Reinforcement learning :

How to solve the Frozen Lake problem with Q-Learning

Brief description of the project:

The aim of this project is to implement a reinforcement algorithm : Reinforcement Q-learning with an $\epsilon - greedy$ training. To do so we will work on the Frozen Lake problem.

Environment Gym

Frozen-Lake is an open source environnment proposed by OpenAI Gym:

The agent controls the movement of a character in a grid world. Some tiles of the grid are walkable, and others lead to the agent falling into the water. Additionally, the movement direction of the agent is uncertain and only partially depends on the chosen direction. The agent is rewarded for finding a walkable path to a goal tile.

Policies:

Random moves:

First we will compute an agent choosing randomly his moves and see how it performs.
For a total of 1000 games simulated, the agent moving randomly played 31428 moves. An average of 31.428 moves per game.
His average reward per game is around : 0.001 which is really small. (A success giving a reward of 1).

The function **print_frames** allows to vizualise the different games dynamicaly.

As we might expected, a random strategy is not really efficient. The agent ends in a hole most of the games.
Let's start learning!

Reinforcement Q-learning:

In order to improve the policy of the agent, we use a Q-learning algorithm.
We define the Q-table.
Thus, for every state the agent follows an $\epsilon - greedy$ strategy: ϵ% of the time he chooses a movement randomly. The rest of the time he chooses the $Argmax_{action}(Q\_table(state, action))$ The the Q-table is updated based on the **Bellman equation**:

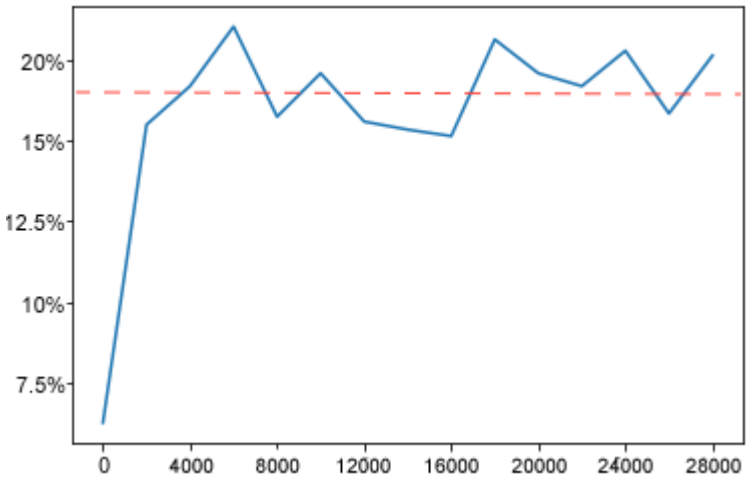
$$Q(state, action) \leftarrow (1 - \alpha)Q(state, action) + \alpha(reward + \gamma \max_{action}(next\_state, all\_actions))$$

Results and hyperparameters :

first results:

Hyper Parameters :

- **Learning rate** $\alpha = 0.1$
- **Discount factor** $\gamma = 0.6$
- **Exploration rate** $\epsilon = 0.1$

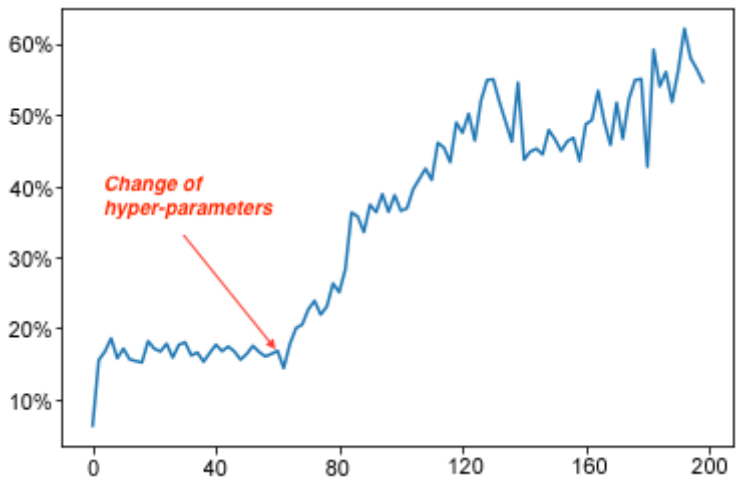


As we might see, the success rate seems to quickly stagnate around 15 to 20%. The learning rate α and the discount factor γ might be too small.
Moreover, I suspect an exploration ratio too important so we might consider decreasing ϵ.

Results of an improve of the hyper parameters :

Hyper Parameters :

- **Learning rate** $\alpha = 0.8$
- **Discount factor** $\gamma = 0.9$
- **Exploration rate** ϵ = starting at 0.07 and decreasing until reaching 1% of exploration



Thus, we can notice that this modification of the hyper parameters lead to an improve of the learn. The agent policy keeps improving, and eventually reach 60% of success after 200 thousands episodes.

Considering the problem, this success rate seems quit fair. Indeed, not every actions succeed and sometime a good choice of action in a given state might lead to a hole. (Ice is slippery and the probability of achieving an action properly is not that high).
However, despite this fair success rate, the training appears to be quit slow.