# Deep Reinforcement Learning with Double Q-learning

**Bio-Medical Computing Laboratory**

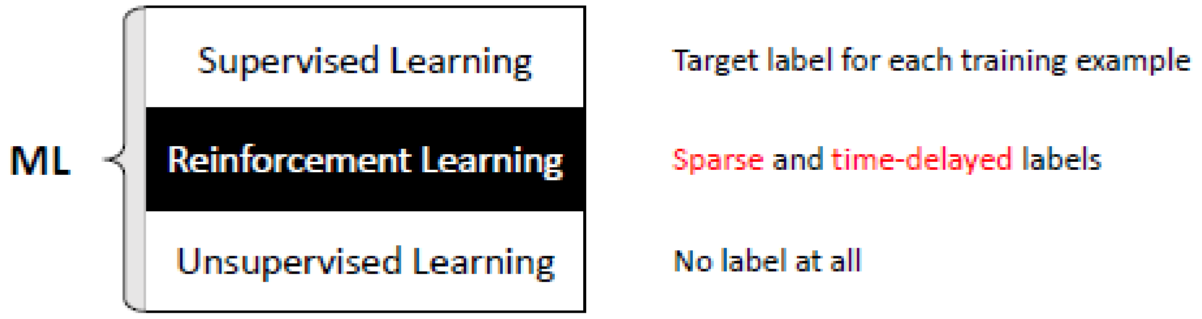
**CHO TAEHEUM**

**05th Mar, 2018**

Hado van Hasselt and Arthur Guez and David Silver. AAAI(2015, Impact factor :1.87), 316 times referred

## INDEX

## Background



ML

| | |
|---|---|
| Supervised Learning | Target label for each training example |
| Reinforcement Learning | Sparse and time-delayed labels |
| Unsupervised Learning | No label at all |

## Background

To solve sequential decision problems we can learn estimates for the optimal value of each action, defined as **the expected sum of future rewards**.

$$Q_\pi(s,a) \equiv \mathbb{E}\left[R_1 + \gamma R_2 + \ldots \mid S_0 = s, A_0 = a, \pi\right]$$

π : given policy
a : the true value of an action
s : a state
Γ : a discount factor

## Background

The optimal value equation is

$$Q_*(s,a) = \max_\pi Q_\pi(s,a)$$

An optimal policy is easily derived from the optimal values by selecting the highest valued action in each state.

## Background

It's too large to learn all action values in all states separately. Instead, we can learn a parameterized value function

$$Q(s, a; \boldsymbol{\theta}_t)$$

The standard Q-learning update for the parameters after taking **action(At)** in **state(St)** and observing the immediate **reward(Rt+1)** and resulting **state(St+1)** is then,

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha(Y_t^{\mathrm{Q}} - Q(S_t, A_t; \boldsymbol{\theta}_t))\nabla_{\boldsymbol{\theta}_t} Q(S_t, A_t; \boldsymbol{\theta}_t)$$

$$Y_t^{\mathrm{Q}} \equiv R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \boldsymbol{\theta}_t)$$

Target value

This update the current value $Q(S_t, A_t; \boldsymbol{\theta}_t)$ towards a target value $Y_t^{\mathrm{Q}}$.

## Background

Deep Q-Network provides a stable solution to deep value-based RL

1. Use experience replay
   - Break correlations in data
   - Learn from all past policies
   - Using off-policy Q-learning

2. Freeze target Q-network
   - Avoid oscillations
   - Break correlations between Q-network and target

$$Y_t^{\mathrm{DQN}} \equiv R_{t+1} + \gamma \max_a Q(S_{t+1}, a; \theta_t^-)$$

To avoid oscillations, fix parameters used in Q-learning target

## Background

### Double Q-learning

The max operator( $\max\limits_a$ ) in standard Q-learning and DQN, uses the same values both to select and to evaluate an action. This makes it more likely to select overestimated values, resulting in overoptimistic value estimates.

In the original Double Q-learning algorithm, two value functions are learned by assigning each experience randomly to update one of the two value functions, such that there are two sets of weights, θ and θ'.

$$Y_t^{\text{DoubleQ}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \operatorname*{argmax}_a Q(S_{t+1}, a; \boldsymbol{\theta}_t); \boldsymbol{\theta}'_t)$$

## Over-optimism due to estimation errors

- If the action values contain random errors uniformly distributed then each target is overestimated
- van Hasselt argued that noise in the environment can lead to overestimations even when using tabular representation, and proposed Double Q-learning as a solution.
- In addition, Thrun and Schwartz give a concrete example in which these overestimations even asymptotically lead to sub-optimal policies
- Finally estimation errors of any kind can induce an upward bias, regardless of whether these errors are due to environmental noise, function approximation, non-stationarity, or any other source.

## Double DQN

- The idea of Double Q-learning is to reduce overestimations by decomposing the max operation in the target into **action selection** and **action evaluation**
- Authors therefore proposed to evaluate the greedy policy according to the online network, but using the target network to estimate its value

estimating the value of the greedy policy, online weight

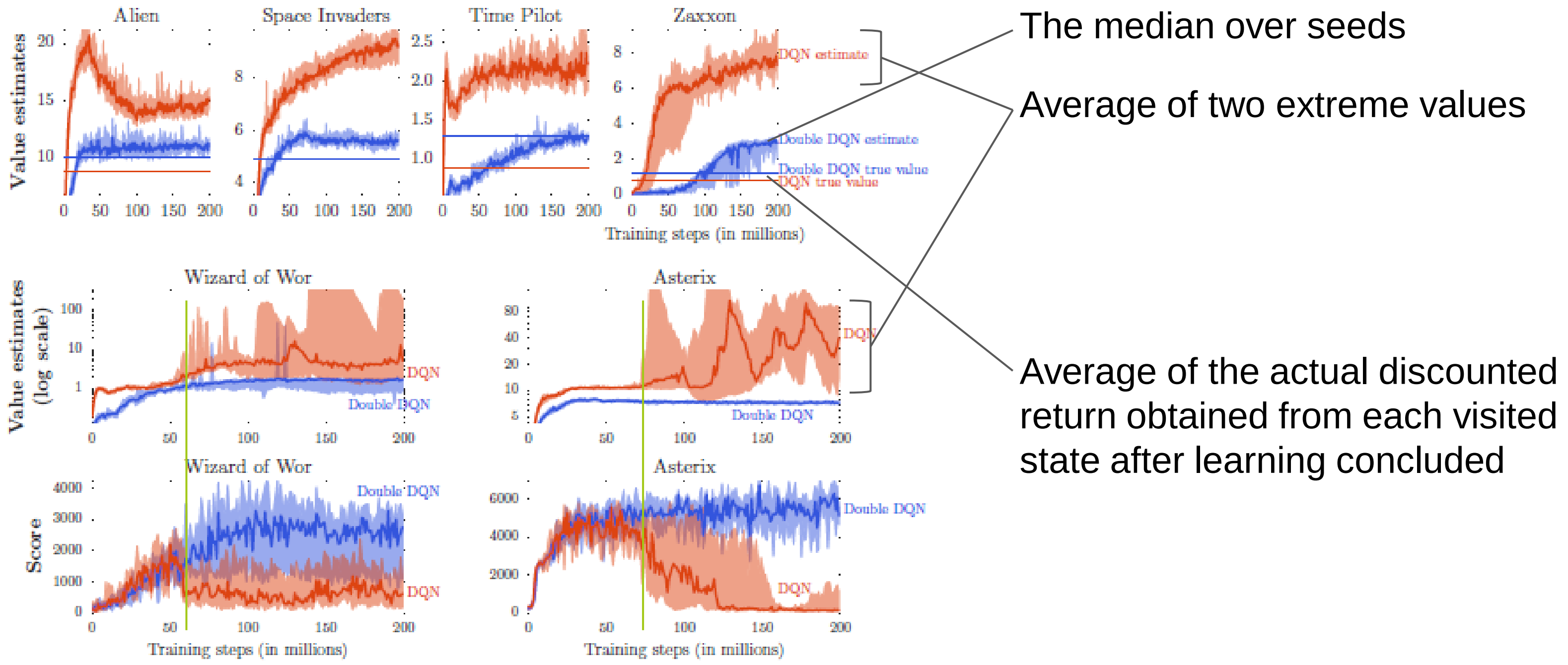
$$Y_t^{\text{DoubleDQN}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \operatorname*{argmax}_a Q(S_{t+1}, a; \theta_t), \theta_t^-)$$

$$Y_t^{\text{DoubleDQN}} \equiv R_{t+1} + \gamma Q(S_{t+1}, \operatorname*{argmax}_a Q(S_{t+1}, a; \theta_t), \theta_t^-)$$

evaluation of the current greedy policy

- Its update is the same as for DQN

# Result



- The **middle row** shows the value estimates (in log scale) for two games in which DQN's over-optimism is quite extreme.
- The **bottom row** shows the detrimental effect of this on the score achieved by the agent as it is evaluated during training: the scores drop when the overestimations begin.

## Discussion

This paper has five contributions

1. Why Q-learning can be overoptimistic in large-scale problems, even if these are deterministic.
2. by analyzing the value estimates on Atari games we have shown that these overestimations are more common and severe in practice than previously acknowledged.
3. Double Q-learning can be used at scale to successfully reduce this over-optimism.
4. Double DQN, that uses the existing architecture and deep neural network of the DQN algorithm without requiring additional networks or parameters.
5. Double DQN finds better policies, obtaining new state-of-the-art results on the Atari 2600 domain.