Introduction to Deep Reinforcement Learning

Francesco Pugliese, PhD

neural1977@gmail.com

We will discuss about…

- ✓ **Reinforcement Learning**
- ✓ **Introduction to Markov Decision Process**
- ✓ **OpenAI Gym**

Reinforcement Learning

- ✓ **Reinforcement Learning (RL)** is a field of **Machine Learning (ML).** Differently to the other approaches such as **Supervised** and **Unsupervised** learning, Reinforcement Learning models adopts the "trial-and-error" technique by interacting with its environment.

- ✓ We may say that RL models have a "**body**" acting within an environment, other than a "**mind**", unlike traditional supervised and unsupervised machine learning which **only** have a "**mind**".

- ✓ Among the AI Researchers Community, there is the belief that RL will be the successful solution to build an **Artificial General Intelligence** finally.

- ✓ The main reason for this Reinforcement Learning success in recent years, is due to the rise of **Deep Reinforcement Learning (DRL)** which is a composition of Deep Learning (so Deep Neural Networks) and Reinforcement Learning.

Main Key Concepts of DRL

- ✓ **Agent:** An **Agent** (or learner) is a model/algorithm which learns to display **intelligent behaviours** (from a human perspective) during its lifetime. Therefore, an agent is used to take **intelligent** choices. o make intelligent decisions. For example: a chess player is an agent since the player learns to make intelligent decisions (best moves) in order to win the game.

- ✓ **Environment:** The **Environment** is the world where the agent lives and interacts with. For instance, in the chess game, the **chessboard** is the environment.

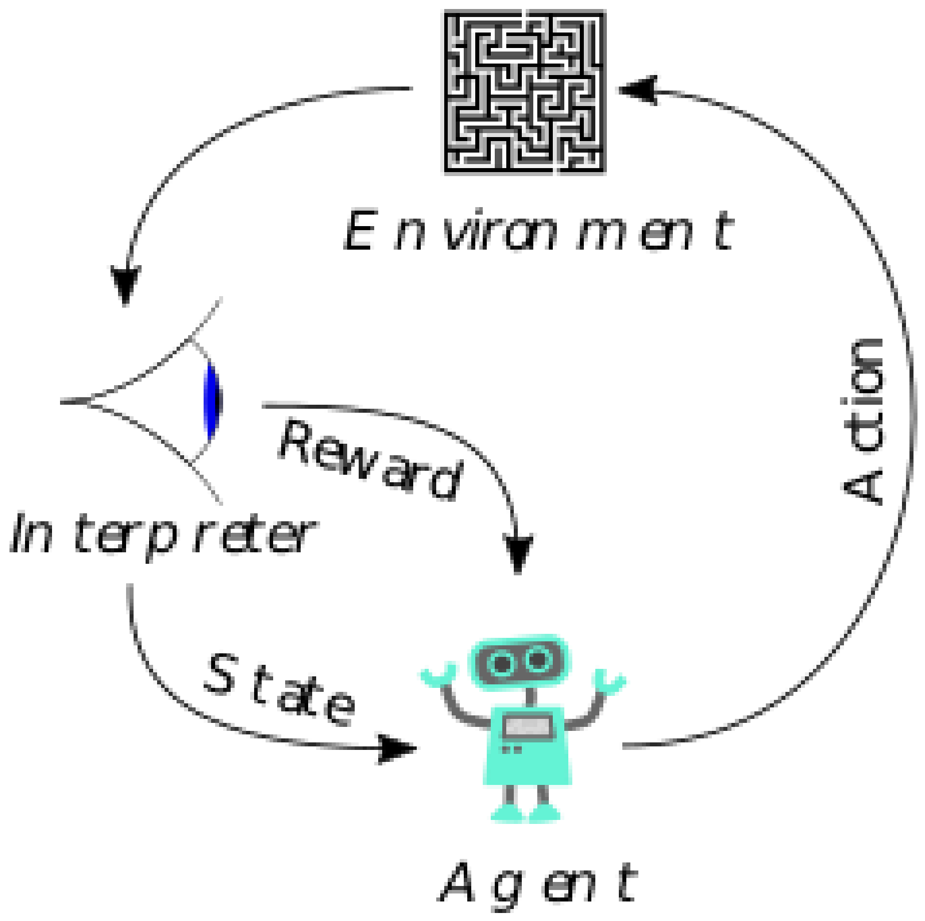
- ✓ **State:** A state is a position of the agent or a moment within the environment. For instance, in our chess game example, each position on the chessboard is called the state. The state is generally indicated with *s*.

- ✓ **Action:** During the interaction with the environment, the agent moves from one state to another by executing an action. In the chess game environment, the action is the move performed by the agent. In general, the action is denoted by *a*.

- ✓ **Reward:** According to the taken decision and the performed action, the agent receives a reward. A reward can be +1 for a good action and -1 for a bad action. In the chess game, if the agent makes a move in which it takes one of the opponent's chess pieces, then it is considered a good action and the agent receives a positive reward, it is negative, viceversa. The reward is denoted by *r*.

Idea behind DRL

- ✓ Similarly, an **RL** experimental setup, we will not teach the agent the actions the output to produce and te actions to perform, but will give a reward to the agent for every action it does.
- ✓ The agent will infer the needed output and actions autonomously, at the begin the agent is completely naïve performing random actions, then it becomes smarter thanks to rewards.
- ✓ That is why DRL is considered a trial-and-error algorithm.



Typical DRL Setting Steps:

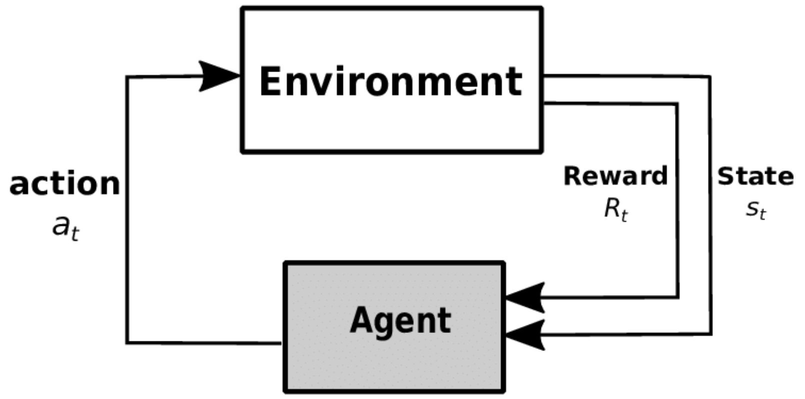
1) The agent interacts with the environment executing an action.

2) Th action taken by the agent leads it to move from one state to another.

3) Then the agent will receive a reward based on the action it performed.

4) According to the reward, the agent understands the goodness of the action, if it was good the agent will likely repeat the action, later on.

5) If the action was bad, the agent will try to perform other actions seeking for a positive reward.



Introduction to Markov Decision Process

- ✓ The **Markov Decision Process (MDP)** is a mathematical methodology for solving the Reinforcement Learning problems.
- ✓ The **Markov Assumption (or Property)** states that the future depends only on the present and not on the past. We already saw this property when we talked about recurrent neural networks.
- ✓ The **Markov Chain (or Markov Process)**, consists of a sequence of states that strictly obey the Markov property; in other words the Markov chain is the probabilistic model that solely depends on the current state to predict the next state and not the previous states.
- ✓ For example, a **Markov Process** may model **Weather Forecasting**: if we want to predict the weather and we know that the current state is cloudy, we can predict that the next state could be rainy. We may conclude that the next state is likely to be rainy only by considering the current state (cloudy) and not the previous states, which might have been sunny, windy, and so on.
- ✓ However, the **Markov Property** does not hold for all processes. For instance, throwing a dice (the next state) has no dependency on the previous number that showed up on the dice (the current state).

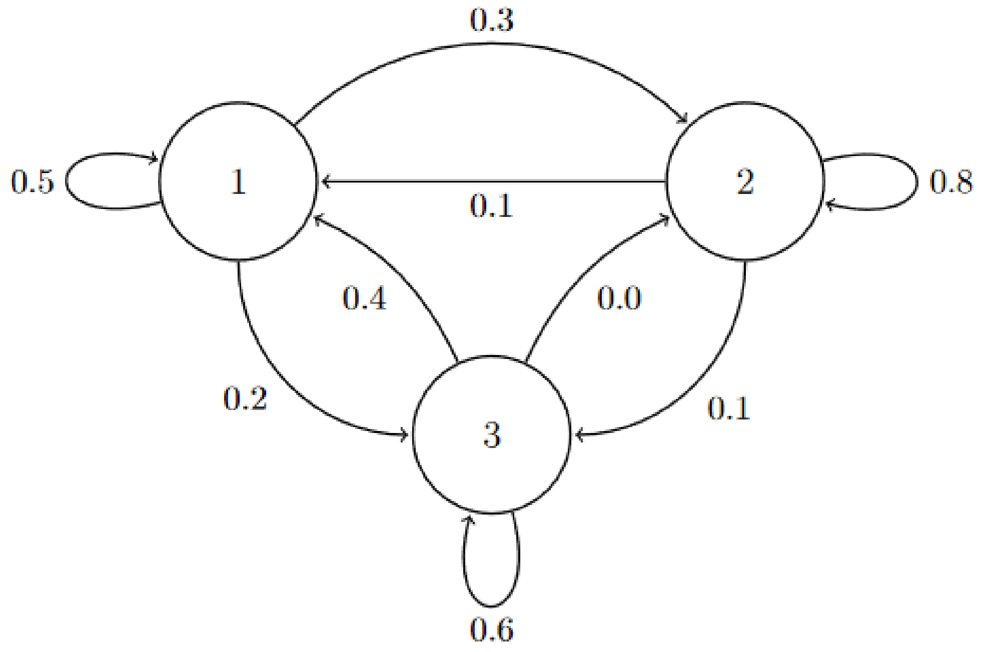
Markov Reward Process

- ✓ We can represent the transition information of the **Markov Chain** with a table or by a state diagram

- ✓ The **Markov Reward Process (MRP)** is an extension of the Markov chain with the reward function. That is, we learned that the Markov chain consists of **states** and a **transition** probability. Instead, the MRP consists of states, a transition probability, and also a **reward** function.

- ✓ A **reward** function tells us the reward we obtain in each state, this is denoted as $R(s)$.

- ✓ Is the Markov property applicable to the RL setting? Yes! In the **RL** environment, the agent makes decisions only based on the current state and not based on the past states. So, we can model an **RL** environment as an **MDP**.



OpenAI Gym Toolkit

- ✓ In 2015 **Elon Musk**, Sam Altman and other founded **OpenAI** with is a non-profit research organization in **Artificial Intelligence (AI)** with the purpose to build an Artificial General Intelligence (**AGI**). The company, considered a competitor to **DeepMind**, conducts experiments in the field of AI with the stated goal of promoting and developing friendly AI in a way that benefits humanity as a whole. Musk resigned from the board in February 2018 but remained a donor.

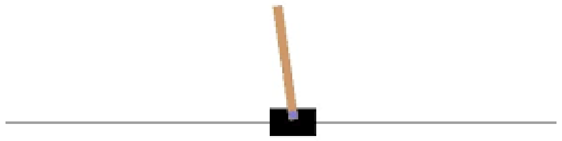
- ✓ Last **OpenAI** discover is **GPT-3,** which is an autoregressive model (based on Transformers) performing a wide variety of natural language tasks, and Codex, which translates natural language to code. GPT-3 was released after some years by OpenAI for fear of the consequences. This model made many experts think we achieved the "Singularity", namely a very True General Artificial Intelligence.-

- ✓ Furthermore, **OpenAI** provides a famous toolkit called Gym for training a reinforcement learning agent.

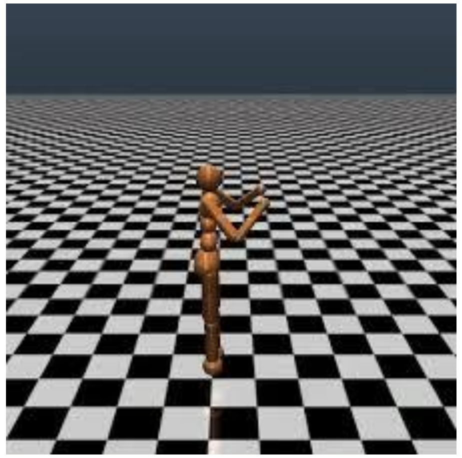
OpenAI Gym Toolkit

- ✓ Why should we need a Gym? When we want to train our agent to drive a car, for example, we cannot train it within a real-workd environment, but we need a **simulated environment**.

- ✓ Indeed, **Deep Reinforcement Learning** (DRL) is a trial-and-error learning process, so while we train our agent, it will make a lot of mistakes during learning and dangers. For example, let's suppose our agent hits another vehicle, and it receives a negative reward. It will then learn that hitting other vehicles is not a good action and will try not to perform this action again. But we cannot train the RL agent in the real-world environment by crashing it with other vehicles. We need a **simulator** as much real as we can.

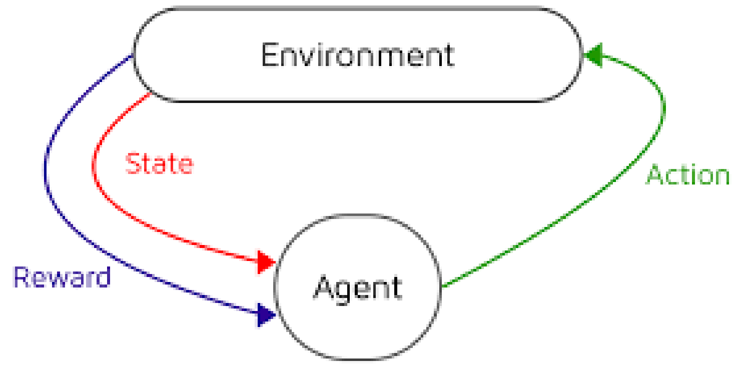


- ✓ There are a lot of **Simulators** for **RL,** but Gym is one of the best for the physics implemented and all the details.

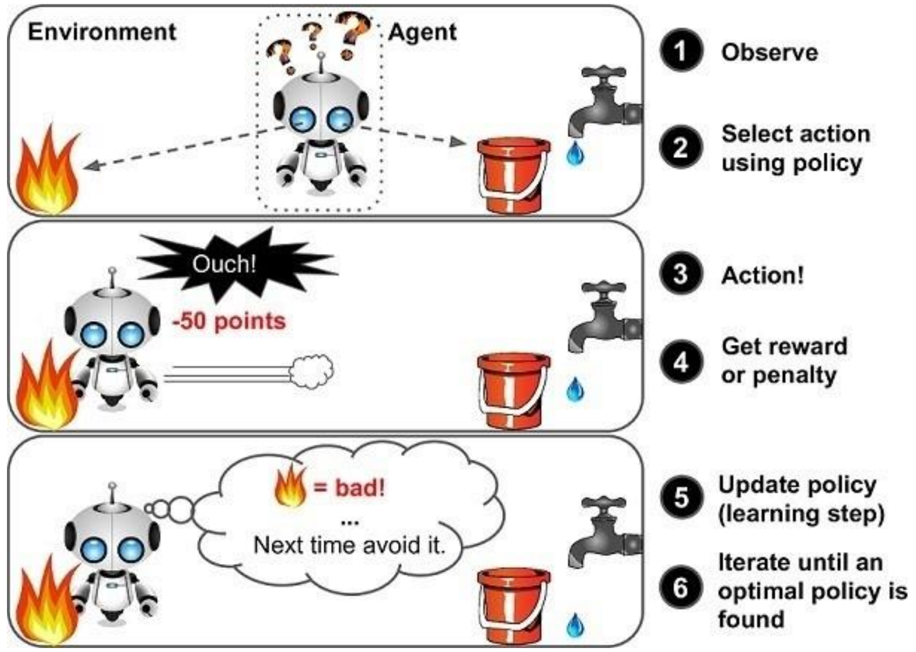


- ✓ Gym provides a variety of environments for training an RL agent ranging from classic control tasks to **Atari** game environments.

OpenAI Gym Chatacteristics



- ✓ Since every **RL Environment** can be modeled as a **Markov Decision Process (MDP)**, so every **OpenAI Gym Environment** is made of:
- ✓ **States:** Possible states of the environment
- ✓ **Actions:** Actions that the agent can perform in each state.
- ✓ **Transition Probability:** The transition probability is denoted by $P(s'|s,a)$ meaning the probability of moving from a state to the next state while performing an action *a.*
- ✓ **Reward function:** Reward function is denoted by $R(s,a,s')$ meaning that the agent obtains a reward *r* moving from a state to the next state while performing an action *a.*


Environment
Agent
1 Observe
2 Select action using policy
Ouch!
-50 points
3 Action!
4 Get reward or penalty
= bad!
...
Next time avoid it.
5 Update policy (learning step)
6 Iterate until an optimal policy is found

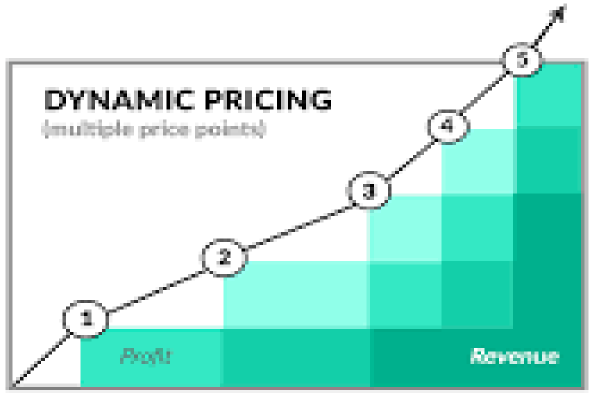
Applications of Deep Reinforcement Learning

- ✓ **Manufacturing**: In manufacturing, intelligent robots are trained using RL to place objects in the right position. The use of intelligent robots reduces labor costs and increases productivity.

- ✓ **Dynamic pricing**: Dynamic pricing implies that we change the price of products based on demand and supply. We can train the RL agent for the dynamic pricing of products with the goal of maximizing revenue.

- ✓ **Inventory management**: RL is used extensively in inventory management, which is a crucial business activity. Some of these activities:
  - I. Chain management,
  - II. Demand forecasting,
  - III. Handling of several warehouse operations such as placing products in warehouses or managing space efficiently.

- ✓ **Self-Driving Cars**: **DRL** bosted this field in recent years



Applications of Deep Reinforcement Learning

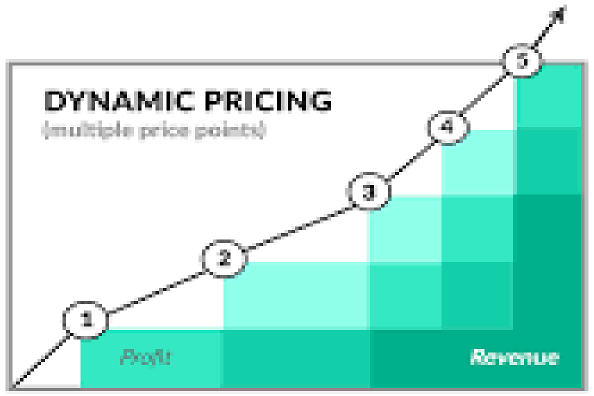
- ✓ **Dynamic Recommendation Systems:** RL is widely used in building a recommendation system where the behavior of the user constantly changes.

- ✓ **Neural Architecture Search (NAS):** this task consists in searching complex neural architectures by training the agent to find the best neural topology, better than human designed ones. **Nas** is the technique adopted by the **NasNet**, Neural Architecture Search-Network.

- ✓ **Natural Language Processing (NLP): DRL** can be used in many NLP tasks, such as abstractive text summarization, chatbots, and so on.

- ✓ **Finance**: **DRL** can be used in financial portfolio management, prediction and trading in commercial transaction markets.

- ✓ And then: **Drones**, **Autonomous Robotics**, etc.



References

Arulkumaran, K., Deisenroth, M. P., Brundage, M., & Bharath, A. A. (2017). Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, *34*(6), 26-38.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*.

Francesco **Pugliese**