NMix Q-learning : Investigating overestimation bias of Q-values

2023.06.12.

20170203 Kyungwook Nam

20190289 Doojin Baek

20233143 Subin Kim

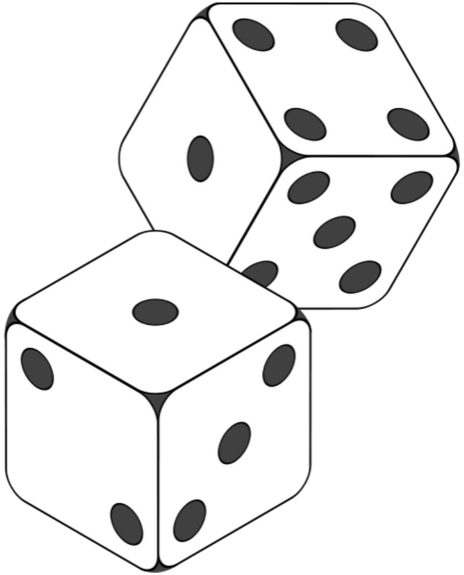
Overestimation bias of Q-value

Example: Throwing dice for N times

Expectation < Maximum value among trials

What if overestimations are not uniform…?

-> Leads to suboptimal policy



How to overcome overestimation?(1)

DQN[1]

$$Q(s,a) = r + \gamma max_{a'}[Q(s\prime, a\prime)]$$

DDQN[2]

$$Q(s,a) = r + \gamma \cdot Q(s\prime, argmax_{a'}[Q(s\prime, a\prime; \theta-)]; \theta)$$

[1] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).
[2] Anschel, et al. "Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning." *International conference on machine learning*. PMLR, 2017.

How to overcome overestimation?(2)

Averaged Q-Learning[3]

	Q^1	Q^2	Q^3	
a^1	$Q^1(s,a^1)$	$Q^2(s,a^1)$	$Q^3(s,a^1)$	$\frac{1}{3}\Sigma Q^k(s,a^1)$
a^2	$Q^1(s,a^2)$	$Q^2(s,a^2)$	$Q^3(s,a^2)$	$\frac{1}{3}\Sigma Q^k(s,a^2)$
a^3	$Q^1(s,a^3)$	$Q^2(s,a^3)$	$Q^3(s,a^3)$	$\frac{1}{3}\Sigma Q^k(s,a^3)$

$$\boxed{max_{a'}(\tfrac{1}{3}\Sigma Q^k(s,a'))}$$

MaxMin Q-Learning[4]

	Q^1	Q^2	Q^3	
a^1	$Q^1(s,a^1)$	$Q^2(s,a^1)$	$Q^3(s,a^1)$	$min_Q\, Q(s,a^1)$
a^2	$Q^1(s,a^2)$	$Q^2(s,a^2)$	$Q^3(s,a^2)$	$min_Q\, Q(s,a^2)$
a^3	$Q^1(s,a^3)$	$Q^2(s,a^3)$	$Q^3(s,a^3)$	$min_Q\, Q(s,a^3)$

$$\boxed{max_{a'}\, min_Q\, Q(s,a')}$$

[3] Van Hasselt, Hado, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. No. 1. 2016.
[4] Lan, Qingfeng, et al. "Maxmin q-learning: Controlling the estimation bias of q-learning." *arXiv preprint arXiv:2002.06487* (2020).

How to overcome overestimation?(3)

Averaged Q-Learning[3]

MaxMin Q-Learning[4]

Q1) For N-network Q-learning, how do Q-selection strategies handle the issue of Q-value overestimation?

⬇

Q2) How about the strategy of taking *min* operation along *max* Q-values of each network?

$max_{a'}(\frac{1}{3}\Sigma Q^k(s,a'))$

$max_{a'}\ min_Q\ Q(s,a')$

[3] Van Hasselt, Hado, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. No. 1. 2016.
[4] Lan, Qingfeng, et al. "Maxmin q-learning: Controlling the estimation bias of q-learning." *arXiv preprint arXiv:2002.06487* (2020).

NMix : N-network Min-max Q-learning

	Q^1	…	Q^j	…	Q^n	
a^1	$Q^1(s,a^1)$		$Q^j(s,a^1)$		$Q^n(s,a^1)$	$min_Q\ Q(s,a^1)$
$\vdots$						
a^i	$Q^1(s,a^i)$		$Q^j(s,a^i)$		$Q^n(s,a^i)$	$min_Q\ Q(s,a^i)$
$\vdots$						
a^m	$Q^1(s,a^m)$		$Q^j(s,a^m)$		$Q^n(s,a^m)$	$min_Q\ Q(s,a^m)$
	$max_a\ Q^1(s,a)$		$max_a\ Q^j(s,a)$		$max_a\ Q^n(s,a)$	

$max_a\ min_Q\ Q(s,a)$

MaxMin

NMix $\boxed{min_Q\ max_a\ Q(s,a)}$

NMix Q-target: $max_{a'}\ Q_{target}(s',a') = r + \gamma \cdot \boxed{min_Q\ max_{a'}\ Q(s',a')}$

NMix-MaxMin Comparison

Let's say the **MaxMin** output is $\underline{Q^j(s,a^i)}$, without losing generality.

Since **NMix** takes ***max*** over actions,

$$Q^j(s,a^i) \le C^j = max_a\, Q^j(s,a)$$

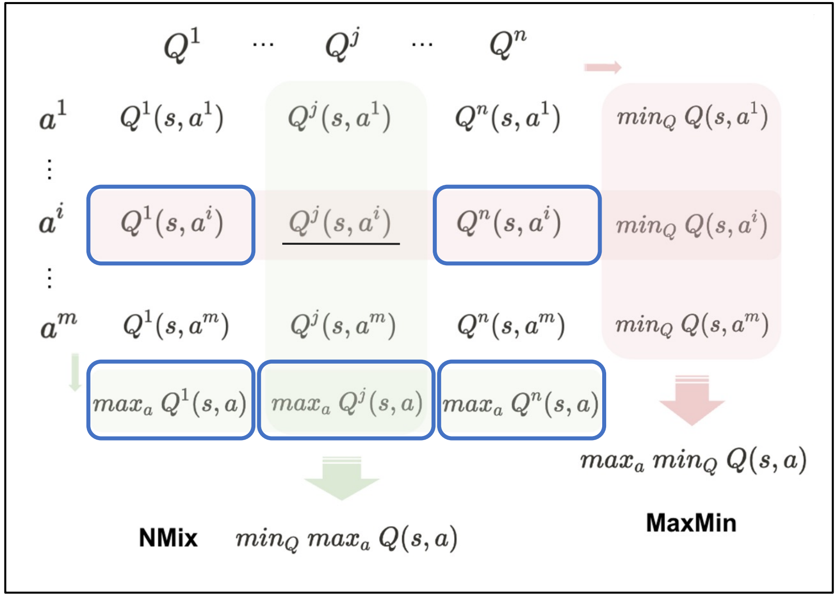
Because **MaxMin** takes ***min*** over Q's,

$$Q^j(s,a^i) \le Q^k(s,a^i) \le C^k = max_a\, Q^k(s,a)$$

In other words,

$$Q^j(s,a^i) \le c\ ,\ \forall c \in \{C^1, \cdots C^n\}$$

Note that the output of **NMix** is the following



$$min_Q max_a Q(s,a) = min(\{C^1, \cdots C^n\})$$ and hence $$min_Q max_a Q(s,a) \in \{C^1, \cdots C^n\}$$

Therefore, the NMix output is always greater than or equal to the MaxMin output ■

Experiment Design

We hypothesize the extent of overestimation bias in Q-learning based algorithms and support it empirically through experiments.

(underestimation) DDQN < MaxMin < NMix < DQN (overestimation)

Experiment Design

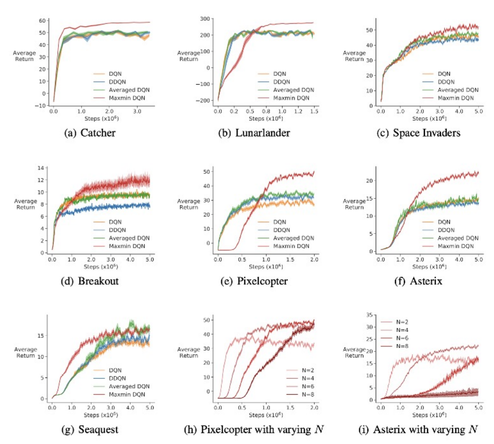
1. **Observe the impact of q-value overestimation bias across various environments.**

- **Experiment 1.** <u>Average return</u> of the algorithms over the three Atari games, Catcher, Copter, and Asterix.
- **Experiment 2.** Mean of <u>estimated q-values</u> per each step

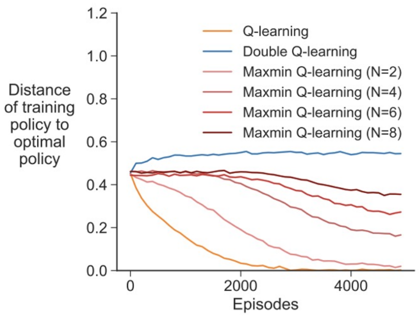
2. **Experiment the robustness of Q-learning based algorithms in stochastic MDP environment.**

- **Experiment 3.** Evaluate algorithms on simple MDP environment where <u>overestimation/underestimation is beneficial</u>

Referred from [4]



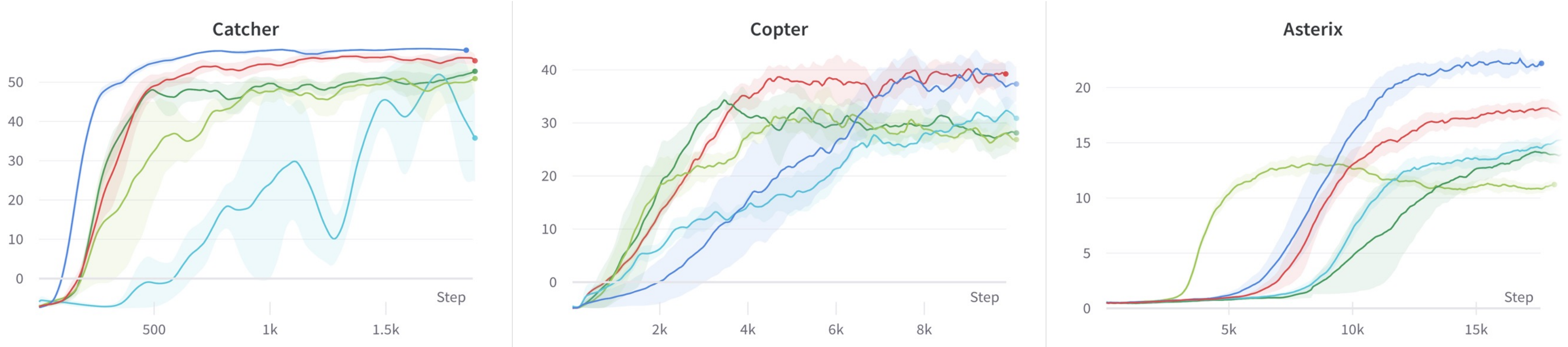
Referred from [4]



(a) $\mu = +0.1$ (overestimation helps)

[4] Lan, Qingfeng, et al. "Maxmin q-learning: Controlling the estimation bias of q-learning." *arXiv preprint arXiv:2002.06487* (2020).

Result 1. Average return



Each line is the average return of the two best models of each algorithm (selected among many hyperparameters)

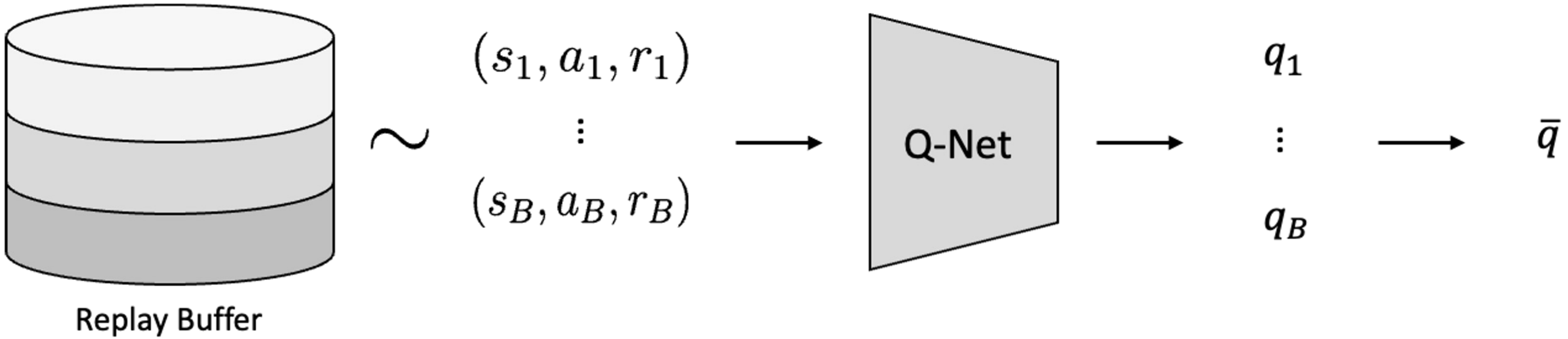
NMix Averaged Q-Learning DDQN Maxmin DQN

Performance

MaxMin ≥ **NMix** > DQN, DDQN, Averaged Q-Learning

Experiment 2. Q-value estimate

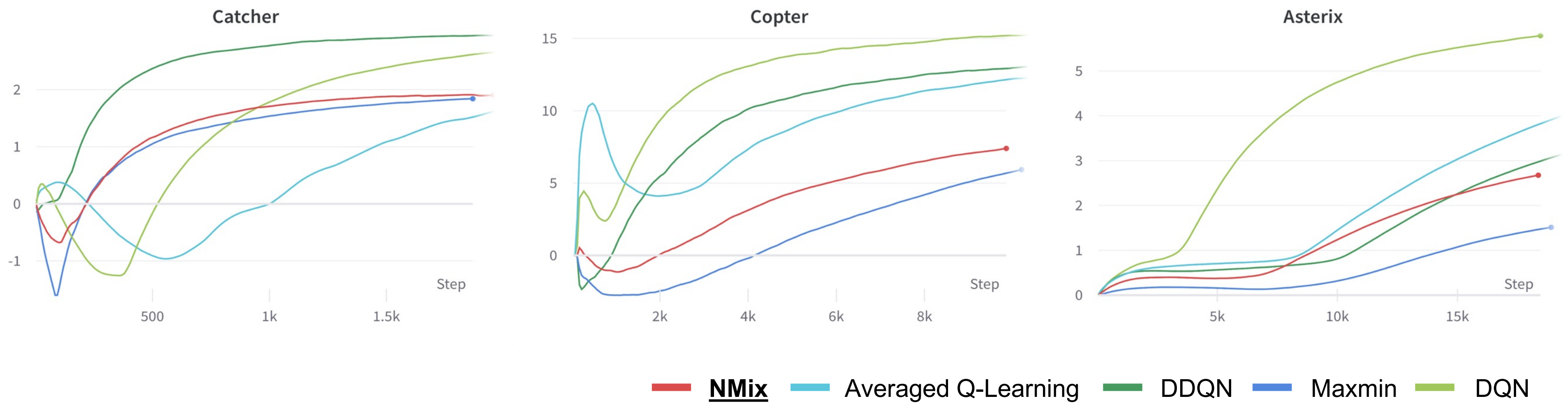
How to compute Q-Value Estimates



For every step

1. Sample *B* (state, action) pairs from the buffer
2. Compute the q values of each pair using Q-Net that will be updated at this step
3. Consider the average of the q values as the estimate

Result 2. Q-value estimate



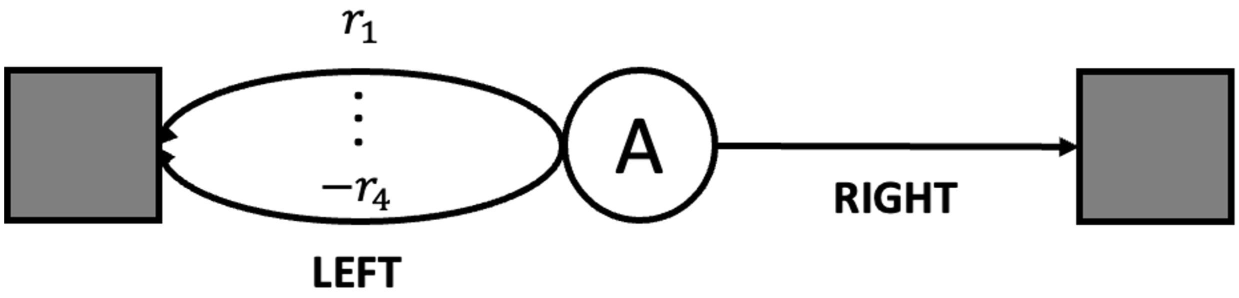
Estimated Q-value per step

MaxMin < **NMix** << DQN, DDQN, Averaged Q-Learning

Experiment 3. Stochastic MDP

Referred from [5]

$$r_i \sim \mu + U(-1,1)$$



Overestimation (or underestimation) may be helpful in some cases

1. μ = +0.3

Overestimating ‘Left’ is beneficial

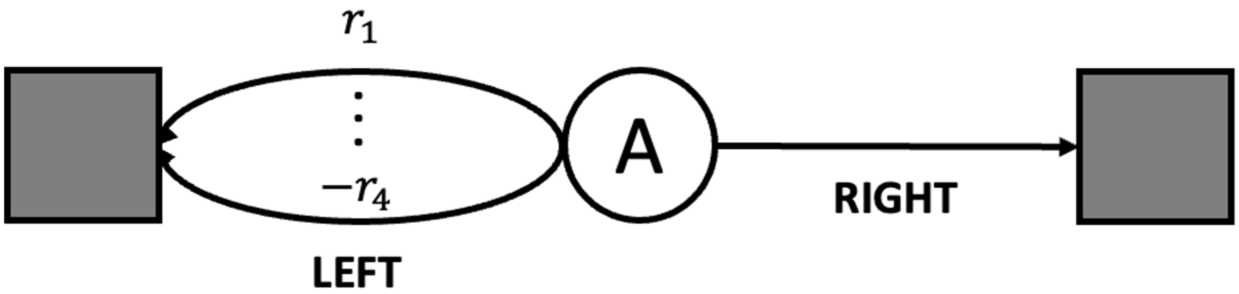
2. μ = -0.3

Underestimating ‘Left’ is beneficial

[5]Richard S Sutton and Andrew G Barto. Reinforcement Learning: An Introduction. MIT Press, second edition, 2018.

Experiment 3. Stochastic MDP

$$r_i \sim \mu + U(-1, 1)$$

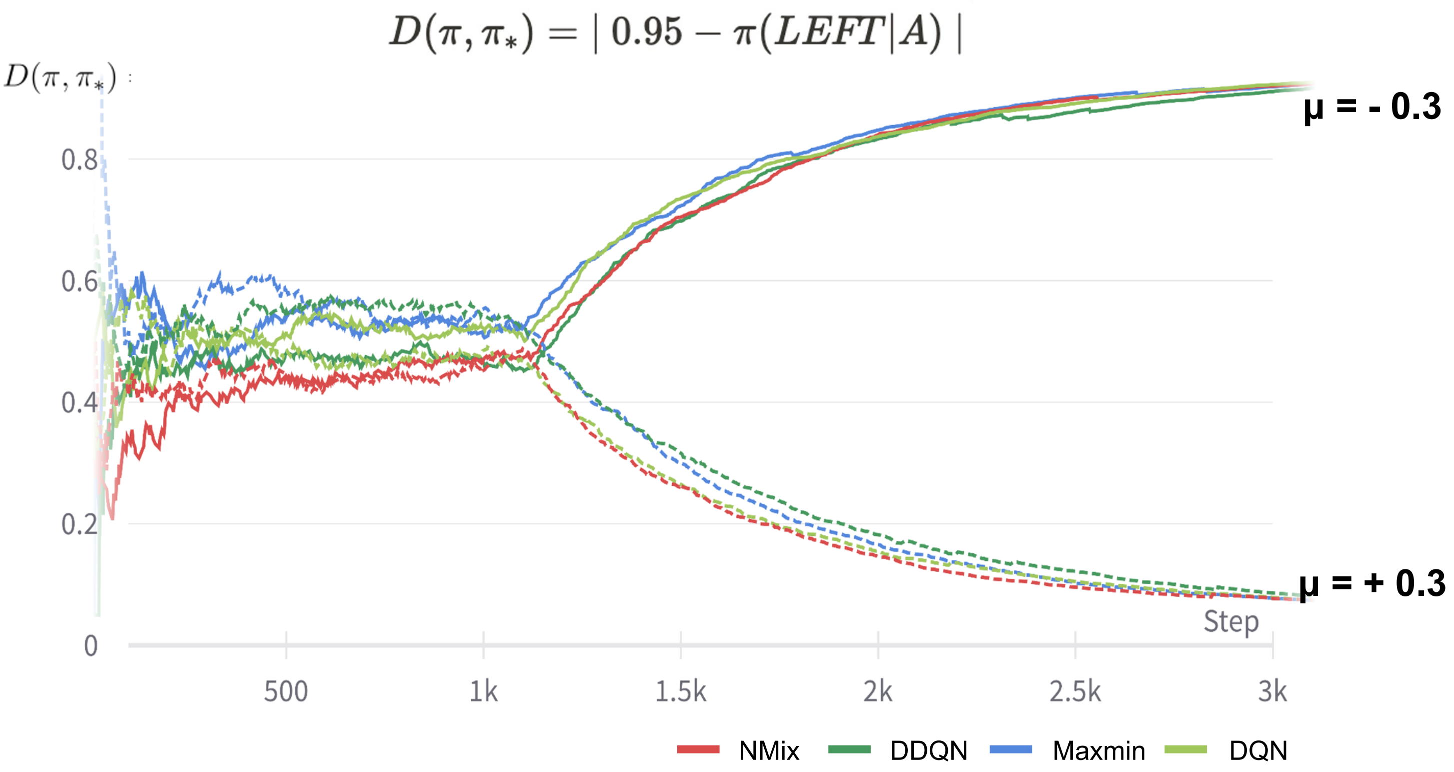


Measuring distance to the ϵ-greedy policy (ϵ = 0.1) :

$$D(\pi, \pi_*) = |\ \pi_*(LEFT|A) - \pi(LEFT|A)\ | = |\ 0.95 - \pi(LEFT|A)\ |$$

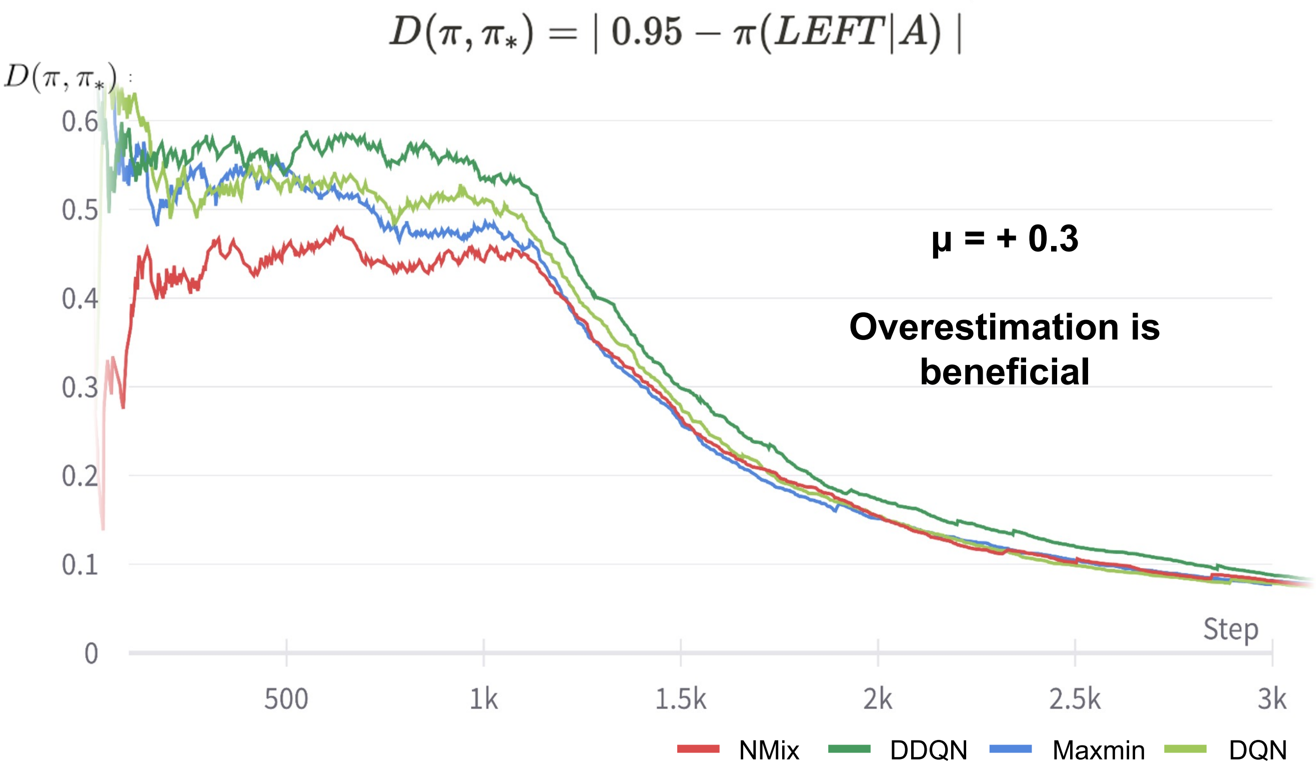
Optimal policy $\pi_*(LEFT|A)$ = 0.95 on the environment where 'Left' is good (μ > 0)
When ϵ = 0.1, (follow greedy policy) + (follow random policy)*0.5 = 0.9 + 0.1*0.5 = 0.95

Experiment 3. Stochastic MDP



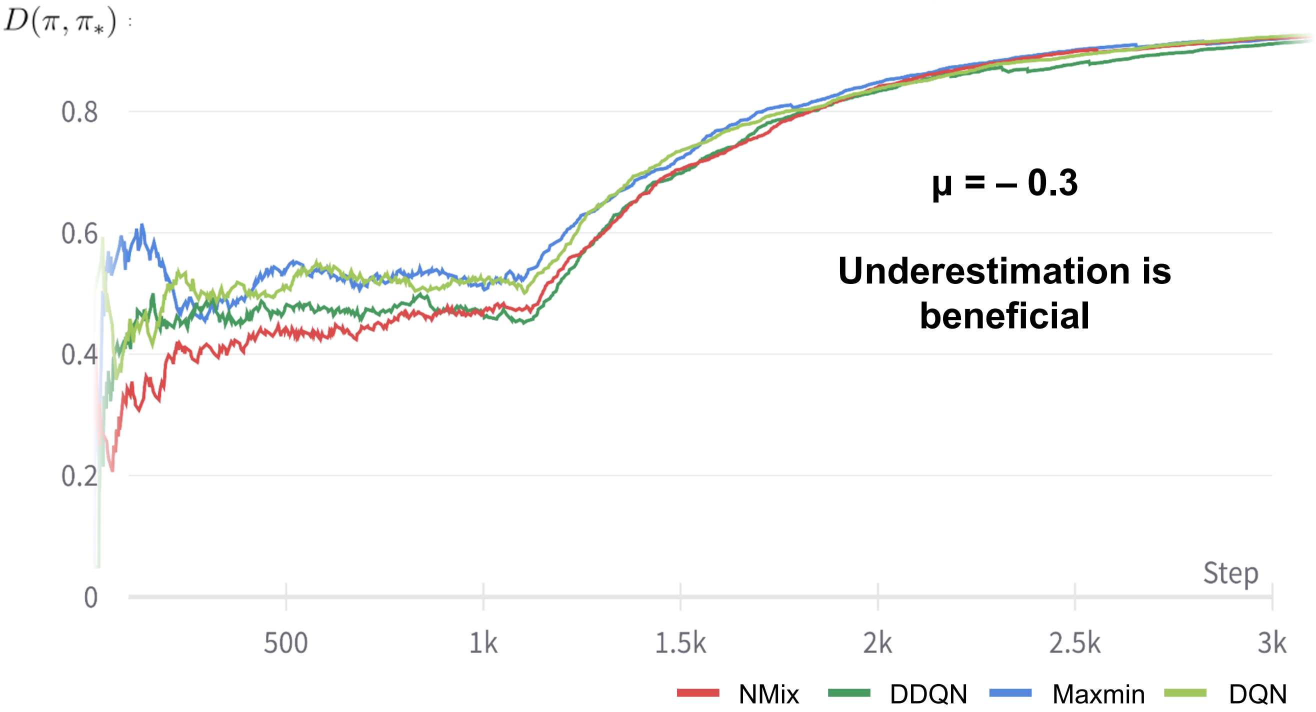
KAIST

Experiment 3. Stochastic MDP



Experiment 3. Stochastic MDP

$$D(\pi, \pi_*) = |\, 0.95 - \pi(LEFT|A)\, |$$



μ = – 0.3

Underestimation is beneficial

Contribution & Result

1. Devised **NMix Q-learning** algorithm to mitigate Q-value overestimation bias

$$Q(s,a) = r + \gamma \cdot min_Q \, max_a \, Q(s,a)$$

2. Hypothesized and observed the impact of q-value overestimation bias across various environments.
   - **(Hypothesis)** DDQN < MaxMin < **NMix** < DQN (overestimate)
   - **(Average return)** MaxMin ≥ **NMix** > DQN, DDQN, Averaged Q-Learning
   - **(Q-value estimate)** MaxMin < **NMix** << DQN, DDQN, Averaged Q-Learning
   - **(MDP-overestimation beneficial)** DQN > DDQN
   - **(MDP-underestimation beneficial)** DQN > DDQN

NMix Q-learning is effective to decrease the overestimation bias, and as hypothesized, magnitude of overestimation was larger than MaxMin Q-Learning.

Future Works

1. Improvement on simple MDP environment
    - Behavioral tendency of MaxMin and NMix Q-learning is not clearly distinguishable
    - Model size is comparatively big for simple MDP environment

2. Evaluate on complicated environments and analyze the effect of overestimation / underestimation

3. Prove convergence of NMix Q-learning.

4. Additional ablation study on hyperparameters:

    Number of target networks, replay buffer capacity, epsilon, etc..

Reference

[1] Mnih, Volodymyr, et al. "Playing atari with deep reinforcement learning." *arXiv preprint arXiv:1312.5602* (2013).

[2] Anschel, et al. "Averaged-dqn: Variance reduction and stabilization for deep reinforcement learning." *International conference on machine learning*. PMLR, 2017.

[3] Van Hasselt, Hado, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning." *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. No. 1. 2016.

[4] Lan, Qingfeng, et al. "Maxmin q-learning: Controlling the estimation bias of q-learning." *arXiv preprint arXiv:2002.06487* (2020).

[5]Richard S Sutton and Andrew G Barto. Reinforcement Learning: An Introduction. MIT Press, second edition, 2018.

Thank you!

$$
\begin{array}{cccccc}
 & Q^1 & \cdots & Q^j & \cdots & Q^n \\
a^1 & Q^1(s,a^1) & & Q^j(s,a^1) & & Q^n(s,a^1) & min_Q\ Q(s,a^1) \\
\vdots & & & & & & \\
a^i & Q^1(s,a^i) & & Q^j(s,a^i) & & Q^n(s,a^i) & min_Q\ Q(s,a^i) \\
\vdots & & & & & & \\
a^m & Q^1(s,a^m) & & Q^j(s,a^m) & & Q^n(s,a^m) & min_Q\ Q(s,a^m) \\
 & max_a\ Q^1(s,a) & & max_a\ Q^j(s,a) & & max_a\ Q^n(s,a) &
\end{array}
$$

$max_a\ min_Q\ Q(s,a)$

MaxMin

NMix $\boxed{min_Q\ max_a\ Q(s,a)}$

NMix Q-target:

$$max_{a'}\ Q_{target}(s',a') = r + \gamma \cdot \boxed{min_Q\ max_{a'}\ Q(s',a')}$$

KAIST