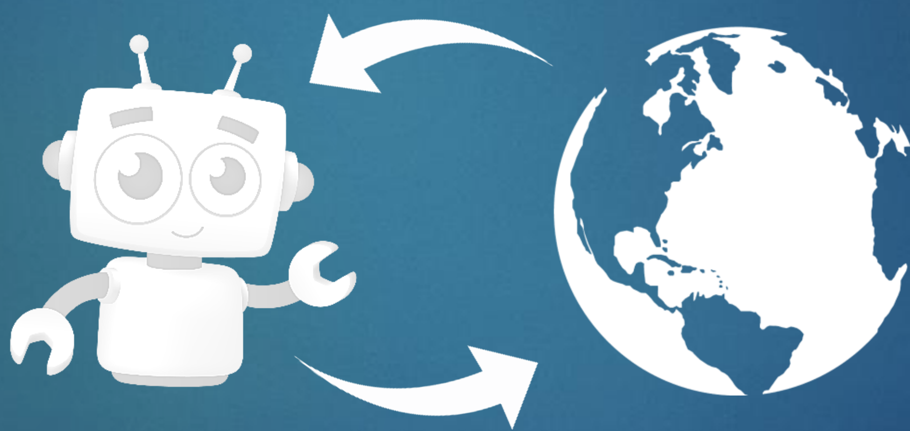
alitourani

Deep Learning from Scratch

Session #9: Reinforcement Learning



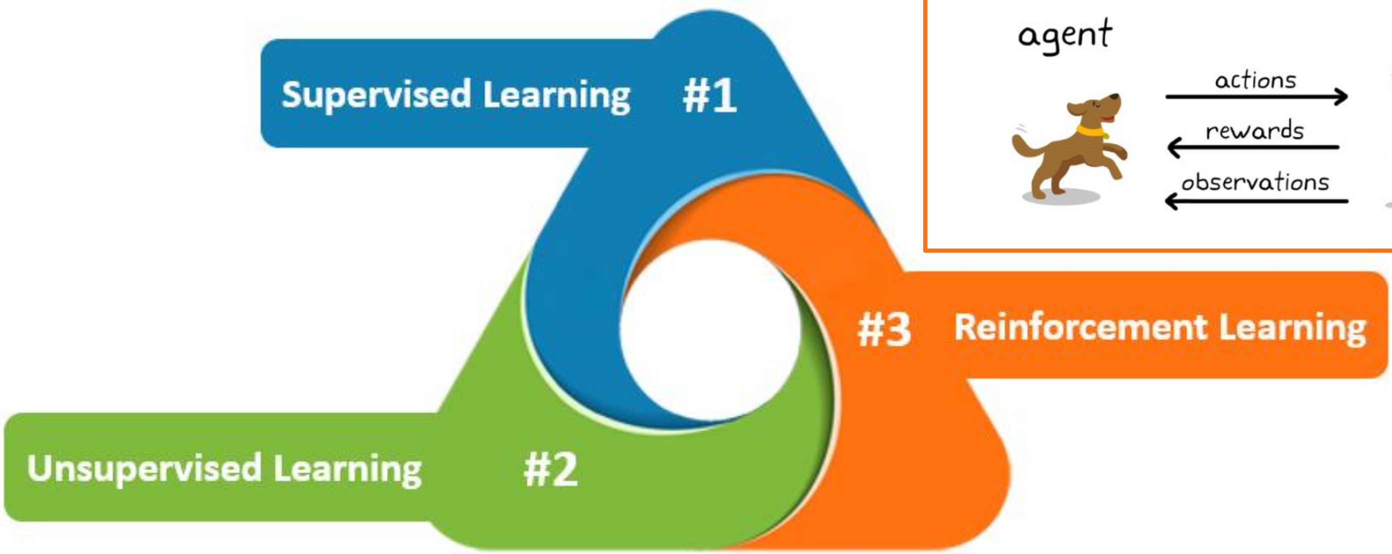
by: Ali Tourani – Summer 2021

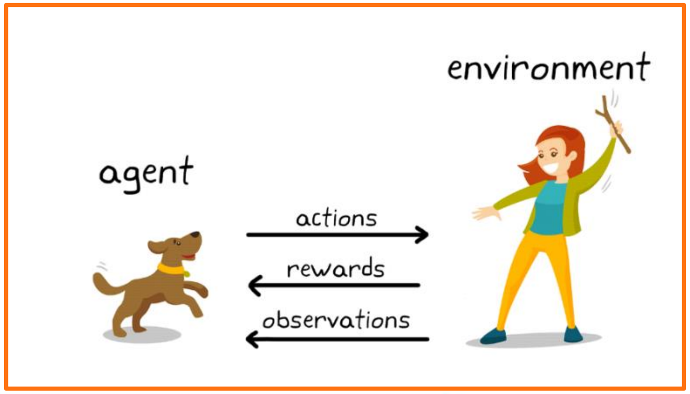
Agenda

- Reinforcement Learning
- Applications of RL
- Deep RL Algorithms
- Deep Q-Learning

Reinforcement Learning

- Different learning paradigms
  - Recall: Session#3 – Feeding DNNs

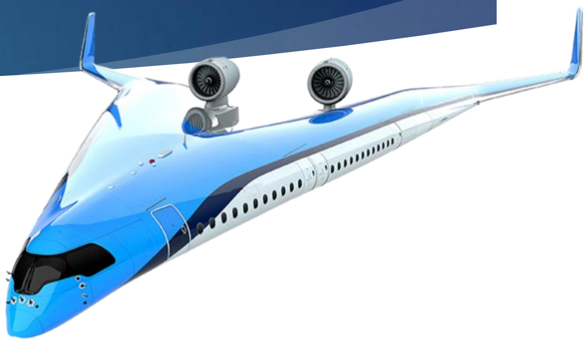




Reinforcement Learning

- Using DNNs to interact with real dynamic environments
  - **Data:** state(observations)-action(behaviors) pairs
  - **Goal:** Maximizing future rewards
- Trying to maximize the total (cumulative) reward
  - An agent learns to achieve a goal in an uncertain environment
  - Based on Reward and Penalty
- The model itself should find the solution with a maximized reward
  - Finding a solution with the lowest possible costs in future
  - Maybe even with trial and error

Reinforcement Learning



- Training models to make a sequence of decisions
- Very common in game-like situations
- Advantage: empowering machines' creativity!
  - How? By gathering experience from thousands of parallel routes
- Challenges:
  - Building a realistic simulation of environment
    - **Examples:** how to test a self-driving airplane in all possible challenging conditions?
  - Communication with the network through controlling the agent
  - Finding the local optimum for the agent (finishing the assigned tasks)

Image source: https://www.ft.com

Reinforcement Learning

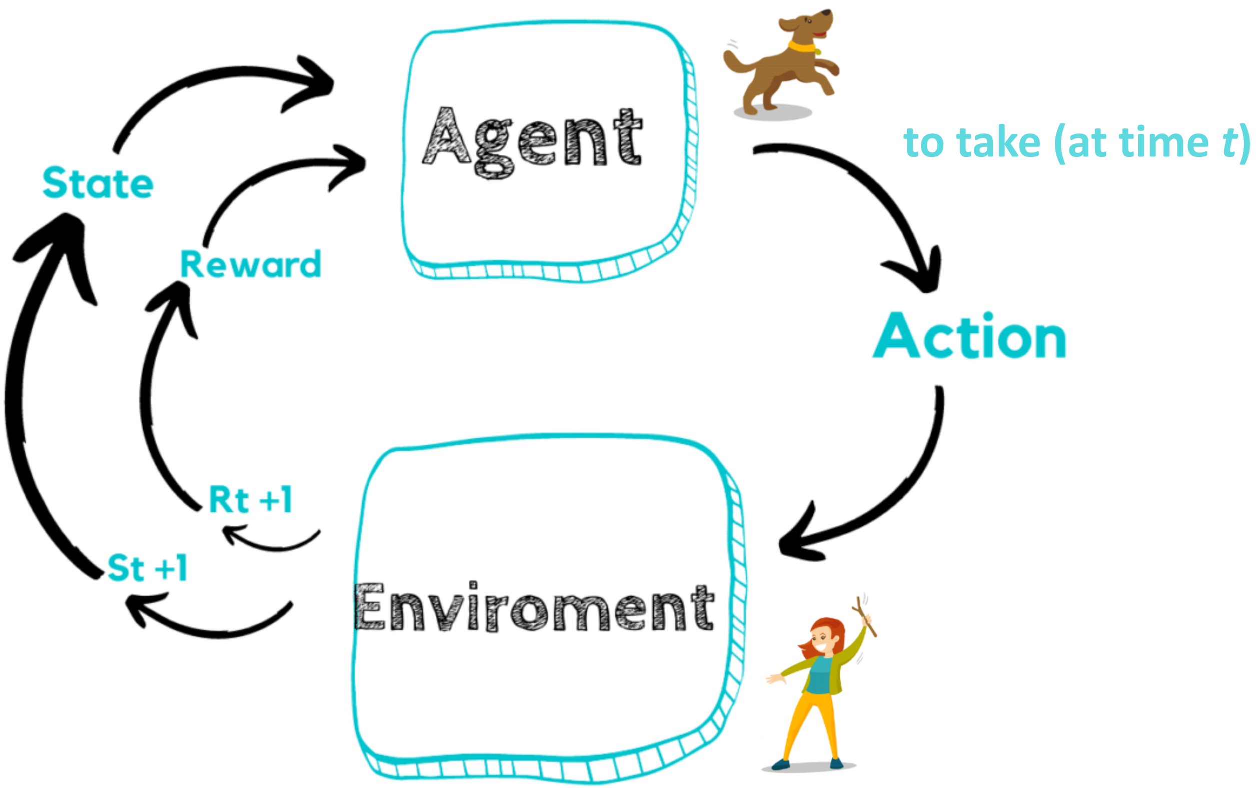
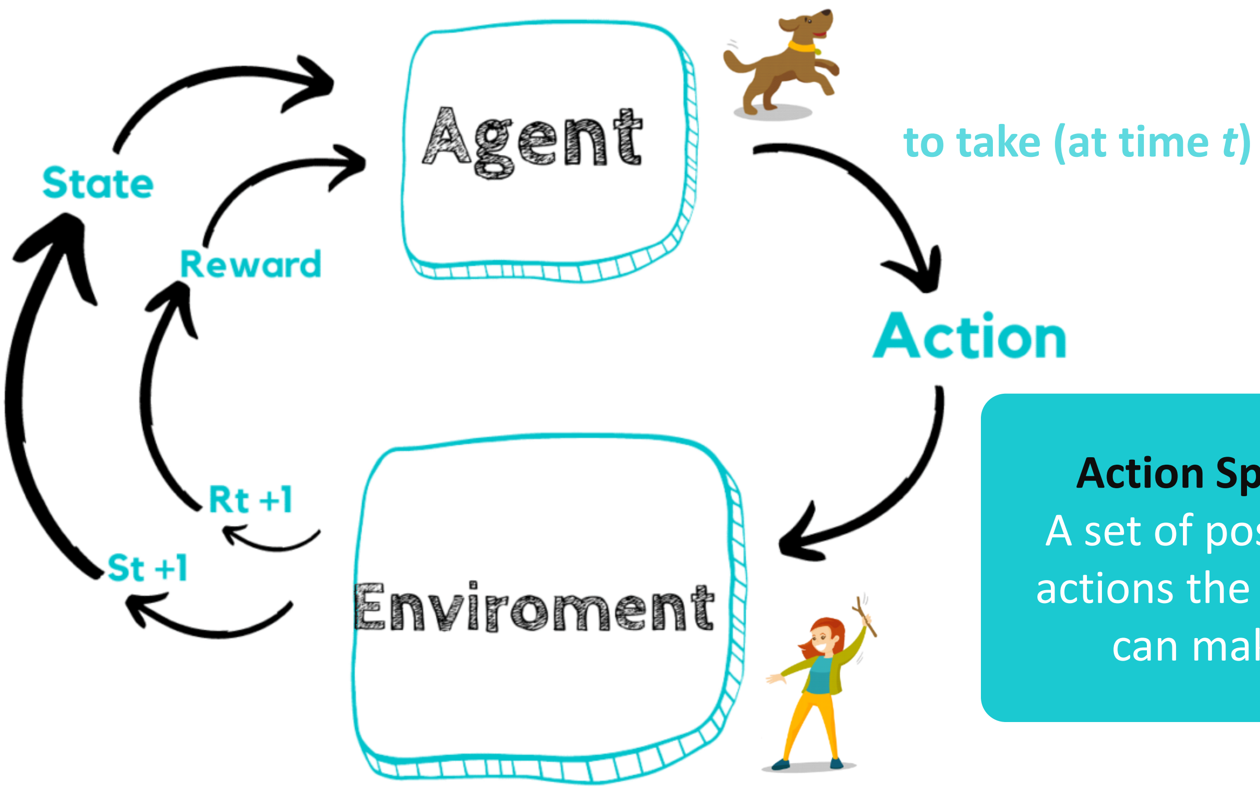


Image source: https://www.kitrum.com

Reinforcement Learning



Action Space
A set of possible actions the agent can make

Image source: https://www.kitrum.com

Reinforcement Learning

State
a situation in which the agent finds itself

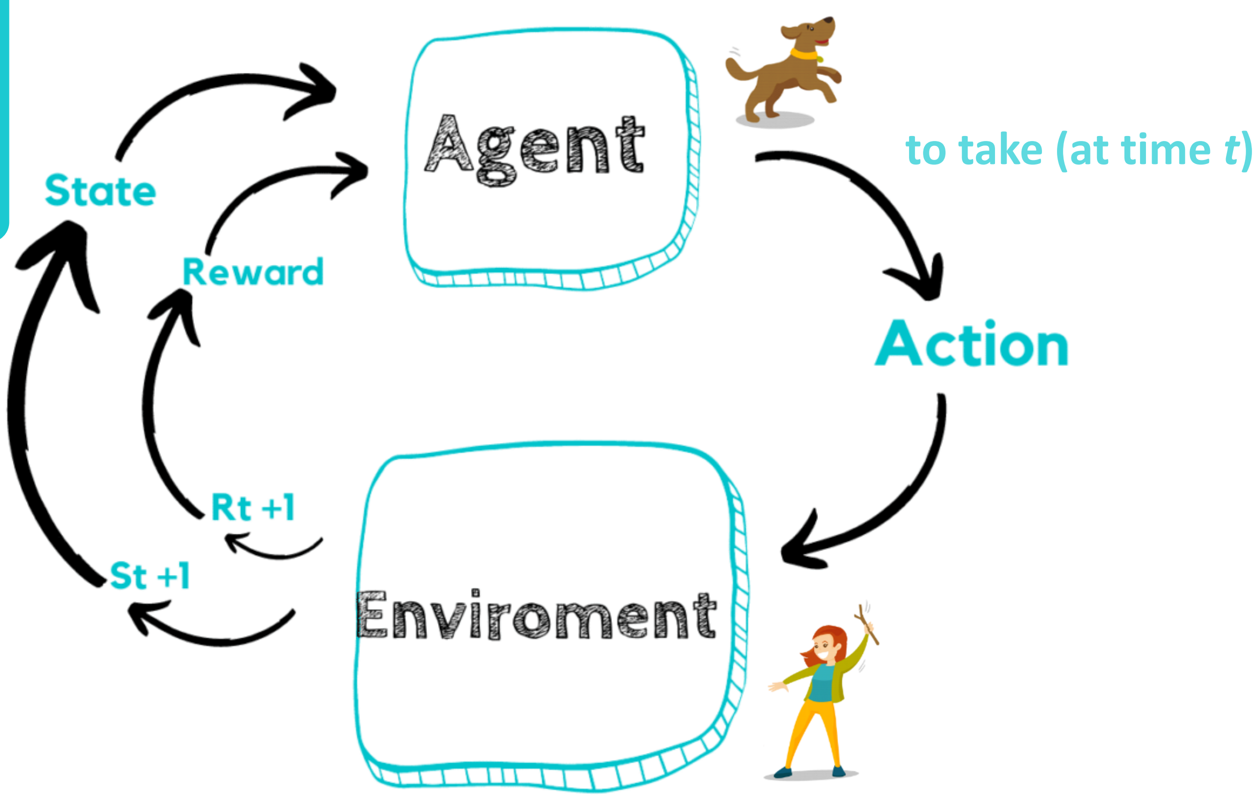
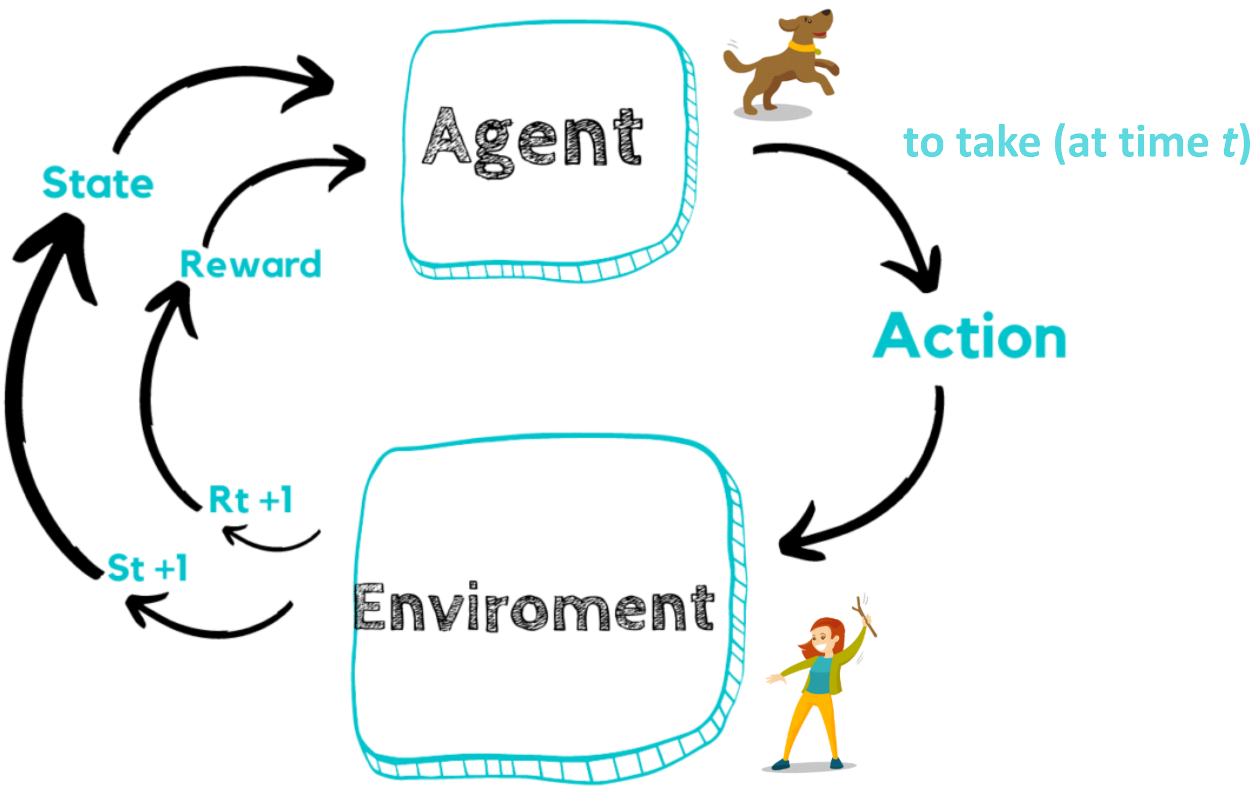


Image source: https://www.kitrum.com

Reinforcement Learning



Reward
a feedback
showing the
success or failure
of the action

Immediate or delayed

Image source: https://www.kitrum.com

Reinforcement Learning

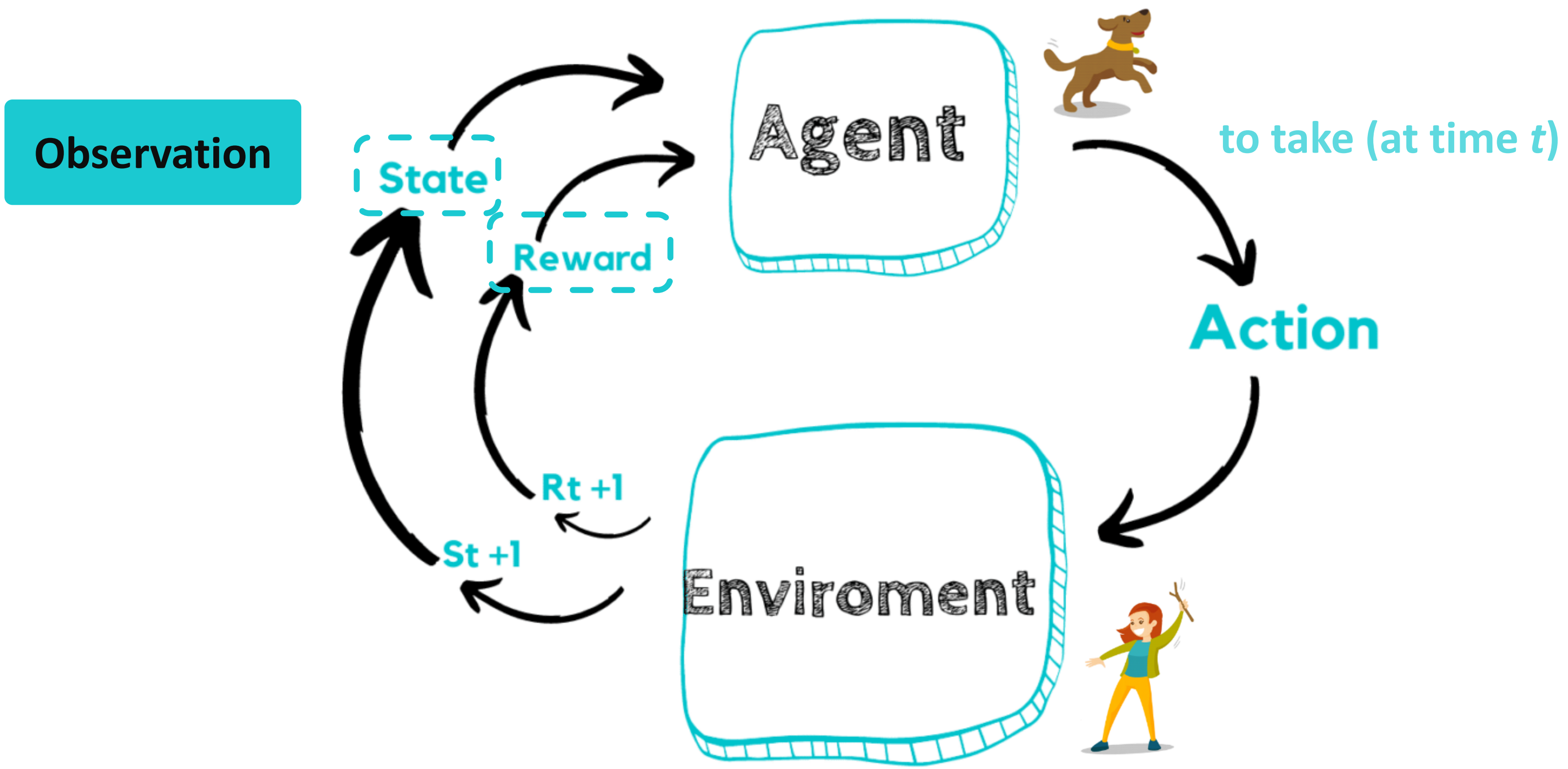


Image source: https://www.kitrum.com

Reinforcement Learning

Image source: https://www.retrogame.cc

Reinforcement Learning

Image source: https://www.retrogame.cc

Reinforcement Learning

Image source: https://www.retrogame.cc

Reinforcement Learning

Image source: https://www.retrogame.cc

Reinforcement Learning

Important notes on RL

- Total Reward (TR) is a key concept, which is equal to the sum of all rewards

$$R_{total} = \sum_{i=t}^{\infty} reward_i = \sum_{i=t}^{\infty} \gamma^i . reward_i \qquad (0 < \gamma < 1)$$

  - Where γ is the discounting factor to make future rewards less effective
  - **Goal:** enforcing short-term learning for the algorithm

- Q-function is another key concept that takes the current state and action, and returns the <u>expected</u> total reward

$$Q(state_t, action_t) = E[R_{total} \mid state_t, action_t]$$

Reinforcement Learning

Important notes on RL

- The main role of the Q-function is to define the best possible action
  - How? Just choose a policy to maximize the future reward when **different actions** are fed in a **known state**
- Agents in RL take random decisions

in the environment to learn selecting

the right choice

- A **policy** is a mapping $s \rightarrow a$
  - Reinforcing the agent to learn to perform the

best actions by experience

Image source: https://www.bayanbox.ir

Applications of RL

Self-driving (autonomous) Cars

Image source: https://www.washingtoninjury.com

Applications of RL

Natural Language Processing (NLP): Question Answering

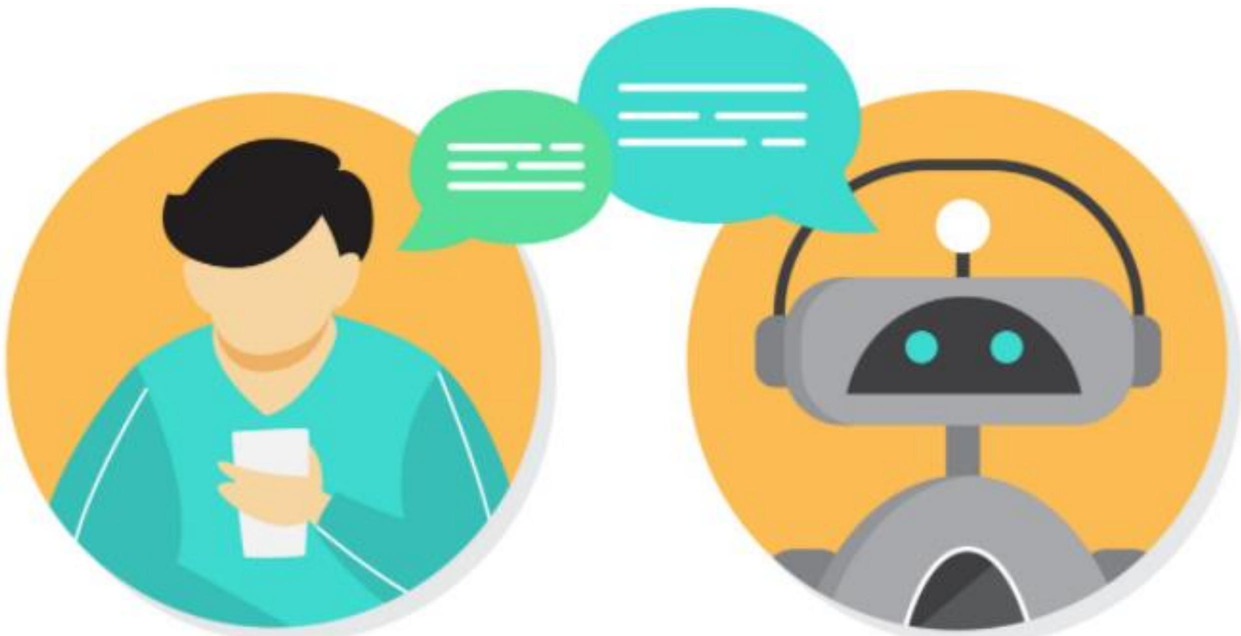


Image source: https://www.towardsdatascience.com

Applications of RL

DeepMind: Autonomous Data Center Cooling Systems

Image source: https://www.deepmind.com

Applications of RL

Healthcare

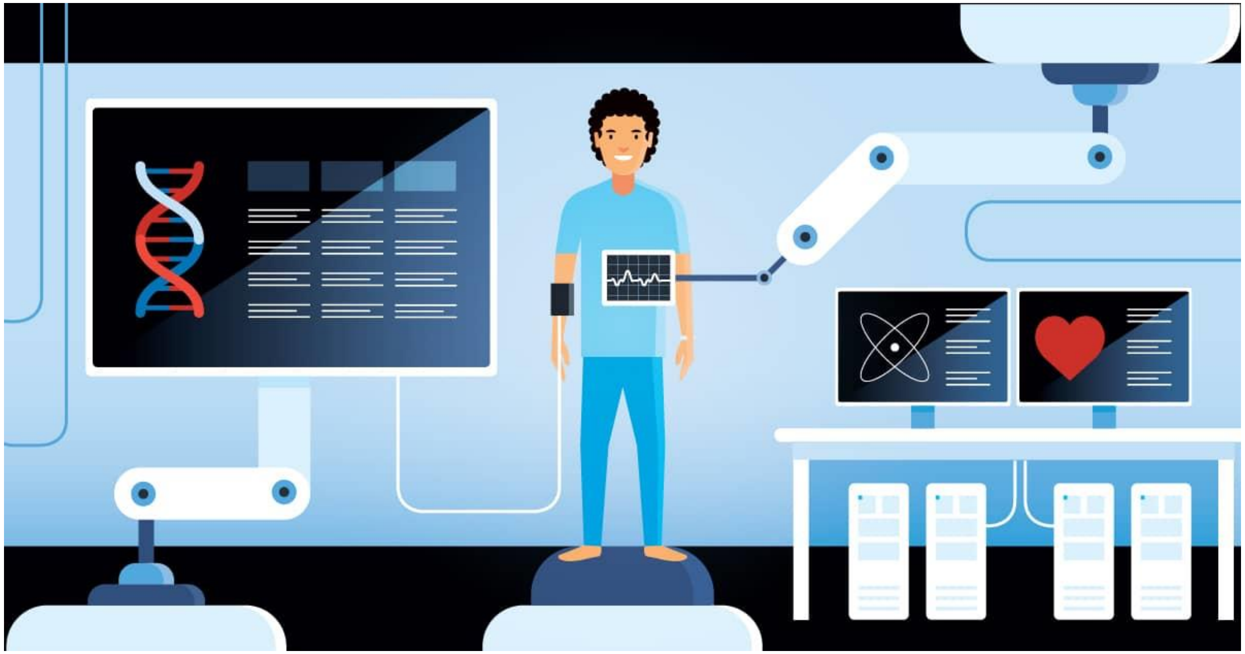


Image source: https://www.healthinformatics.uic.edu

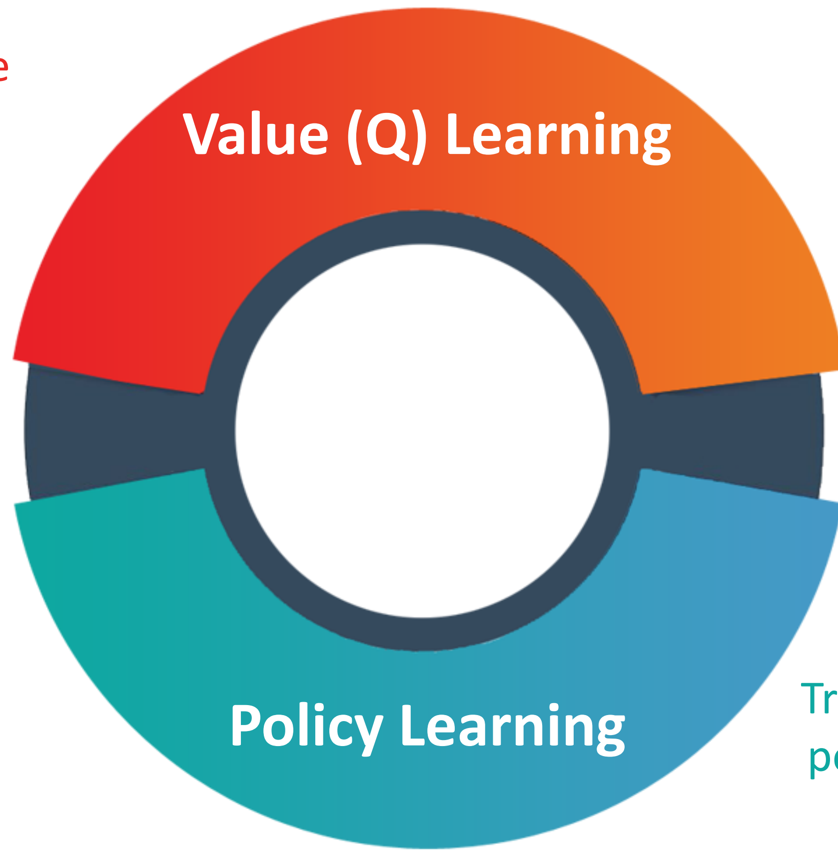
Applications of RL

Computer Games

Image source: https://www.pcmag.com

Deep RL Algorithms

Trying to calculate the Q-function instead of storing policies, and picking the action with max value



Trying to directly learn the policy (state to action mapping) showing what actions to take/avoid

Deep RL Algorithms

Value Learning (Q-Learning) Networks

- Calculating a cumulative score for each state (cheat sheet) and choosing the states with most possible reward
- Learning which actions should be performed at the current state to get maximum reward
  - **Example#1:** accelerate + →
  - **Example#2:** accelerate + ←
  - **Example#3:** brake + →
  - **Example#4:** brake + ←

Image source: https://www.retrogame.cc

Deep RL Algorithms

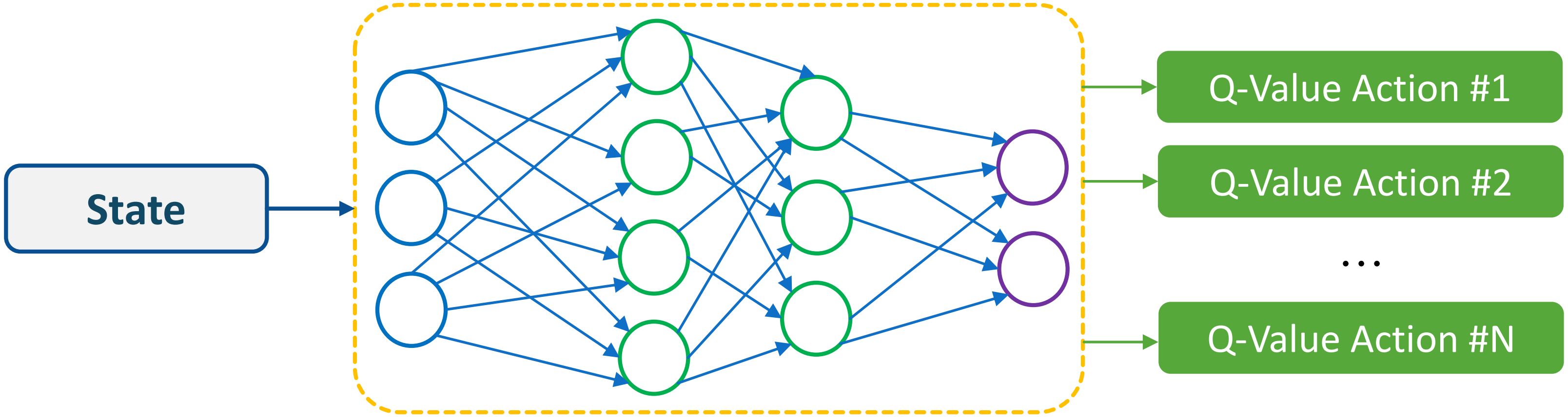
Policy Learning Networks

- Building a representation of a policy and keep it in memory during learning
- Learning to give a definite output by giving a particular input
  - **Example#1:** $\boldsymbol{action_1}$ will always result in $\boldsymbol{state_1}$
  - **Example#2:** staying on the road will always increase the $\boldsymbol{score}$
  - **Goal:** learn how to use $\boldsymbol{action_1}$ to stay on the road

Image source: https://www.retrogame.cc

Deep Q-Learning

- **Goal:** to provide a cheat sheet for the agent to find the best action
  - We need more powerful infrastructure to control thousands of (states, actions)
- A Deep Q-Network (DQN) is designed for challenging scenarios!
  - Using DNNs to approximate the Q-value function
  - **Input**: current (candidate) state, **Output**: the Q-value of all possible actions



Deep Q-Learning

Important Notes on DQNs

- All the past experiences should be stored in memory
- The next action is actually the maximum output of the Q-network
- The loss function in a DQN is MSE of the predicted and target Q-value

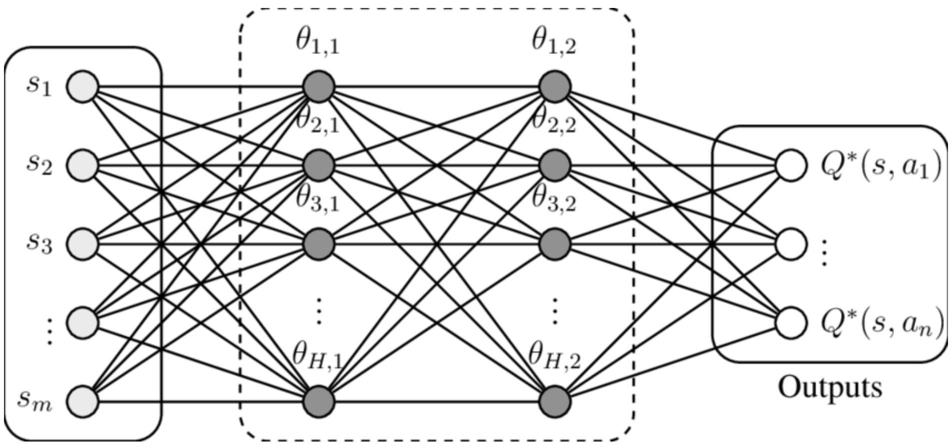


Image source: https://www.researchgate.net

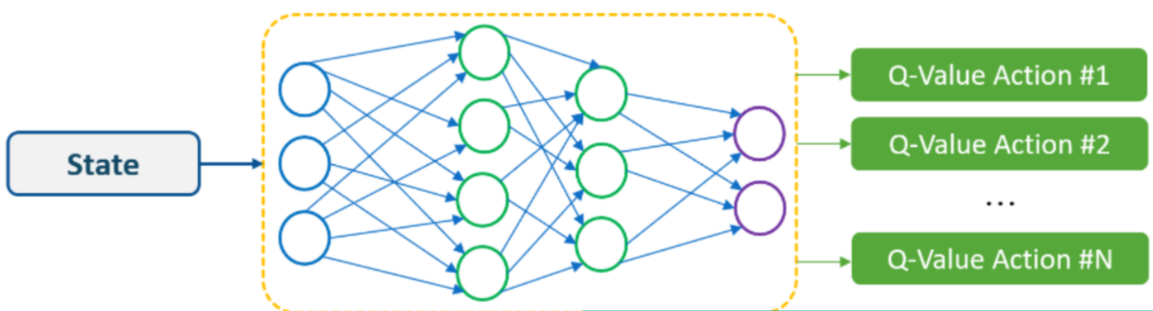
Deep Q-Learning

16 lines (11 sloc) | 938 Bytes

Raw Blame

Deep Q-Networks (DQNs)

Reinforcement Learning is the usage of DNNs to interact with real dynamic environments and training models to make a sequence of decisions. A Deep Q-Network is a common architecture for RL. The main goal in DQNs it to provide a cheat sheet for the agent to find the best action! DQNs are used to approximate the Q-value function, where the input is current (candidate) state, and the output is the Q-value of all possible actions



Codes

#	File	Description
0	Basic introduction	To be added soon

Full code on GitHub

References

- http://introtodeeplearning.com/
- https://towardsdatascience.com/policy-networks-vs-value-networks-in-reinforcement-learning-da2776056ad2
- https://deepsense.ai/what-is-reinforcement-learning-the-complete-guide/
- https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/

Questions?