

# Extensions to Q-Learning

---

Reinforcement Learning  
School of Data Science  
University of Virginia

Last updated: October 3, 2023

# Double Q-Learning

# Main Idea

DQN requires target estimates of this form:

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

The max() operation is used to estimate value

There may be noise in the system

Tends to produce a bias: overestimating value of Q

*Paper: Deep Reinforcement Learning with Double Q-learning*

*Hado van Hasselt, Arthur Guez, David Silver. Google DeepMind*

# Main Idea

## 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. Although not fully decoupled, the target network in the DQN architecture provides a natural candidate for the second value function, without having to introduce additional networks. We therefore propose to evaluate the greedy policy according to the online network, but using the target network to estimate its value. In reference to both Double Q-learning and DQN, we refer to the resulting algorithm as Double DQN. Its update is the same as for DQN, but replacing the target  $Y_t^{\text{DQN}}$  with

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

Decompose max() operation into action selection, action evaluation

Notice there are two Qs

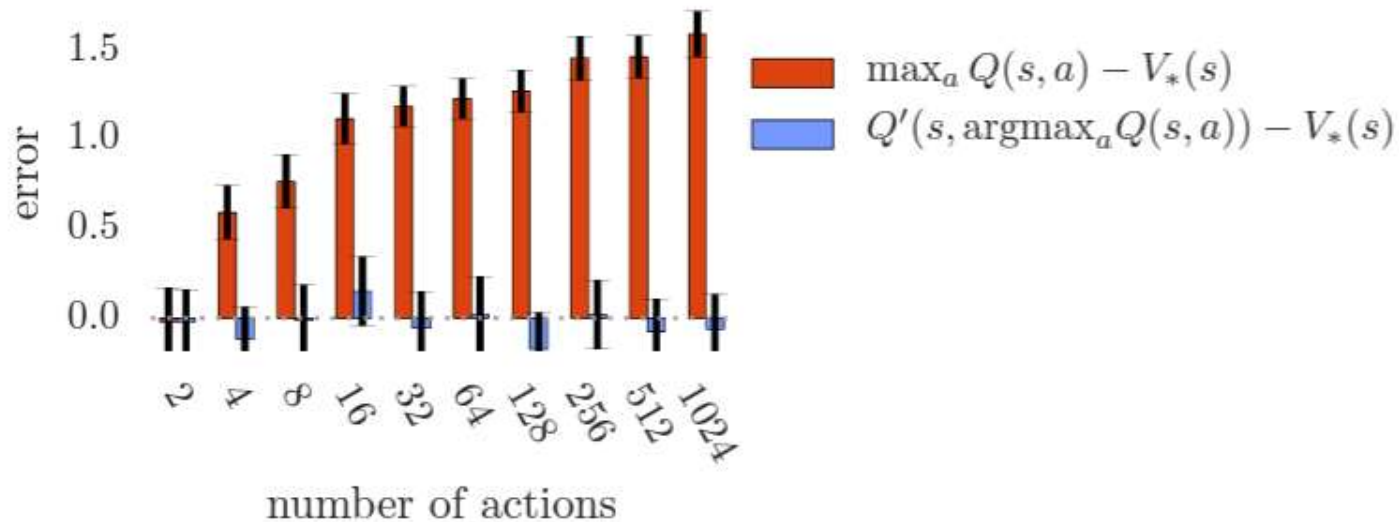
- One determines greedy policy using online network  $\theta_t$
- Another fairly evaluates the policy using the target network  $\theta_t^-$

# Bias Estimates

Consider state  $s$  where all true optimal action values are equal  $Q_*(s, a) = V_*(s)$ .

As number of actions increases, bias will increase for DQN

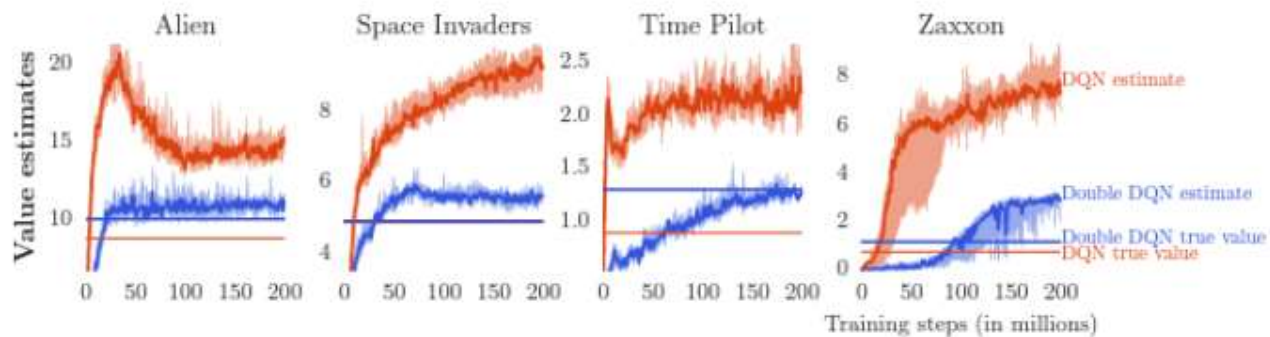
However, for Double Q-Learning, bias remains small



# Bias Estimates - Atari

Horizontal lines are computed by running agents after learning concluded and averaging discounted returns

Corresponds to values if there were no bias



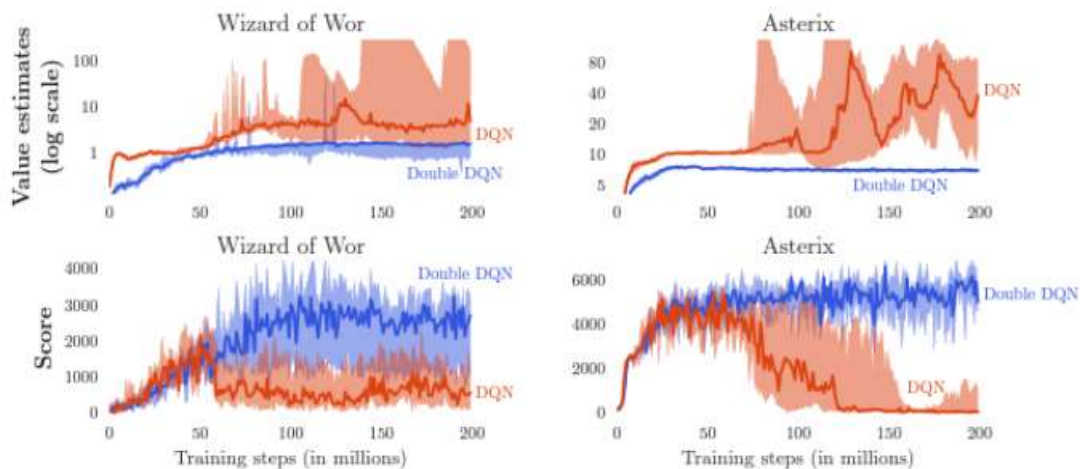
# Bias Estimates – Atari, Extreme Cases

In these cases, overoptimism in DQN is extreme (red curve)

Where red separates from blue, bias increases

This has detrimental effect on score (Double Q-Learning outperforms)

Double Q-Learning is also more stable



# Dueling Deep Q-Network



# Dueling Networks

Deep RL before this paper used conventional architectures (CNN, LSTM)

Focus here is new architecture better suited to model-free RL

*Dueling architecture* separates state values and action advantages

*Paper: Dueling Network Architectures for Deep Reinforcement Learning*  
*Wang et. al.*

# Dueling Architecture

Separate streams for value and advantage functions

Common convolution feature

Top figure is Q-network

Bottom figure is dueling Q-network

Value and advantage streams are combined

For each network, outputs are  $Q(s,a)$

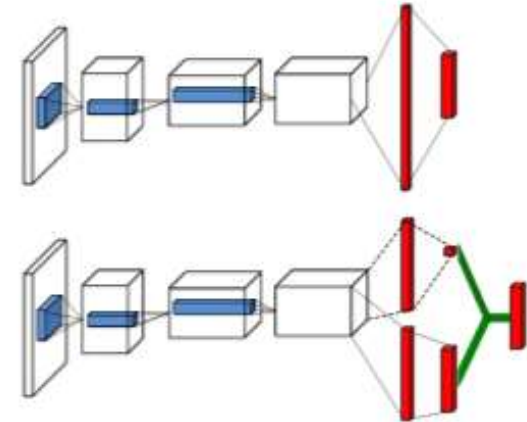


Figure 1. A popular single stream Q-network (**top**) and the dueling Q-network (**bottom**). The dueling network has two streams to separately estimate (scalar) state-value and the advantages for each action; the green output module implements equation (9) to combine them. Both networks output  $Q$ -values for each action.

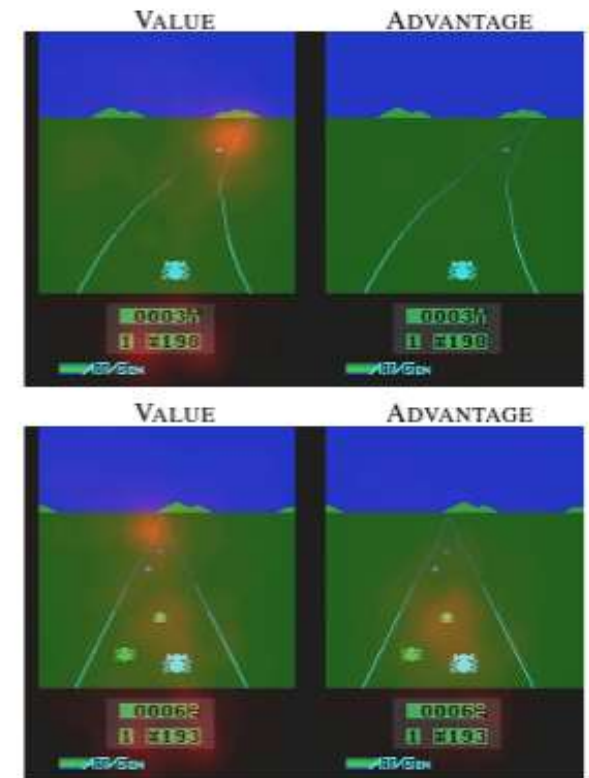
# Value and Advantage Function Saliency

Figure shows saliency map for two time steps

Value stream pays attention to horizon and score

Advantage stream: when no cars are on road, action doesn't matter (no attention paid)

When cars are on road, advantage stream pays attention to car in front



# Value and Advantage Function Definitions

Value functions:

$$Q^{\pi}(s, a) = \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$$

$$V^{\pi}(s) = \mathbb{E}_{a \sim \pi(s)} [Q^{\pi}(s, a)].$$

The advantage function isolates effect of action taken

$$A^{\pi}(s, a) = Q^{\pi}(s, a) - V^{\pi}(s)$$

$V(s)$  measures how good it is to be in state  $s$

$A(s,a)$  measures relative importance of each action

Sometimes, the action doesn't matter

# Identifiability

Given  $Q = V + A$ , we cannot recover  $V$  and  $A$  uniquely  
(adding any  $c$  to  $V$ , and subtracting  $c$  from  $A$ , leaves  $Q$  unchanged)

Add a constraint: subtract the average advantage from  $A$ :

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + \left( A(s, a; \theta, \alpha) - \frac{1}{|\mathcal{A}|} \sum_{a'} A(s, a'; \theta, \alpha) \right)$$

For actions with above average advantage, the second term in (\*) will be positive  
For action with greatest advantage, (\*) term will be largest across all actions

# Review of Dueling Architecture Code

We can see an implementation of Dueling Q-Network here:

Paper: *Reinforcement Learning for optimal sepsis treatment policies* (2017)

Authors: Raghu, Komorowski, Ahmed, Celi, Szolovits, Ghassemi

GitHub repo:

[https://github.com/aniruddhraghu/sepsisrl/blob/master/continuous/q\\_network.ipynb](https://github.com/aniruddhraghu/sepsisrl/blob/master/continuous/q_network.ipynb)

See section:

# advantage and value streams

# Implementation

Our two streams (V, A) for Q are part of the model architecture

Training step runs the same as a standard Q-network: backpropagation

# Evaluation

Sets up simple environment called *corridor*.

Redness of state signifies reward

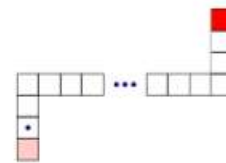
Can move up, down, left, right, or no move.

$\epsilon$ -greedy policy, measure performance by squared error (SE) vs. true action values

Single and Duel networks use MLP with three layers

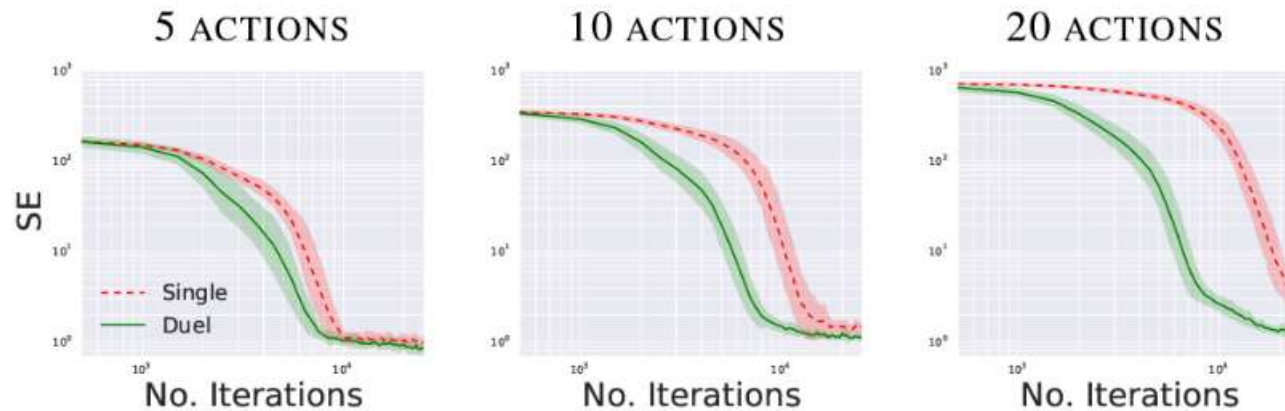
For duel network, after first hidden layer, network branches off to two streams

CORRIDOR ENVIRONMENT





# Evaluation, contd.



Dueling network converges faster than Single

More pronounced as  $|A|$  increases

As  $|A|$  increases, SE is lower for Dueling network. It performs better.