Deep Reinforcement Learning

Scientific Initiation

Lucas Emanuel Resck Domingues

Escola de Matemática Aplicada
Fundação Getulio Vargas

March 2020

Table of Contents

Table of Contents

Motivation

- I wanted to do Scientific Initiation in Machine Learning;
- Reinforcement Learning is a quite promising area, because of the potential of applications;
- Deep Learning is one of the most powerful Machine Learning areas, because DNN can reach such degree of generalization;
- It seems that Deep Reinforcement Learning has very impressive powerful applications and it's on mainstream.

Intuition

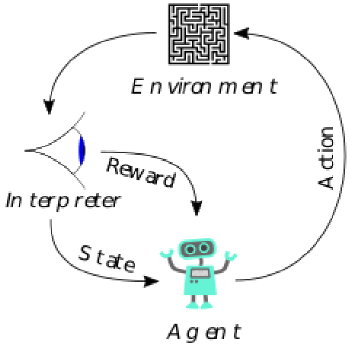


Figure 1: Reinforcement Learning example (Wikipedia)

Table of Contents

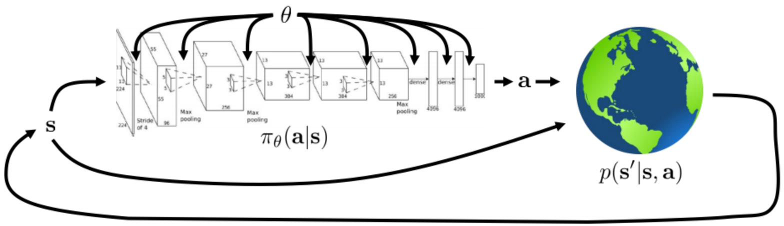
Markov Decision Process

Markov Decision Process is a 4-tuple (S, A, T, r):

- S: space of states s;
- A: space of actions a;
- T: transition operator, with probabilities $P(s_{t+1}|s_t, a_t)$;
- r: reward function $S \times A \to \mathbb{R},\ (s, a) \mapsto r(s, a)$.

The goal of Reinforcement Learning

Figure 2: Reinforcement Learning world (Deep RL Berkeley Course).



θ
$\mathbf{s}$
$\mathbf{a}$
$\pi_\theta(\mathbf{a}|\mathbf{s})$
$p(\mathbf{s}'|\mathbf{s},\mathbf{a})$

$$p_\theta(\tau) = p_\theta(s_1, a_1, \cdots, s_T, a_T) = p(s_1)\prod_{t=1}^{T}\pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)$$

$$\theta^* = \arg\max_\theta E_{\tau\sim p_\theta(\tau)}\left[\sum_t r(s_t, a_t)\right]$$

Too many algorithms

- Policy iteration;
- Value iteration;
- Q-learning;
- Deep Q-learning.

More definitions

Q-Function

$$Q^{\pi}(s_t, a_t) = \sum_{t'=t}^{T} E_{\pi_\theta}[r(s_{t'}, a_{t'})|s_t, a_t]$$

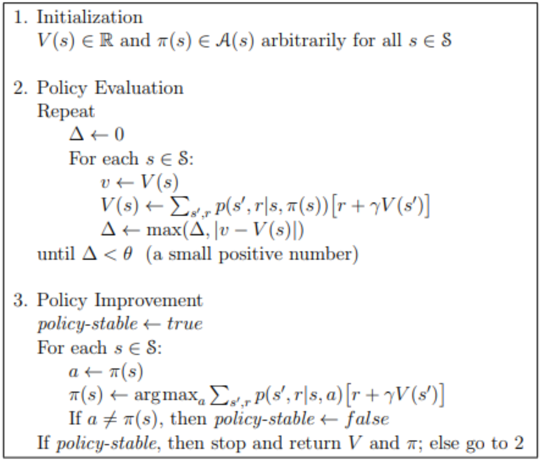
The total expected reward for taking an action a_t when the state is s_t.

Value Function

$$V^{\pi}(s_t) = E_{a_t \sim \pi_\theta(a_t|s_t)}[Q^{\pi}(s_t, a_t)]$$

The total expected reward when the state is s_t.

Policy iteration



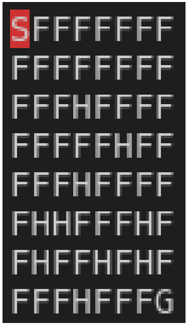
```
1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
       $\Delta \leftarrow 0$
       For each $s \in \mathcal{S}$:
           $v \leftarrow V(s)$
           $V(s) \leftarrow \sum_{s',r} p(s',r|s,\pi(s))\big[r + \gamma V(s')\big]$
           $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$  (a small positive number)

3. Policy Improvement
   *policy-stable* $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
       $a \leftarrow \pi(s)$
       $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
       If $a \neq \pi(s)$, then *policy-stable* $\leftarrow$ *false*
   If *policy-stable*, then stop and return $V$ and $\pi$; else go to 2
```

Figure 3: Policy iteration (Sutton and Andrew, Reinforcement Learning: An Introduction, 2nd edition).

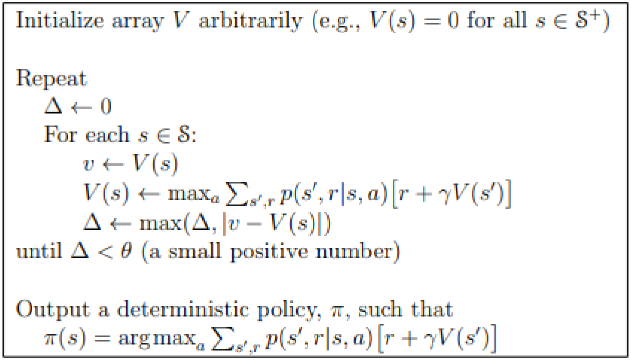
Policy iteration example: FrozenLake8x8



```
SFFFFFFF
FFFFFFFF
FFFHFFFF
FFFFFHFF
FFFHFFFF
FHHFFFHF
FHFFHFHF
FFFHFFFG
```

Figure 4: FronzenLake8x8 environment.

Value iteration



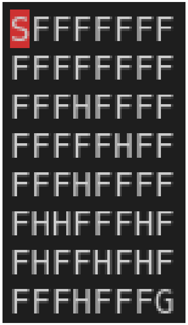
```
Initialize array $V$ arbitrarily (e.g., $V(s) = 0$ for all $s \in \mathcal{S}^+$)

Repeat
    $\Delta \leftarrow 0$
    For each $s \in \mathcal{S}$:
        $v \leftarrow V(s)$
        $V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$ (a small positive number)

Output a deterministic policy, $\pi$, such that
    $\pi(s) = \arg\max_a \sum_{s',r} p(s',r|s,a)\big[r + \gamma V(s')\big]$
```

Figure 5: Value iteration (Sutton and Andrew, Reinforcement Learning: An Introduction, 2nd edition).

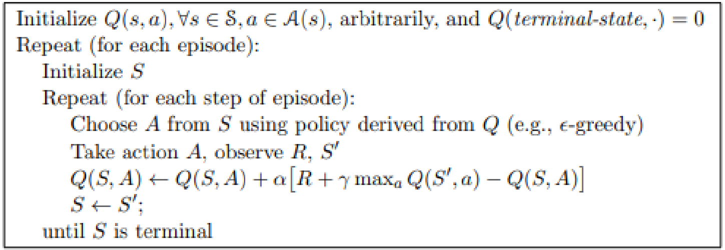
Value iteration example: FrozenLake8x8



```
SFFFFFFF
FFFFFFFF
FFFHFFFF
FFFFFHFF
FFFHFFFF
FHHFFFHF
FHFFHFHF
FFFHFFFG
```

Figure 6: FronzenLake8x8 environment.

Q-Learning



```
Initialize $Q(s, a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
    Initialize $S$
    Repeat (for each step of episode):
        Choose $A$ from $S$ using policy derived from $Q$ (e.g., $\epsilon$-greedy)
        Take action $A$, observe $R$, $S'$
        $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
        $S \leftarrow S'$;
    until $S$ is terminal
```

Figure 7: Q-Learning (Sutton and Andrew, Reinforcement Learning: An Introduction, 2nd edition).

Q-Learning example: MountainCar

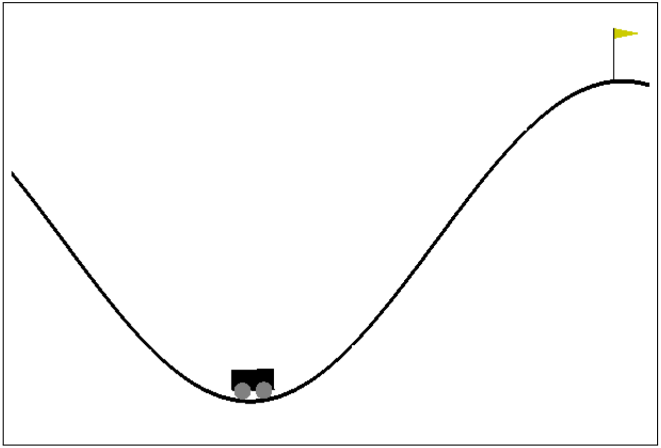


Figure 8: MountainCar enviroment.

Approximate Q-Learning

We assume the existence of a feature function $f : (s, a) \mapsto f_1(s, a), \cdots, f_n(s, a)$, with $f_i(s, a)$ a feature value.

Approximate Q-Function

$$Q(s, a) = \sum_{i=1}^{n} f_i(s, a) w_i$$

Approximate Q-learning iteration

$$w_i \leftarrow w_i + \alpha \cdot \mathit{difference} \cdot f_i(s, a)$$

$$\mathit{difference} = (r + \gamma \max_{a'} Q(s', a')) - Q(s, a)$$

Approximate Q-Learning example: PacMan

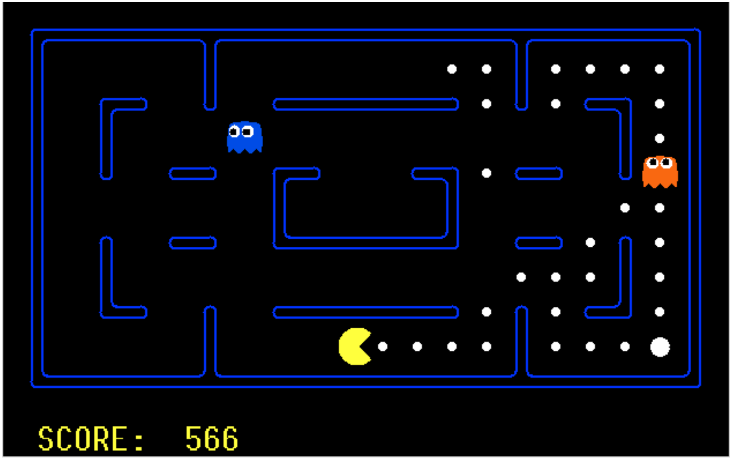
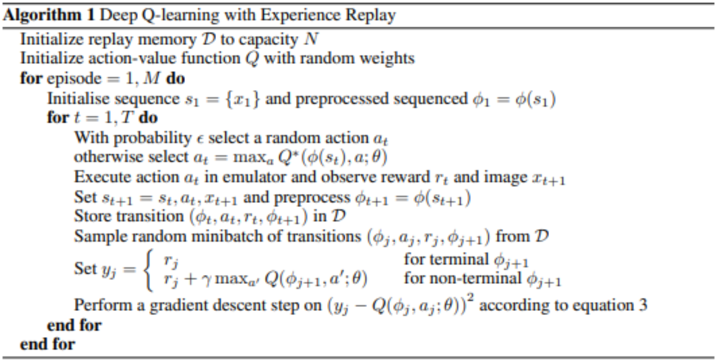


Figure 9: PacMan environment (Berkeley Intro to AI).

Table of Contents

Deep Q-Learning with Experience Replay



Algorithm 1 Deep Q-learning with Experience Replay

```
Initialize replay memory $\mathcal{D}$ to capacity $N$
Initialize action-value function $Q$ with random weights
for episode $= 1, M$ do
    Initialise sequence $s_1 = \{x_1\}$ and preprocessed sequenced $\phi_1 = \phi(s_1)$
    for $t = 1, T$ do
        With probability $\epsilon$ select a random action $a_t$
        otherwise select $a_t = \max_a Q^*(\phi(s_t), a; \theta)$
        Execute action $a_t$ in emulator and observe reward $r_t$ and image $x_{t+1}$
        Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
        Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
        Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$
        Set $y_j = \begin{cases} r_j & \text{for terminal } \phi_{j+1} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a'; \theta) & \text{for non-terminal } \phi_{j+1} \end{cases}$
        Perform a gradient descent step on $(y_j - Q(\phi_j, a_j; \theta))^2$ according to equation 3
    end for
end for
```

Figure 10: Deep Q-Learning algorithm (DeepMind, Playing Atari with Deep Reinforcement Learning).

Deep Q-Learning with Experience Replay example: MountainCar

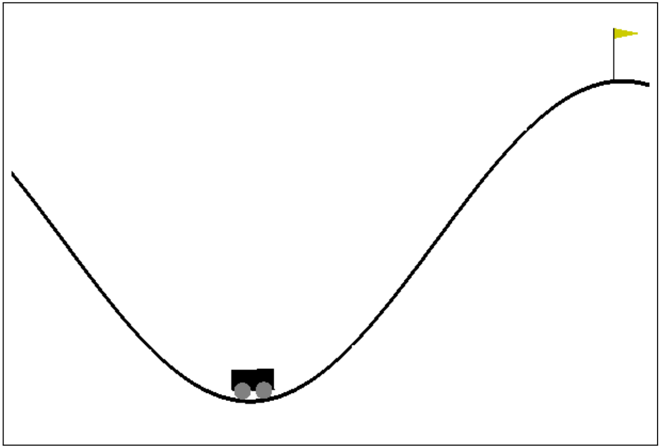


Figure 11: MountainCar enviroment.

Table of Contents

Deep Q-Learning + CNN

- Learn from a game image;
- Use convolutional neural networks to learn to treat this image.

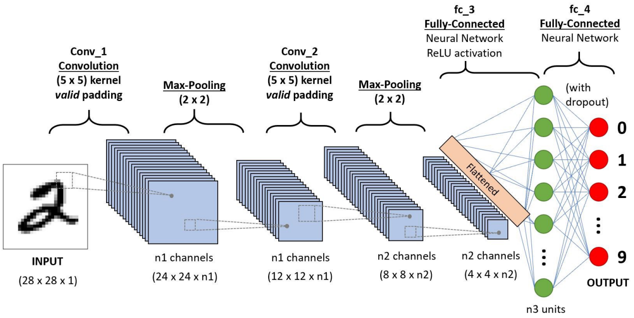


Figure 12: Convolutional Neural Network example (Saha, A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way).

Atari 2600

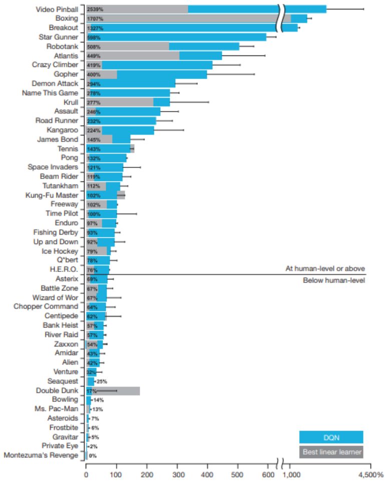


Figure 13: Comparison of DQN with linear methods, both normalized by professional players (100%) and random players (0%) (Human-level control through deep reinforcement learning).

Atari 2600

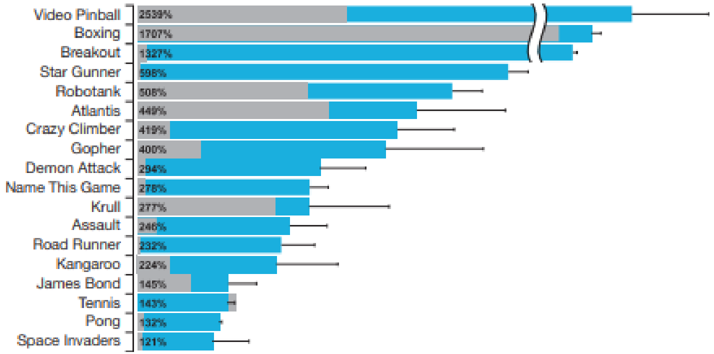


Figure 14: Comparison of DQN with linear methods, both normalized by professional players (100%) and random players (0%) (Human-level control through deep reinforcement learning).

Super Mario Bros.

Figure 15: Super Mario Bros. Open AI environment (PyPI).

Table of Contents

Partial considerations

- Reinforcement Learning & Deep Reinforcement Learning today consist in many algorithms, so that many chalenging problems can be solved with them;
- We have today different powerful world applications, and many great things are being done.
- There are many paths to follow in order to learn and to research.

Thank you

"Instead of trying to produce a program to simulate the adult mind, why not rather try to produce one which simulates the child's? If this were then subjected to an appropriate course of education one would obtain the adult brain."

— Alan Turing