Seminar 09
DQN modifications

1.04.2024

Nikolay Karpachev

DQN Problems: Overestimation

We use “max” operator to compute the target

$$L(s,a) = (Q(s,a) - (r + \gamma \max_{a'} Q(s',a')))^2$$

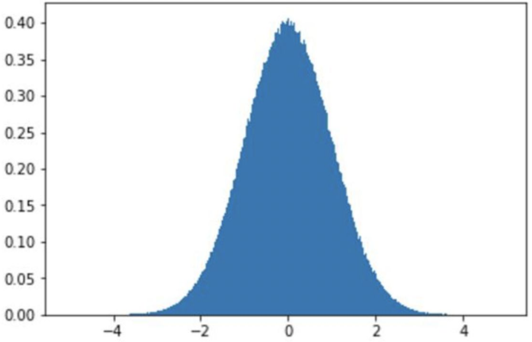
We have a problem

(although we want $\mathrm{E}_{s\sim S,a\sim A}[L(s,a)]$ to be equal zero)

DQN Problems: Overestimation

Normal distribution
$3*10^6$ samples

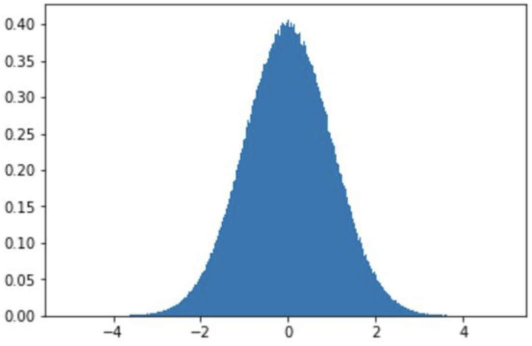
mean: ~0.0004



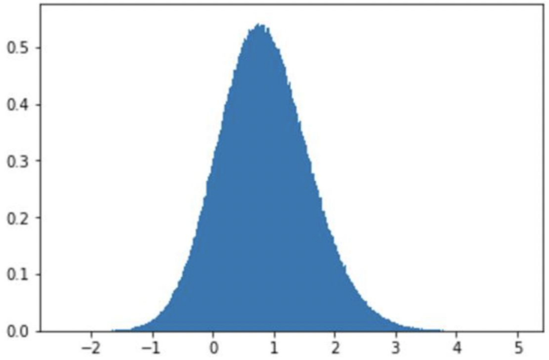
DQN Problems: Overestimation

Normal distribution
$3*10^6$ samples

mean: ~0.0004



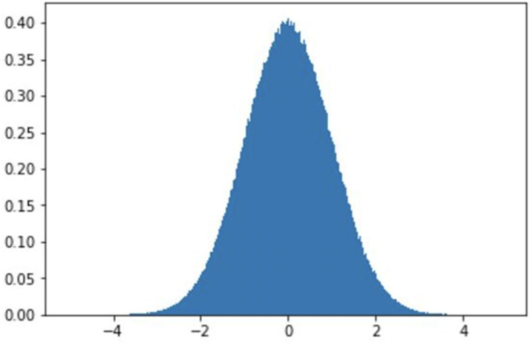
Normal distribution
$3*10^6$ x 3 samples
Then take maximum of every tuple
mean: ~0.8467



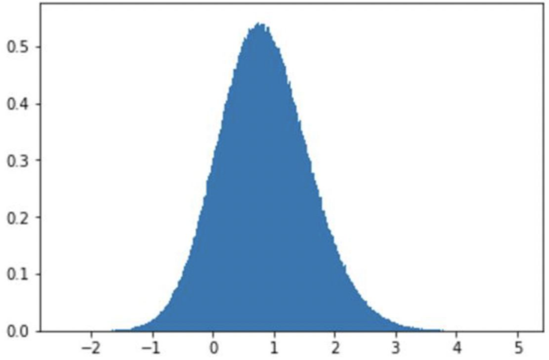
DQN Problems: Overestimation

Normal distribution
$3*10^6$ samples

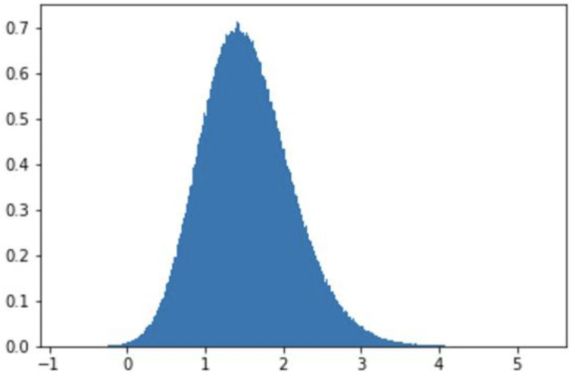
mean: ~0.0004



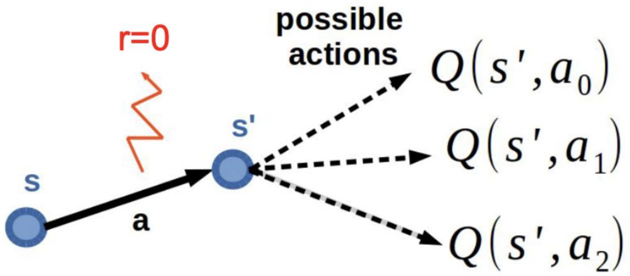
Normal distribution
$3*10^6$ x 3 samples
Then take maximum of every tuple
mean: ~0.8467



Normal distribution
$3*10^6$ x 10 samples
Then take maximum of every tuple
mean: ~1.538



DQN Problems: Overestimation

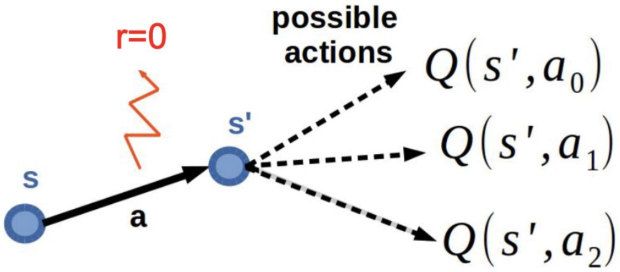


Suppose true $Q(s', a')$ are equal to **0** for all a'

But we have an approximation (or other) error $\sim N(0, \sigma^2)$

So $Q(s, a)$ should be equal to **0**

DQN Problems: Overestimation



But if we update $Q(s,a)$ towards $r + \gamma \max_{a'} Q(s', a')$

we will have overestimated $Q(s,a)$ > **0** because

$$\mathrm{E}[\max_{a'} Q(s', a')] >= \max_{a'} E[Q(s', a')]$$

Double Q-Learning

$y = r + \gamma \max_{a'} Q(s', a')$ - Q-learning target

$y = r + \gamma Q(s', \operatorname{argmax}_{a'} Q(s', a'))$ - Rewritten Q-learning target

Idea: use two estimators of q-values: Q^A, Q^B
They should compensate mistakes of each other because they will be independent
Let’s get argmax from another estimator!

$y = r + \gamma Q^A(s', \operatorname{argmax}_a Q^B(s', a'))$ - Double Q-learning target

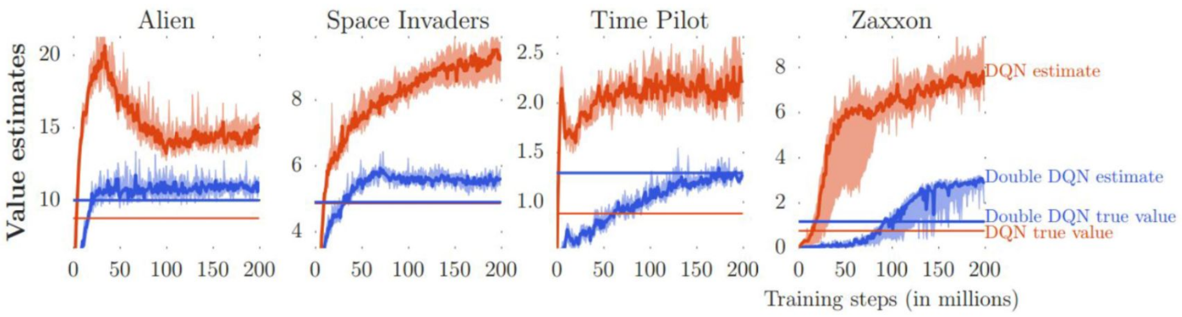
Double Deep Q-Learning (DDQN)

Deep RL with Double Q-learning

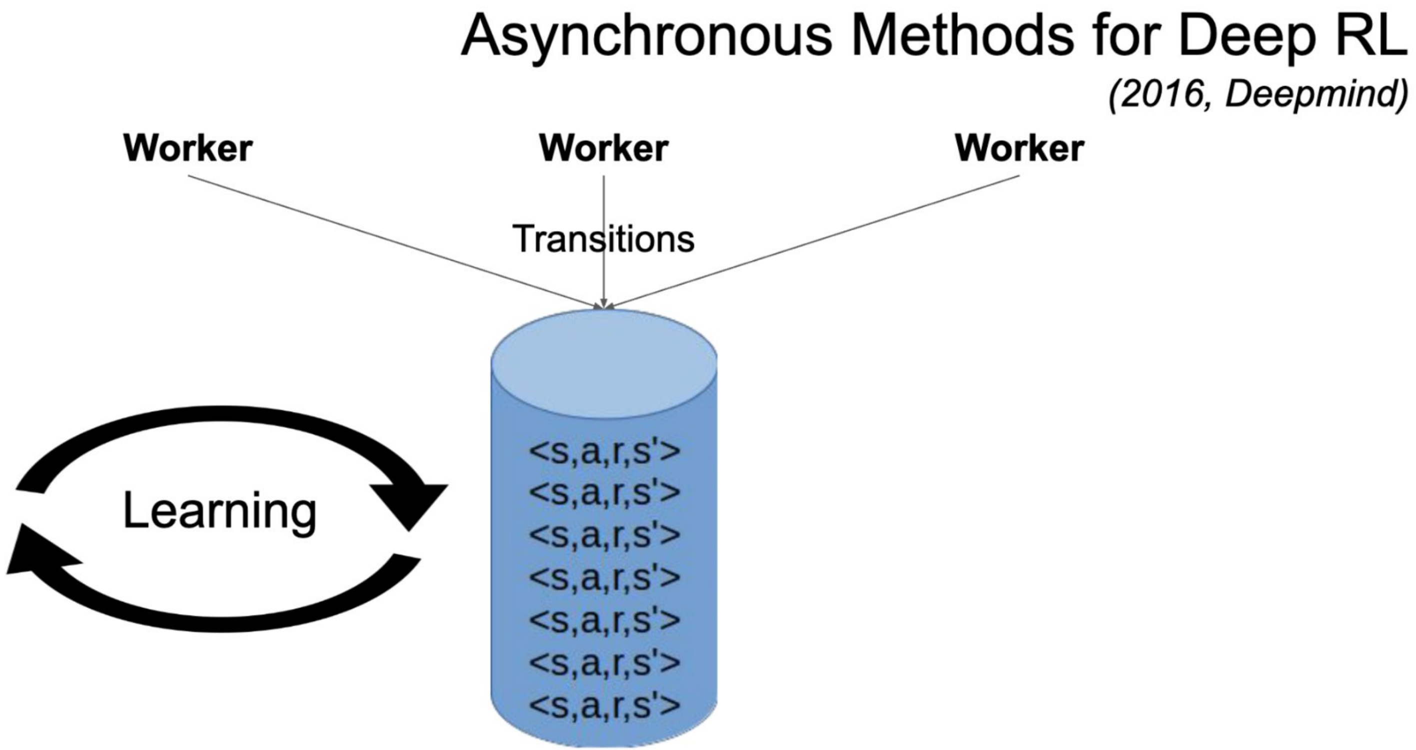
(Deepmind, 2015)

Idea: use main network to choose action!

$$y_{dqn} = r + \gamma \max_{a'} Q(s', a', \Theta^-)$$
$$y_{ddqn} = r + \gamma Q(s', \mathrm{argmax}_{a'} Q(s', a', \Theta), \Theta^-)$$



Async DQN


Asynchronous Methods for Deep RL
(2016, Deepmind)
Worker
Worker
Worker
Transitions
Learning
<s,a,r,s'>
<s,a,r,s'>
<s,a,r,s'>
<s,a,r,s'>
<s,a,r,s'>
<s,a,r,s'>
<s,a,r,s'>

Prioritized Experience Replay

(2016, Deepmind)

Idea: sample transitions from xp-replay cleverly

We want to set probability for every transition. Let's use the absolute value of TD-error of transition as a probability!

$$\text{TD-error } \delta = Q(s,a) - (r + \gamma Q(s', \text{argmax}_{a'} Q(s', a', \Theta), \Theta^-))$$

$$p = |\delta|$$

$$P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha}$$ where α is the priority parameter (when α is 0 it's the uniform case)

Do you see the problem?

Transitions become non i.i.d. and therefore we introduce the bias.

Prioritized Experience Replay

(2016, Deepmind)

Solution: we can correct the bias by using importance-sampling weights

$$w_i = \left(\frac{1}{N} \cdot \frac{1}{P(i)}\right)^{\beta}$$

where β is the parameter

So we sample using $P(i) = \frac{p_i^\alpha}{\sum_k p_k^\alpha}$ and multiply error by w_i