The Deep Quality-Value family of Deep Reinforcement Learning Algorithms

IJCNN 2020

Matthia Sabatelli[1], Gilles Louppe[1], Pierre Geurts[1]
Marco A. Wiering[2]

November 8, 2019

[1]Montefiore Institute, Department of Electrical Engineering and Computer Science, Université de Liège, Belgium

[2]Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence University of Groningen, The Netherlands

Presentation outline

1. Model-Free Deep Reinforcement Learning (DRL)
2. The Deep Quality-Value Family of DRL Algorithms
3. Analysis of the Algorithms

Model-Free Reinforcement Learning

Model-Free Reinforcement Learning

In model-free RL we care about learning **value functions** which give us information about the policy π our agent is following.

The state-value (V) function

$$V^{\pi}(s) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^{k} r_{t+k} \middle| s_t = s, \pi\right].$$

Model-Free Reinforcement Learning

In model-free RL we care about learning **value functions** which give us information about the policy π our agent is following.

The state-action value (Q) function

$$Q^{\pi}(s, a) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^{k} r_{t+k} \middle| s_t = s, a_t = a, \pi\right].$$

Model-Free Reinforcement Learning

In model-free RL we care about learning **value functions** which give us information about the policy π our agent is following.

Deriving π

$$\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q(s_{t+1}, a)$$

Q-Learning and its DQN extension

Q-Learning and its DQN extension

Q-Learning is probably one of the most popular model-free RL algorithms, which learns the Q function in an *off-policy* learning setting.

Q-Learning

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Q-Learning and its DQN extension

Q-Learning is probably one of the most popular model-free RL algorithms, which learns the Q function in an *off-policy* learning setting.

Q-Learning

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

IEEE WCCI 2020
IEEE World Congress on Computational Intelligence
Glasgow, Scotland, United Kingdom – July 19-24, 2020

Q-Learning and its DQN extension

Q-Learning is probably one of the most popular model-free RL algorithms, which learns the Q function in an *off-policy* learning setting.

Q-Learning

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Q-Learning and its DQN extension

Q-Learning is probably one of the most popular model-free RL algorithms, which learns the Q function in an *off-policy* learning setting.

Q-Learning

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Deep Q-Learning (DQN)

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^-) - Q(s_t, a_t; \theta) \right)^2 \right].$$

Q-Learning and its DQN extension

Q-Learning is probably one of the most popular model-free RL algorithms, which learns the Q function in an *off-policy* learning setting.

Q-Learning

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Deep Q-Learning (DQN)

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^-) - Q(s_t, a_t; \theta) \right)^2 \right].$$

Q-Learning and its DQN extension

Q-Learning is probably one of the most popular model-free RL algorithms, which learns the Q function in an *off-policy* learning setting.

Q-Learning

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Deep Q-Learning (DQN)

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^{-}) - Q(s_t, a_t; \theta) \right)^2 \right].$$

Q-Learning and its DQN extension

Q-Learning is probably one of the most popular model-free RL algorithms, which learns the Q function in an *off-policy* learning setting.

Q-Learning

$$Q(s_t, a_t) := Q(s_t, a_t) + \alpha \left[r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Deep Q-Learning (DQN)

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^-) - Q(s_t, a_t; \theta) \right)^2 \right].$$

IEEE WCCI 2020
IEEE World Congress on Computational Intelligence
Glasgow, Scotland, United Kingdom – July 19-24, 2020

The Deep Quality-Value Family of DRL Algorithms

QV(λ)-Learning

- Tabular RL algorithm which jointly learns the $V(s)$ function and the $Q(s, a)$ function in an on-policy setting [3].

[3]Wiering, Marco A. "QV(λ)-Learning: A new on-policy reinforcement learning algrithm." Proceedings of the 7th European Workshop on Reinforcement Learning. 2005.

QV(λ)-Learning

- Tabular RL algorithm which jointly learns the $V(s)$ function and the $Q(s, a)$ function in an on-policy setting [3].

TD-Learning for learning $V(s)$

$$V(s) := V(s) + \alpha\big[r_t + \gamma V(s_{t+1}) - V(s_t)\big]e_t(s).$$

[3] Wiering, Marco A. "QV(λ)-Learning: A new on-policy reinforcement learning algrithm." Proceedings of the 7th European Workshop on Reinforcement Learning. 2005.

QV(λ)-Learning

- Tabular RL algorithm which jointly learns the $V(s)$ function and the $Q(s, a)$ function in an on-policy setting [3].

TD-Learning for learning $V(s)$

$$V(s) := V(s) + \alpha\big[r_t + \gamma V(s_{t+1}) - V(s_t)\big]e_t(s).$$

$\approx$ Q-Learning for learning $Q(s, a)$

$$Q(s, a) := Q(s, a) + \alpha\big[r_t + \gamma V(s_{t+1}) - Q(s, a)\big].$$

[3] Wiering, Marco A. "QV(λ)-Learning: A new on-policy reinforcement learning algrithm." Proceedings of the 7th European Workshop on Reinforcement Learning. 2005.

DQV-Learning

DQV-Learning

Two neural networks for two different value functions

- $V(s;\Phi) \approx V(s)$

DQV-Learning

Two neural networks for two different value functions

- $V(s;\Phi) \approx V(s)$
- $Q(s,a;\theta) \approx Q(s,a)$

DQV-Learning

Two neural networks for two different value functions

- $V(s; \Phi) \approx V(s)$
- $Q(s, a; \theta) \approx Q(s, a)$

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma V(s_{t+1}; \Phi^-) - V(s_t; \Phi) \right)^2 \right]$$

DQV-Learning

Two neural networks for two different value functions

- $V(s;\Phi) \approx V(s)$
- $Q(s,a;\theta) \approx Q(s,a)$

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \left[\left(r_t + \gamma V(s_{t+1};\Phi^{-}) - V(s_t;\Phi) \right)^2 \right]$$

DQV-Learning

Two neural networks for two different value functions

- $V(s; \Phi) \approx V(s)$
- $Q(s, a; \theta) \approx Q(s, a)$

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \Big[\Big(r_t + \gamma V(s_{t+1}; \Phi^-) - V(s_t; \Phi) \Big)^2 \Big]$$

DQV-Learning

Two neural networks for two different value functions

- $V(s; \Phi) \approx V(s)$
- $Q(s, a; \theta) \approx Q(s, a)$

Approximating the state-action value function

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma V(s_{t+1}; \Phi^{-}) - Q(s_t, a_t; \theta) \right)^2 \right]$$

DQV-Learning

Two neural networks for two different value functions

- $V(s;\Phi) \approx V(s)$
- $Q(s,a;\theta) \approx Q(s,a)$

Approximating the state-action value function

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)}\left[\Big(r_t + \gamma V(s_{t+1};\Phi^{-}) - Q(s_t, a_t;\theta)\Big)^2\right]$$

DQV-Learning

Two neural networks for two different value functions

- $V(s;\Phi) \approx V(s)$
- $Q(s,a;\theta) \approx Q(s,a)$

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \left[\Big(r_t + \gamma V(s_{t+1};\Phi^-) - V(s_t;\Phi) \Big)^2 \right]$$

Approximating the state-action value function

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \left[\Big(r_t + \gamma V(s_{t+1};\Phi^-) - Q(s_t, a_t;\theta) \Big)^2 \right]$$

DQV vs DQN and DDQN

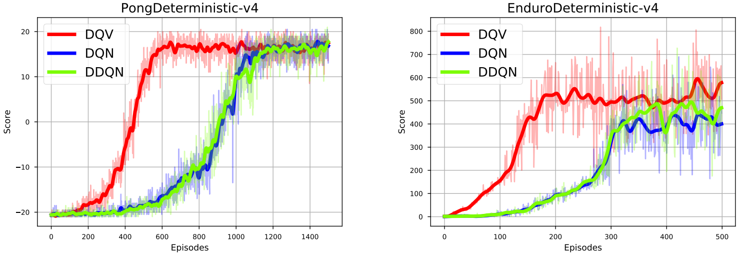


Figure: DQV learns significantly faster than DQN and DDQN [4]

[4] Sabatelli, Matthia, et al. "Deep Quality Value (DQV) Learning." Advances in Neural Information Processing Systems, Deep Reinforcement Learning Workshop. Montreal, 2018.

DQV-Max Learning

1. Can we successfully approximate two value functions while following an **off-policy** learning scheme?

DQV-Max Learning

1. Can we successfully approximate two value functions while following an **off-policy** learning scheme?
2. DQV-Max: a novel off-policy DRL algorithm

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \left[\left(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^{-}) - V(s_t; \Phi) \right)^2 \right].$$

DQV-Max Learning

1. Can we successfully approximate two value functions while following an **off-policy** learning scheme?
2. DQV-Max: a novel off-policy DRL algorithm

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \Big[\big(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^{-}) - V(s_t; \Phi) \big)^2 \Big].$$

DQV-Max Learning

1. Can we successfully approximate two value functions while following an **off-policy** learning scheme?
2. DQV-Max: a novel off-policy DRL algorithm

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \Bigg[\Big(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^{-}) - V(s_t; \Phi) \Big)^2 \Bigg].$$

DQV-Max Learning

1. Can we successfully approximate two value functions while following an **off-policy** learning scheme?
2. DQV-Max: a novel off-policy DRL algorithm

Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \Big[\Big(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^{-}) - V(s_t; \Phi) \Big)^2 \Big].$$

Approximating the state-action value function = DQV

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1}\rangle \sim U(D)} \Big[\Big(r_t + \gamma V(s_{t+1}; \Phi) - Q(s_t, a_t; \theta) \Big)^2 \Big]$$

DQV-Max Learning

1. Can we successfully approximate two value functions while following an **off-policy** learning scheme?
2. DQV-Max: a novel off-policy DRL algorithm

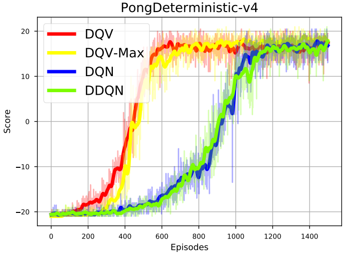
Approximating the state-value function

$$L(\Phi) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma \max_{a \in \mathcal{A}} Q(s_{t+1}, a; \theta^{-}) - V(s_t; \Phi) \right)^2 \right].$$

Approximating the state-action value function = DQV

$$L(\theta) = \mathbb{E}_{\langle s_t, a_t, r_t, s_{t+1} \rangle \sim U(D)} \left[\left(r_t + \gamma V(s_{t+1}; \Phi) - Q(s_t, a_t; \theta) \right)^2 \right]$$

DQV-Max Learning



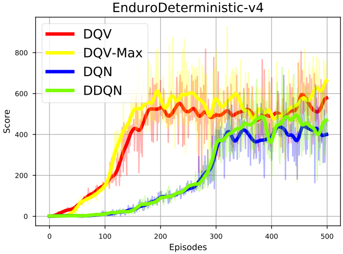


Figure: DQV-Max learns just as fast as DQV.

Empirical Results on the Atari Benchmark

TABLE I: The results obtained by DQV and DQV-Max on a subset of 15 `Atari` games. We can see that our newly introduced algorithms have a comparable, and often even better performance than DQN and DDQN. As highlighted by the green cells the overall best performing algorithm in our set of experiments is DQV-Max while the second-best performing algorithm is DQV (as reported by the yellow cells). Specific attention should be given to the games `BankHeist` and `Enduro` where DQV and DQV-Max are the only algorithms which can master the game with a final super-human performance.

Environment	Random	Human	DQN [6]	DDQN [7]	DQV	DQV-Max
Asteroids	719.10	13156.70	1629.33	930.60	1445.40	1846.08
Bank Heist	14.20	734.40	429.67	728.30	1236.50	1118.28
Boxing	0.10	4.30	71.83	81.70	78.66	80.15
Crazy Climber	10780.50	35410.50	114103.33	101874.00	108600.00	1000131.00
Enduro	0.00	309.60	301.77	319.50	829.33	875.64
Fishing Derby	-91.70	5.50	-0.80	20.30	1.12	20.42
Frostbite	65.20	4334.70	328.33	241.50	271.86	281.36
Gopher	257.60	2321.00	8520.00	8215.40	8230.30	7940.00
Ice Hockey	-11.20	0.90	-1.60	-2.40	-1.88	-1.12
James Bond	29.00	406.70	576.67	438.00	372.41	440.80
Montezuma's Revenge	0.00	4366.70	0.00	0.00	0.00	0.00
Ms.Pacman	307.30	15693.40	2311.00	3210.00	3590.00	3390.00
Pong	-20.70	9.30	18.90	21.00	21.00	21.00
Road Runner	11.50	7845.00	18256.67	48377.00	39290.00	20700.00
Zaxxon	32.50	9173.30	4976.67	10182.00	10950.00	8487.00

On the quality of the learned value-functions

DRL algorithms are prone to damage the quality of the learned value functions when the following elements are present:

1. A **function-approximator** is used when learning a value function

On the quality of the learned value-functions

DRL algorithms are prone to damage the quality of the learned value functions when the following elements are present:

1. A **function-approximator** is used when learning a value function
2. The algorithms rely on **bootstrapping** while the value function is regressed

On the quality of the learned value-functions

DRL algorithms are prone to damage the quality of the learned value functions when the following elements are present:

1. A **function-approximator** is used when learning a value function
2. The algorithms rely on **bootstrapping** while the value function is regressed
3. The algorithms learn **off-policy**

These elements when combined enhance the **overestimation bias** of the Q function which characterizes the DQN algorithm.

On the quality of the learned value-functions

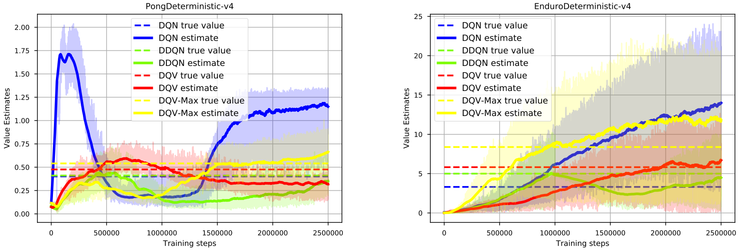


Figure: DQV and DQV-Max suffer less from the overestimation bias of the Q function.

On the neural capacity of the algorithms

1. Do we need two separately parametrized neural networks for successfully approximating $V(s)$ and $Q(s, a)$?

On the neural capacity of the algorithms

1. Do we need two separately parametrized neural networks for successfully approximating $V(s)$ and $Q(s,a)$?
2. We propose different extensions of the original DQV-Learning algorithm

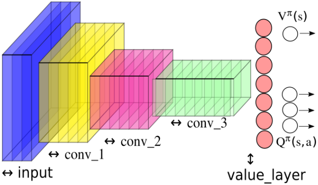


Figure: HARD-DQV Learning

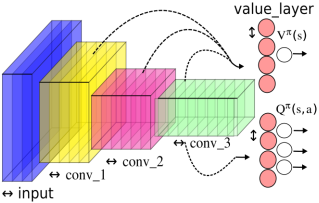


Figure: Dueling-DQV Learning

DQV-Learning Extensions

Two separate neural network with enough capacity are needed to exploit DQV's performance.

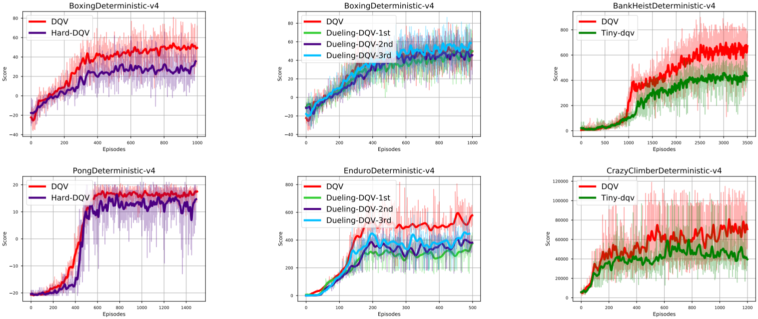


Figure: Results obtained by alternative versions of the DQV-Learning algorithm.

A final thank you note

LIÈGE université

Belgian Science Policy Office
belspo

university of groningen

https://github.com/paintception/Deep-Quality-Value-Family