Deep Reinforcement Learning AI

Maestracci V., Bardes A., Dezerces E.

January, 2019

Overview

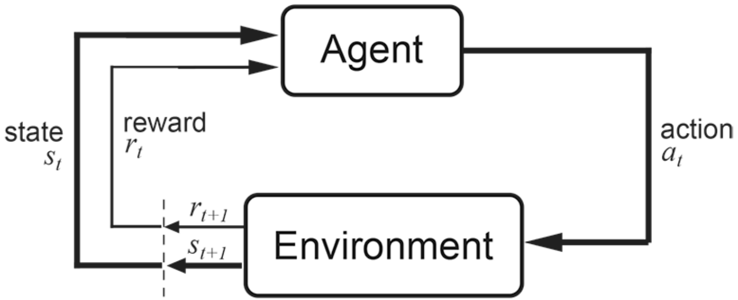
Q-Learning

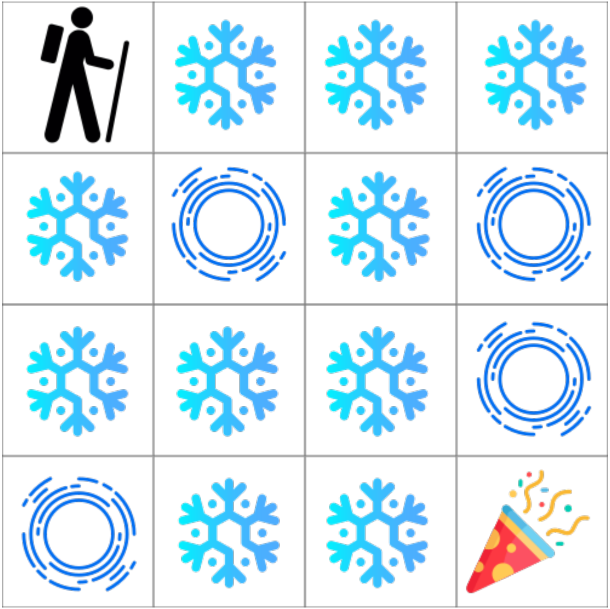
Deep Q-Learning

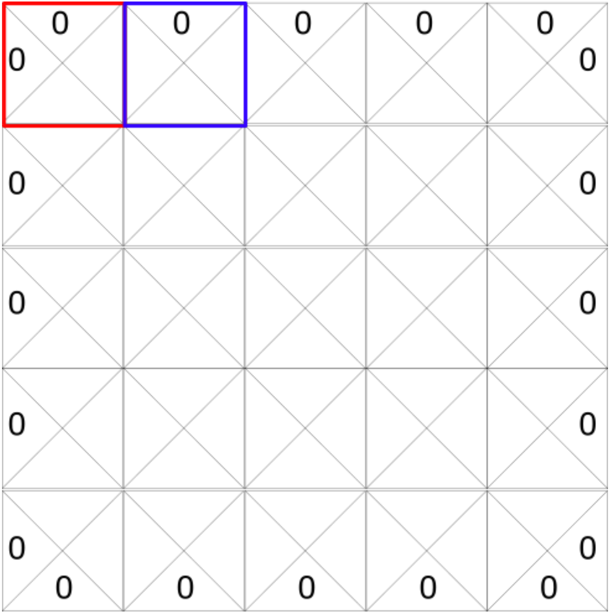
Deep Q-Learning : improvements

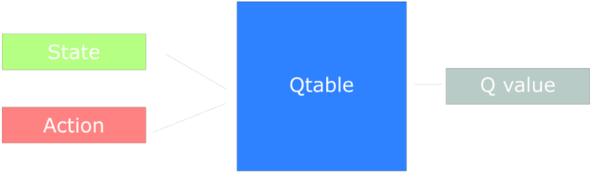
Our project

The principle of Q-learning






0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0
0



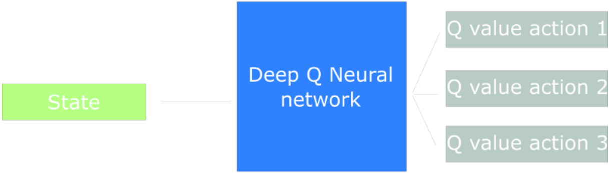
State

Action

Qtable

Q value

Q* learning



State

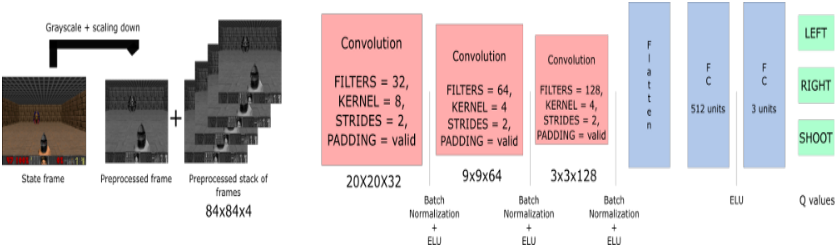
Deep Q Neural network

Q value action 1

Q value action 2

Q value action 3

Deep Q* learning


Grayscale + scaling down
State frame
Preprocessed frame
+
Preprocessed stack of frames
84x84x4
Convolution
FILTERS = 32,
KERNEL = 8,
STRIDES = 2,
PADDING = valid
20X20X32
Batch Normalization + ELU
Convolution
FILTERS = 64,
KERNEL = 4
STRIDES = 2,
PADDING = valid
9x9x64
Batch Normalization + ELU
Convolution
FILTERS = 128,
KERNEL = 4,
STRIDES = 2,
PADDING = valid
3x3x128
Batch Normalization + ELU
Flatten
FC
512 units
ELU
FC
3 units
LEFT
RIGHT
SHOOT
Q values

$$\underbrace{\Delta w}_{\text{Change in weights}} = \underset{\text{learning rate}}{\alpha}[\underbrace{(\underbrace{R + \gamma\ max_a\ \hat{Q}(s', a, w}_{\text{Maximum possible Qvalue for the next\_state (= Q\_target)}})) - \underbrace{\hat{Q}(s, a, w)}_{\text{Current predicted Q-val}}]}_{\text{TD Error}}\ \underbrace{\nabla_w \hat{Q}(s, a, w)}_{\text{Gradient of our current predicted Q-value}}$$

Memory

What if our AI goes in level two?

Memory

What if our AI goes in level two?
It musn't forget about what it learned before !

Memory

What if our AI goes in level two?
It musn't forget about what it learned before !
So let's save our previous experiences.

Fixed Q-target

We are chasing a moving target :

Fixed Q-target

We are chasing a moving target :

How can we make it so that the target doesn't move too much?

Fixed Q-target

We are chasing a moving target :

How can we make it so that the target doesn't move too much?
We can have two different networks!

Double DQN

At the beginning of the learning, our q values are noisy.
What if we favor sub-optimal choices ?

Double DQN

At the beginning of the learning, our q values are noisy.
What if we favor sub-optimal choices ?
$\longrightarrow$ We must decouple the action selection from the q-value generation.
For that we can use our two networks!

$$\underbrace{Q(s,a)}_{\text{TD target}} = r(s,a) + \gamma Q(s', \underbrace{argmax_a Q(s',a)}_{\text{DQN Network choose action for next state}})$$

TD target

DQN Network choose action for next state

Target network calculates the Q value of taking that action at that state

Dueling DQN

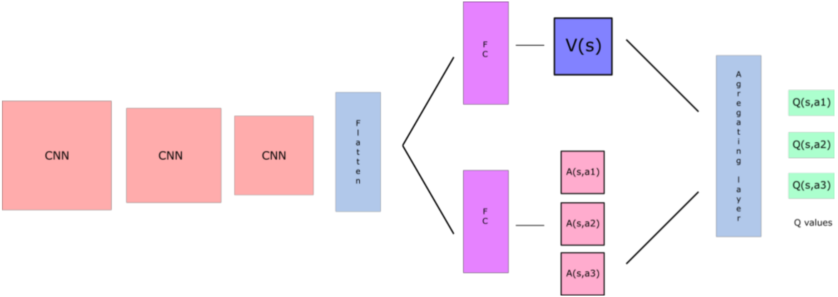
A Q value represents two things :

- How good it is to be in state s.
- How good is it to take the action a in that state.

So we can split it in two : V and A

Dueling DQN

This is the answer to the question : what is the point of knowing $Q(s,a)$ for every a when the state s is bad



Also adds decoupling.

PER

Experiences with a huge loss are more important than others.

PER

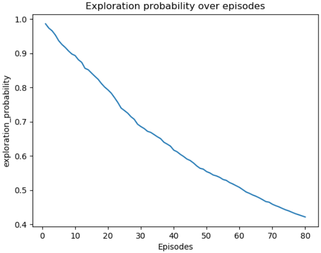
Experiences with a huge loss are more important than others. $\longrightarrow$ So let's prioritize these.

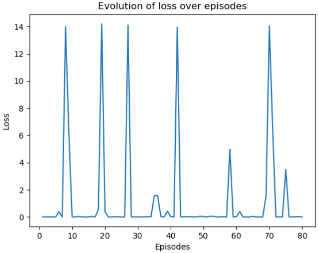
PER

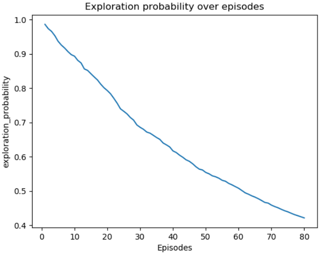
- Add a non uniform probability to be chosen for experience replay.
- Probability of being chosen decrease if you are often chosen

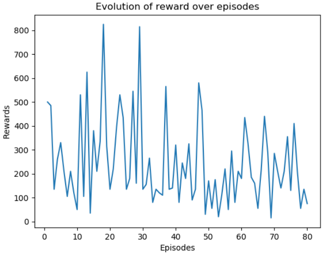
Unfortunately...

Our bot isn't really good...


Exploration probability over episodes


Evolution of loss over episodes


Exploration probability over episodes
exploration_probability
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
10
20
30
40
50
60
70
80
Episodes


Evolution of reward over episodes
Rewards
800
700
600
500
400
300
200
100
0
0
10
20
30
40
50
60
70
80
Episodes

Our Github:
https://github.com/Adrien987k/Deep-Space-Invaders