Reinforcement Learning

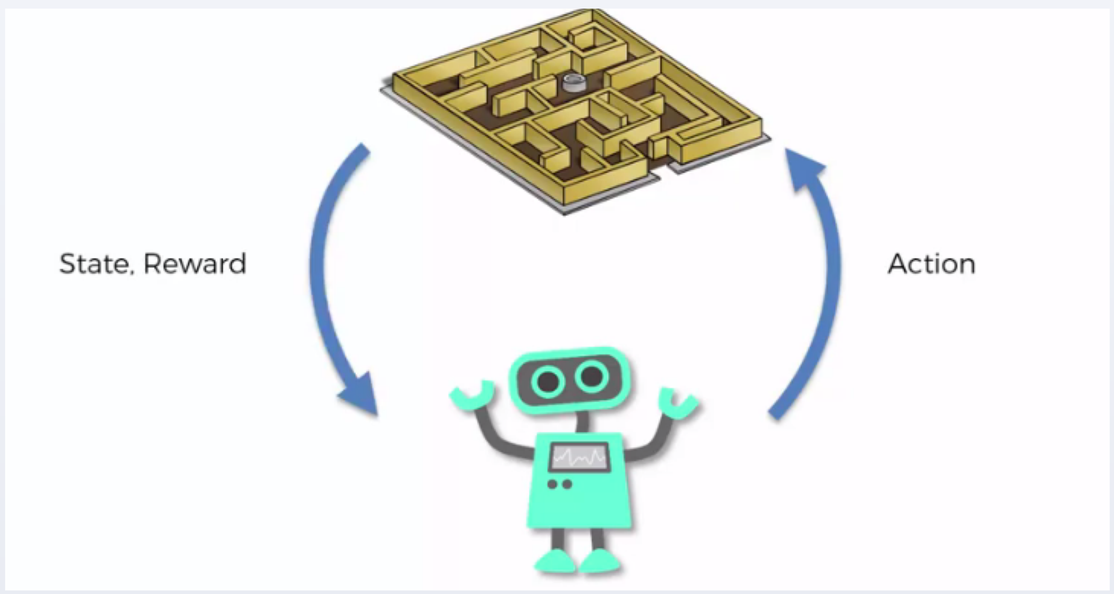
Student(s)
A. Atakan Demir

Faculty Member(s)
Prof. Berrin Yanıkoğlu

Sabancı Üniversitesi

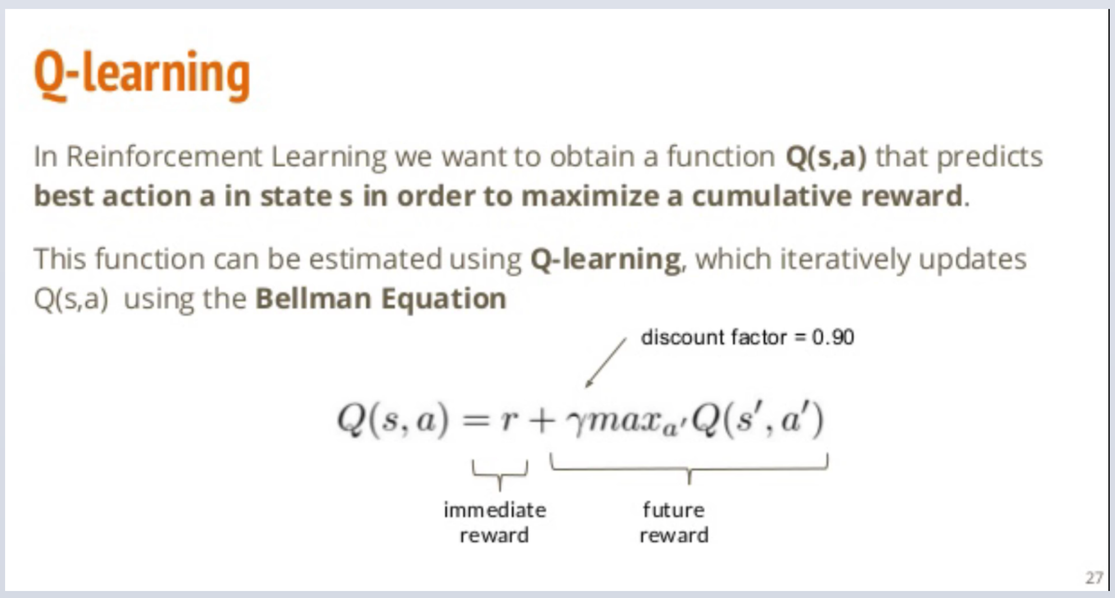
PURE
PROGRAM FOR UNDERGRADUATE RESEARCH

Abstract



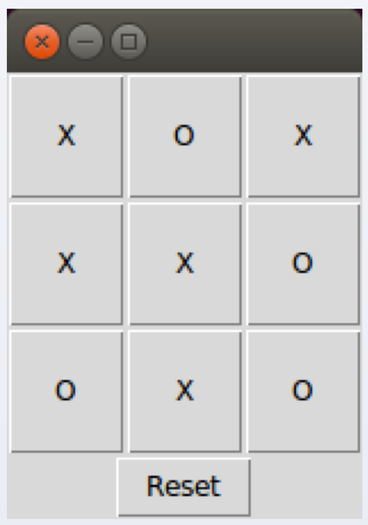
Reinforcement learning, in context of artificial intelligence, is a type of dynamic programming that trains algorithms using a system of reward and punishment. The technique is so similar to how a kid learns to perform a new task actually. You do not teach the agent how to play the game for example, you do not say ‘You should do this, when you come here and you should go there when you see this’. You are just giving + or - points or zero for each move of the agent. So basically, reinforcement learning contrasts with other machine learning approaches in that the algorithm is not explicitly told how to perform a task, but works through the problem on its own. As an agent, which could be a self-driving car or a program playing chess, interacts with its environment, receives a reward state depending on how it performs, such as driving to destination safely or winning a game. Conversely, the agent receives a penalty for performing incorrectly, such as going off the road or being checkmated. The agent over time makes decisions to maximize its reward and minimize its penalty using dynamic programming. The advantage of this approach to artificial intelligence is that it allows an AI program to learn without a programmer spelling out how an agent should perform the task which is cool.

Project Abstract I



Q-learning was introduced by Watkins in 1989. And it uses Belman Equation. Q-learning is an algorithm which produces a Q table that an agent uses to find the best action to take given a state. Q learning ‘at its singlest’ uses tables to store data ‘reward and penalty points’. The goal of Q-Learning is to learn a policy, which tells an agent which action to take under which circumstances. So it does not require a model of the environment and can handle problems with stochastic transitions and rewards, without requiring adaptations.

Project Abstract II



In our game, Tic Tac Toe, at each discrete time step t, the state s of the system is defined by the marks on the board and which player's turn it is, and the available actions a by the empty squares on the board. It is important finding a policy that states belong to action which tells us which action to take in which state to maximize our chance of winning.

Given a policy pi, at any given time each state has a certain *value* V to the pi, which is the expected discounted reward from following that policy for all future time: where rt = rt(st,at) is the *reward* at time step t and Gamma represents Gamma, *discount factor*.

In the implementation of Tic Tac Toe, 'sign convention’ is important. For player "X" , our reward is positive -- specifically, a reinforcement of 1.0 is awarded when player "X" wins, -1.0 when player ”O" wins, and 0.5 in the case of a tie. Therefore, player "X" aims to maximize the value, while player "O" aims to minimize it. The discount factor Gamma is given a value of 0.9.

Conclusions

References

https://skymind.ai/wiki/deep-reinforcement-learning
http://mnemstudio.org/path-finding-q-learning-tutorial.htm
https://medium.freecodecamp.org/an-introduction-to-reinforcement-learning-4339519de419