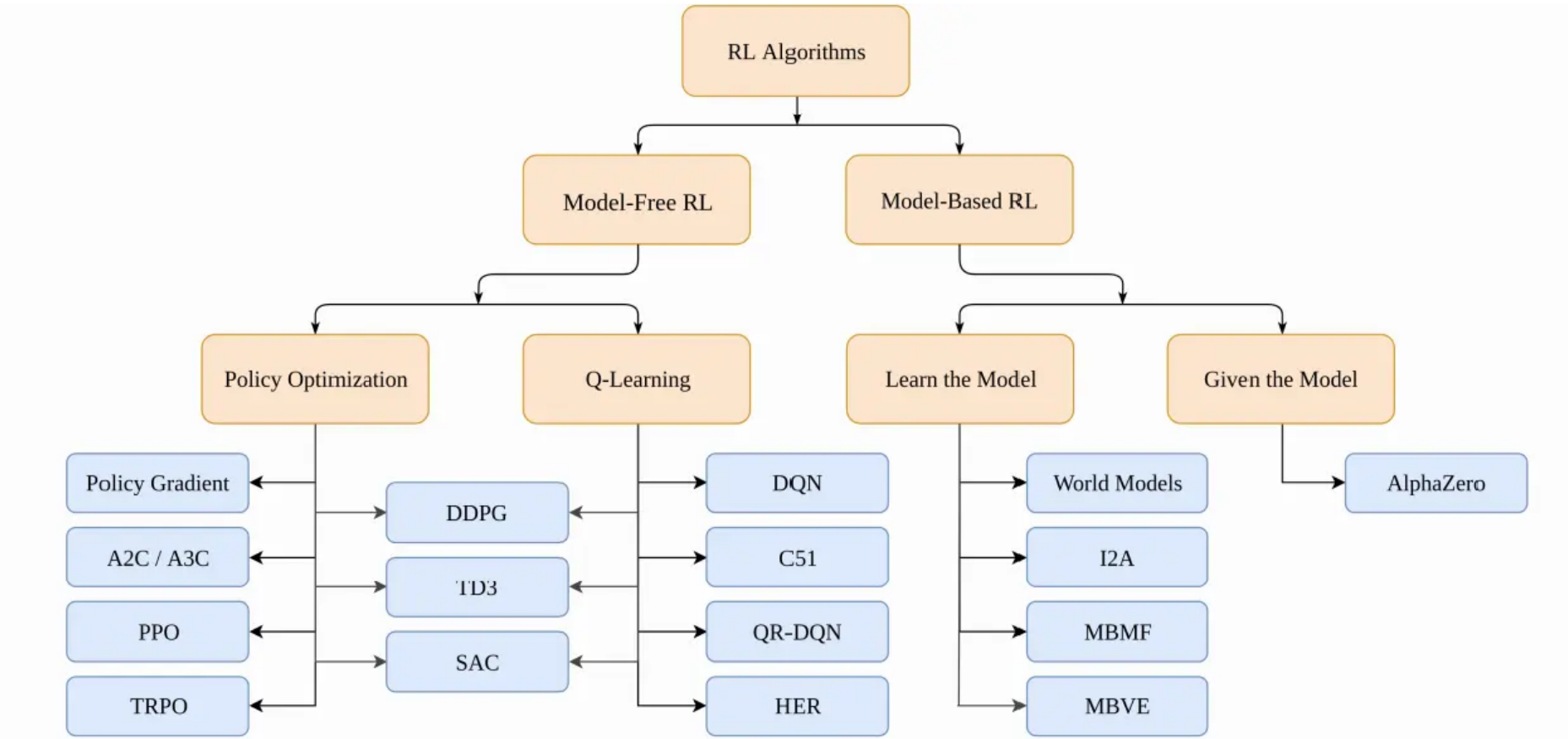


Figure 8: A non-exhaustive list of model-free and model-based algorithms

Q-learning

Summary

- Q-learning and Deep Q-Network are key algorithms in reinforcement learning
- The agent learns a value for each action in each state, called the Q-value, which represents the expected cumulative reward. Over time, the agent learns to make optimal decisions to maximize its total reward.

Current Section

References

- Fig 1 : from the website 'https://favouriteblog.com/15-algorithms-machine-learning-engineers/'
- Fig 3 : from the Time 'https://time.com/4257406/go-google-alphago-lee-sedol/'
- Fig 4 : from the website 'https://people.stfx.ca/jdelamer/courses/csci-531/topics/multi-armed-bandit/the-problem.html'
- Fig 6 : from the paper 'Resource Management with Deep Reinforcement Learning'
- Fig 7 : from the paper 'Human Level Control Through Deep Reinforcement Learning'
- Fig 8 : from the website 'https://spinningup.openai.com/en/latest/'

Current Section

Acknowledgement

Gerchinovitz Sebastien, research scientist at IRT Saint-Exupery and professor at University Paul Sabatier III & INSA Toulouse