CSC2621 Topics in Robotics
Reinforcement Learning in Robotics

Week 4: Q-Value based RL

Animesh Garg

Deep Reinforcement Learning with Double Q-learning

Hado van Hasselt, Arthur Guez, David Silver

Dueling Network Architectures for Deep Reinforcement Learning

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, Nando de Freitas

Topic: Q-Value based RL

Presenter: Haoping Xu

Motivation: Overoptimism

- Q-learning methods are known to be overestimating the Q value
  - DQN and other Q learning methods have this common issue and their performances are lowered due to that.
  - But how bad is this error? And how it affect the model performance.
  - Double Q learning is known to be a solution for this overestimation problem, how to combine it with DQN?

Contributions

- Double DQN
  - Combining DQN and Double Q-learning to solve overoptimism problems for Q values
  - Provide a solid theoretical analysis of overestimation error bound in tradition Q learning
  - Demonstrate large estimation error in DQN and how DDQN fixs it and improve the performance using Atari games

General Background (Q learning)

Discount Return $R_t = \sum_{T=t}^{\infty} \gamma^{T-t} r_t$

State action value function $Q_\pi(s,a) = \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$

State value function $V_\pi(s) = \mathbb{E}_{a \sim \pi(s)}[Q_\pi(s,a)]$

Advantage function $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$

General Background (DQN)

Square Error Loss $$L_t(\theta_t) = \mathbb{E}[(y_t - Q(s_t, a_t; \theta_t))^2]$$

Update gradient $$\nabla_{\theta_t} L_t(\theta_t) = \mathbb{E}[(y_t - Q(s_t, a_t; \theta_t))\nabla_{\theta_t} Q(s_t, a_t; \theta_t)]$$

Q-learning Target $$y_t^Q = r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta_t)$$

DQN Target value $$y_t^{DQN} = r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta_t^-)$$

θ_t^- is a separate and fixed target network. In DQN, it is fixed and copied from the online network θ_t every k steps

General Background (Double Q learning)

DQN Target value $$y_t^{DQN} = r_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a'; \theta_t^-)$$

Rewrite it to Double Q form $$y_t^{DQN} = r_{t+1} + \gamma Q(s_{t+1}, \arg\max_{a'} Q(s_{t+1}, a'; \theta_t^-); \theta_t^-)$$

Double Q learning Target $$y_t^{Double} = r_{t+1} + \gamma Q(s_{t+1}, \arg\max_{a'} Q(s_{t+1}, a'; \theta_t); \theta_t')$$

In double Q learning, two set of weights are maintained, one to determine the action selected by greedy policy and another to determine its Q value.
However, for DQN, only the offline set of weight is used to both choose the action and determine the target value. This can leads to overoptimism problem.

Problem: Overoptimism

- Q-learning methods are known to be overestimating the Q value
  - Even if Q function is unbiased and avg square error is constant C, with m actions, the lower bound for errors is $\sqrt{\frac{C}{m-1}}$

Let $\epsilon_a = Q_t(s,a) - V^*(s)$ Assume exists a setting of all errors such that $\max_a \epsilon_a < \sqrt{\frac{C}{m-1}}$

Let $\{\epsilon_i^+\}$ a set of positive error of size n. and $\{\epsilon_j^-\}$be set of negative errors of size m - n

If $n = m$ then $\sum_a \epsilon_a = 0 \Rightarrow \forall \epsilon_a = 0$ which controdicts $\sum_a \epsilon_a^2 = mC$ then $n < m$

$\sum_{i=1}^{n} \epsilon_i^+ < n \max_i \epsilon_i^+ < n\sqrt{\frac{C}{m-1}}$ also have $\sum_{j=1}^{m-n} \|\epsilon_j^-\| < n\sqrt{\frac{C}{m-1}} \Rightarrow \max_j \|\epsilon_j^-\| < n\sqrt{\frac{C}{m-1}}$

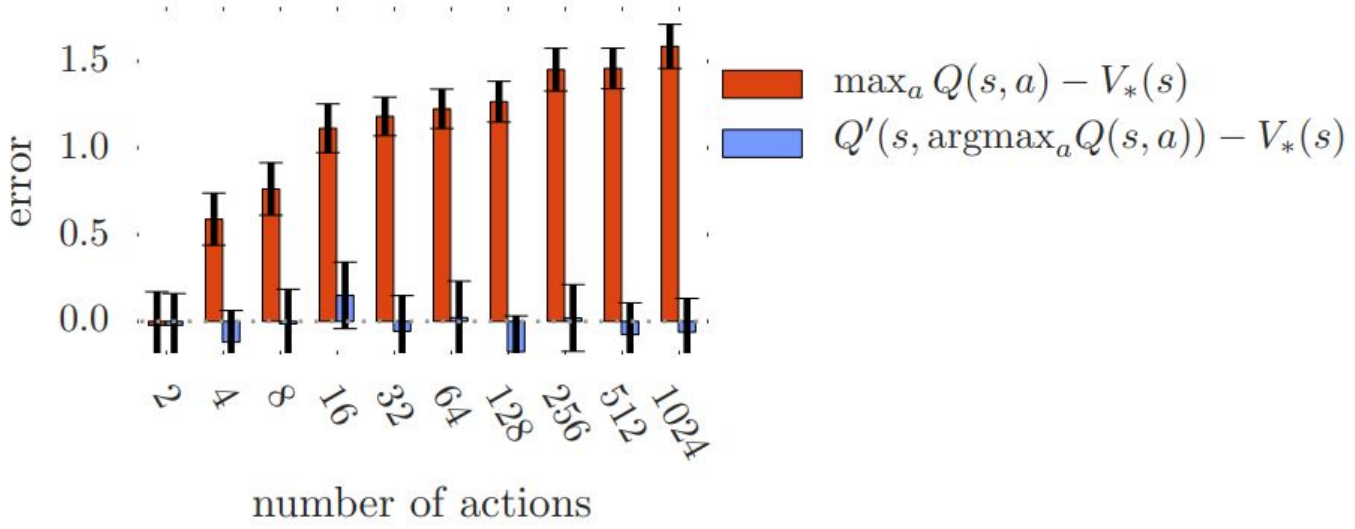
By Holder's inequality $\sum_{j=1}^{m-n} (\epsilon_j^-)^2 \leq \sum_{j=1}^{m-n} \|\epsilon_j^-\| \max_j \|\epsilon_j^-\| < \frac{n^2 C}{m-1}$

$\sum_{a=1}^{m} (\epsilon_a)^2 = \sum_{j=1}^{m-n} (\epsilon_j^-)^2 + \sum_{i=1}^{n} (\epsilon_i^+)^2 < n\frac{C}{m-1} + n^2\frac{C}{m-1} = \frac{n(n+1)}{m-1} C \leq mC$

Contradict with the assumption that $\sum_{a=1}^{m} \epsilon_a^2 < mC$ therefore $\max_a \epsilon_a \geq \sqrt{\frac{C}{m-1}}$

Problem: Overoptimism

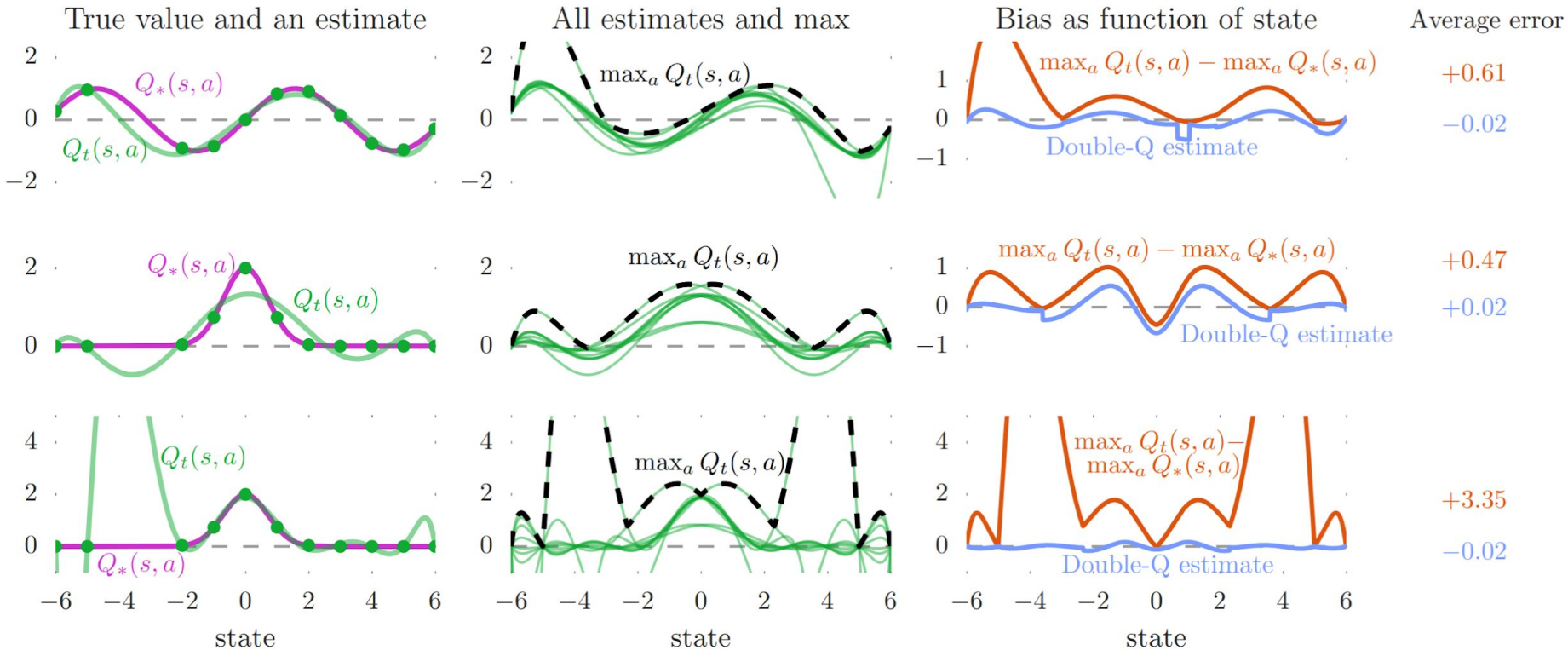
- Q-learning methods are known to be overestimating the Q value
  - In real cases, the estimation error grows as number of actions increases
  - Double Q learning has a 0 error lower bound and performs better than Q learning in real cases



Problem: Overoptimism

- Q-learning methods are known to be overestimating the Q value
  - Even if the true Q values are given, estimating it by sampling points introduces error, which will be amplified by bootstrap multiple estimations and pick the largest

- Q* is the true value
- Q* is sampled in green points
- Q_t is a polynomial estimate of Q* in different degrees
- Bootstrap several Q_t to get their max line



Algorithm Double DQN

Note DQN and Double Q learning both maintains two set of weights, but their usages are different:

- For both of them online network is updated at each step by square error of Q value and target value
- In DQN, another set of weight, target network is used to select and evaluate action
- In Double Q learning, both networks are used in target value function, one for picking best action, one for getting Q value

Combine these two together, we get Double DQN(DDQN):

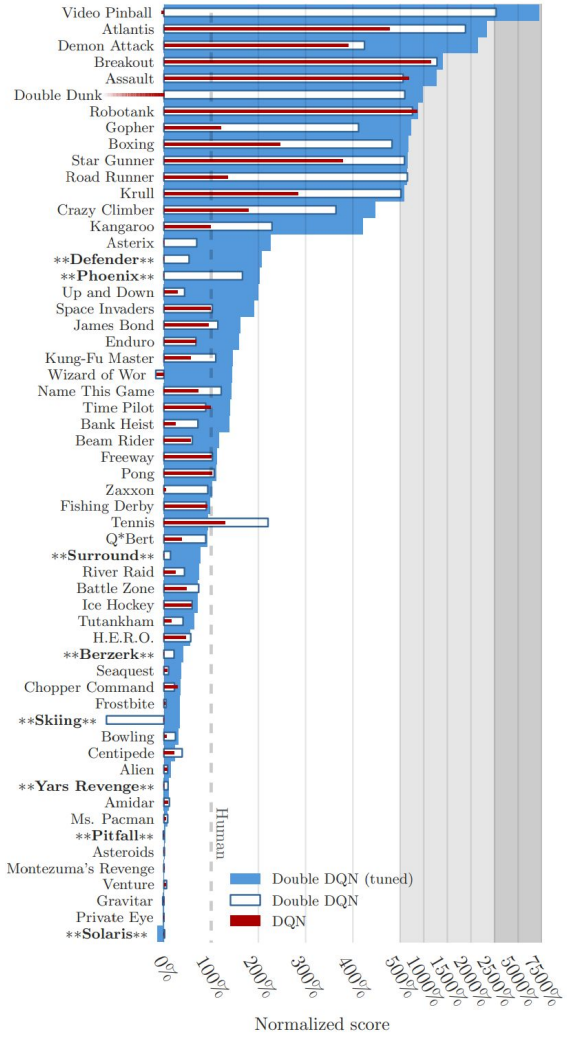
- Keep online and target networks in DQN, but use Double Q learning style target function by using both networks.
- Minimal possible change to DQN, still compatible with all DQN tricks, i.e. experience replay, target network
- Not additional process or weights are required, reusing the online network
- Less likely to overestimate Q value, thanks to Double Q like target function

$$y_t^{DDQN} = r_{t+1} + \gamma Q(s_{t+1}, \arg\max_{a'} Q(s_{t+1}, a'; \theta_t); \theta_t^-)$$

Double DQN Results

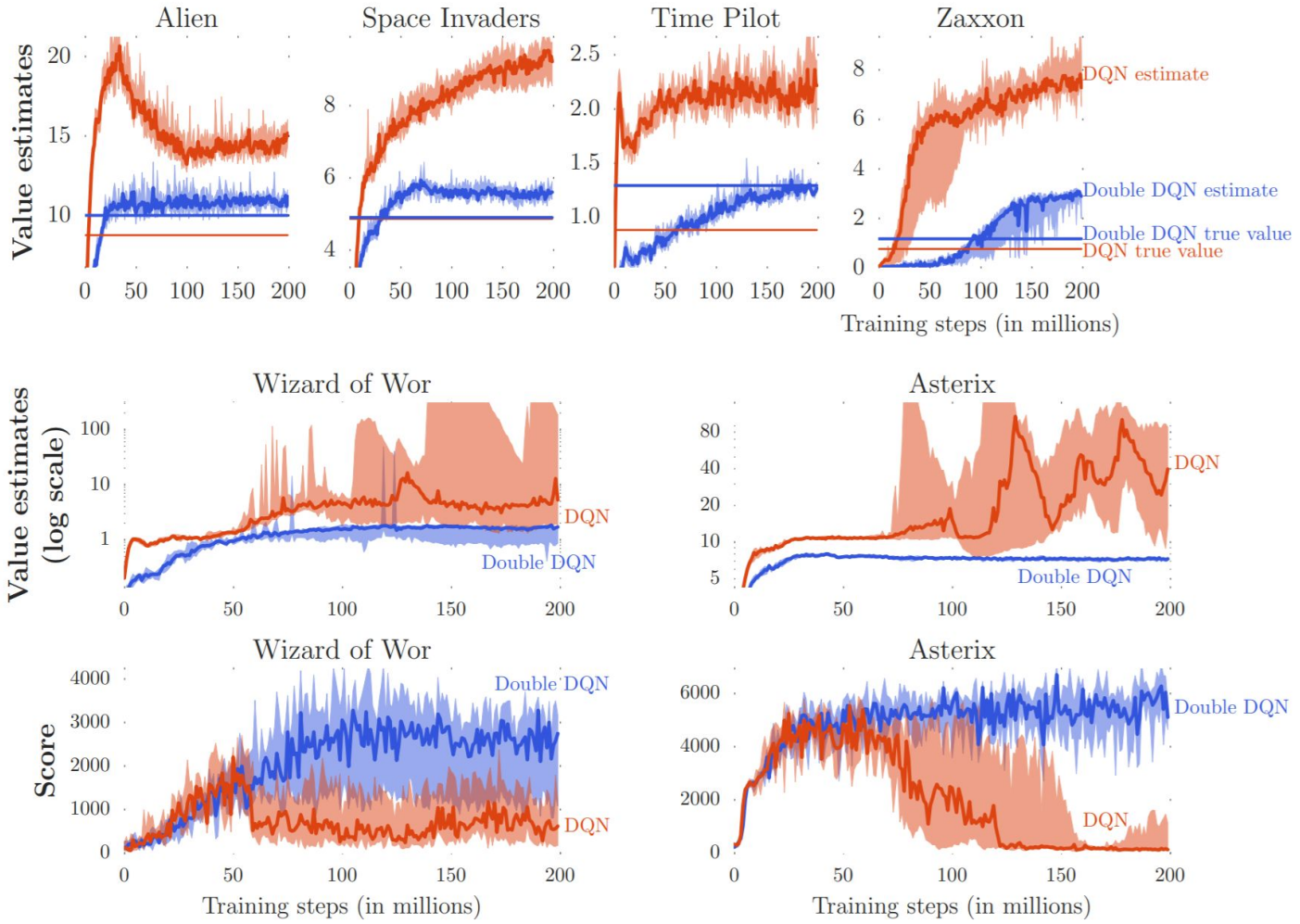
- Clearly outperform DQN, without additional computation cost or tuning.
- And the tuned version is even better

	DQN	Double DQN	Double DQN (tuned)
Median	47.5%	88.4%	116.7%
Mean	122.0%	273.1%	475.2%



Double DQN results

The Q value estimation comparison support the claim about Double DQN effectiveness on reducing errors



Dueling Network Architectures for Deep Reinforcement Learning

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, Nando de Freitas

Topic: Q-Value based RL

Presenter: Haoping Xu

Motivation: Does every action equally important?

- DQN and other methods estimate Q value in one stream
  - Means all possible actions have separate Q values, and are updated independently.
  - Resulting inefficiency state value update, as all actions' Q values needs to be changed
- Usually, most of the actions are not important
  - For example in racing games, an action is not critical unless you are about to crash
  - But the value for each state is always important as Q*(s,a) should be V*
  - To improve state value learning efficiency and ignore useless actions, estimate them separately, in terms of state value V and action advantage A

Contributions

- Dueling DQN
  - Propose a decoupled estimator architecture for state value and action advantages, to replace previous single stream Q value estimator
  - The new architecture can be used together with many existing RL methods
  - In Atari games, Dueling DQN outperforms DDQN, and with prioritized replay, it is the SOTA in ALE benchmark

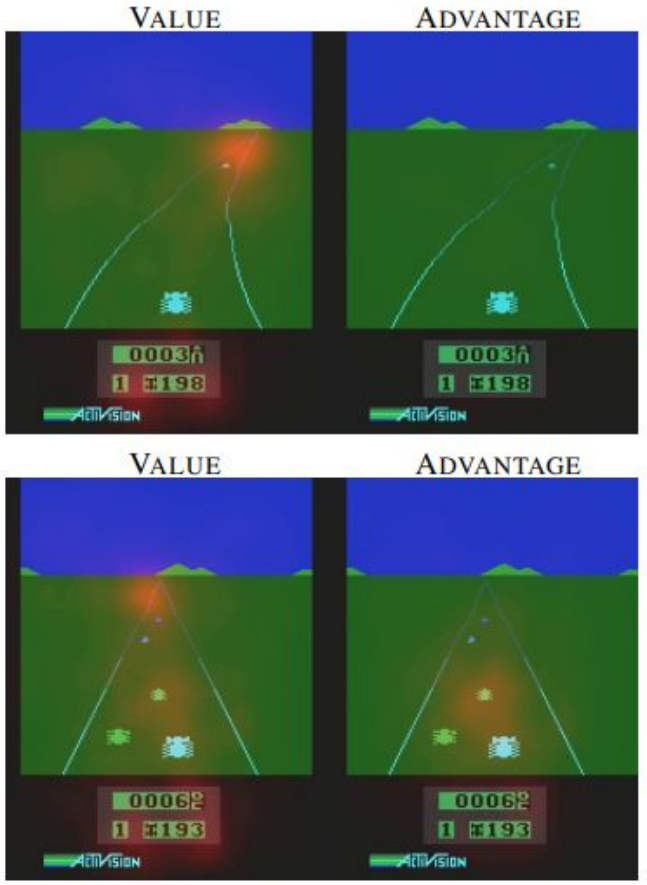
Most actions are useless

$$A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s) = Q_\pi(s,a) - \mathbb{E}_{a\sim\pi(s)}[Q_\pi(s,a)]$$

$$\mathbb{E}_{a\sim\pi(s)}[A_\pi(s,a)] = 0$$

For a deterministic policy, for example greedy $a^* = \arg\max_{a\in A} Q(s,a)$

$$Q(s,a^*) = V(s) \text{ and } A(s,a^*) = 0$$



What is the take away from this?

- State value function has a greater influence to Q value, and the performance of agent
- Advantage value for many action state pairs are not that important, as their are likely to be zero

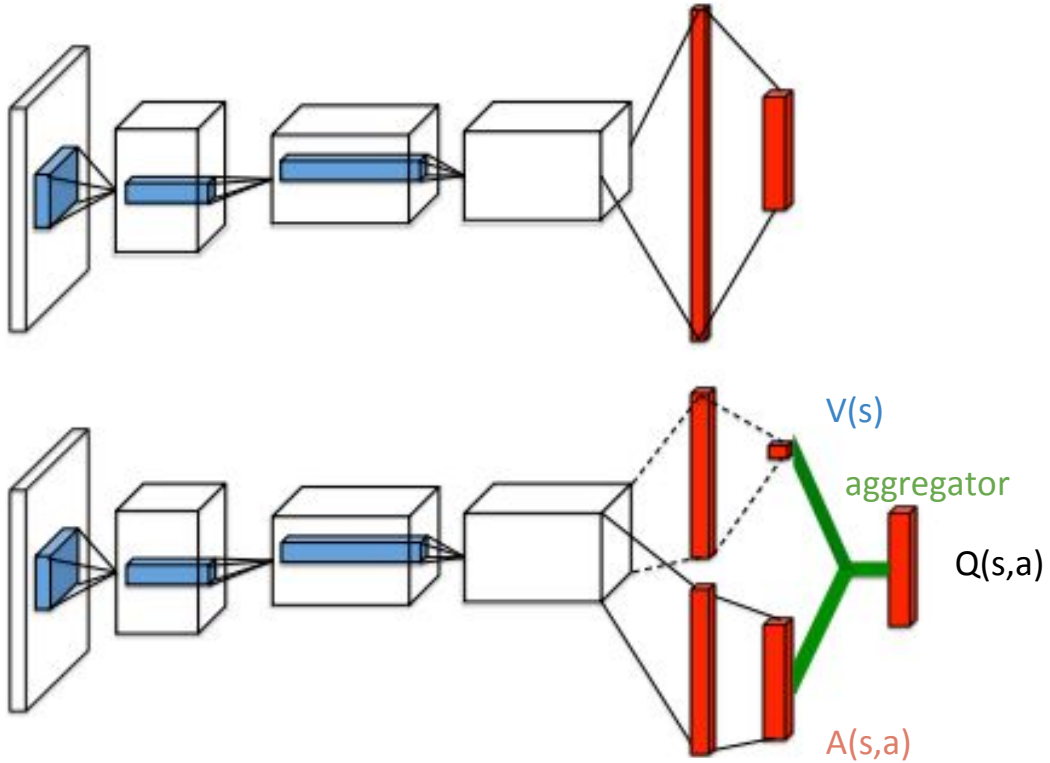
Algorithm Dueling DQN

Dueling network = CNN + two MLP that output:

- A scalar state value $V(s;\theta,\beta)$
- An |A|-dimensional advantage vector $A(s,a;\theta,\alpha)$

Decoupling Q value function into state value and advantage:

- Use aggregating module to recombine these two parts
- $Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + A(s,a;\theta,\alpha)$



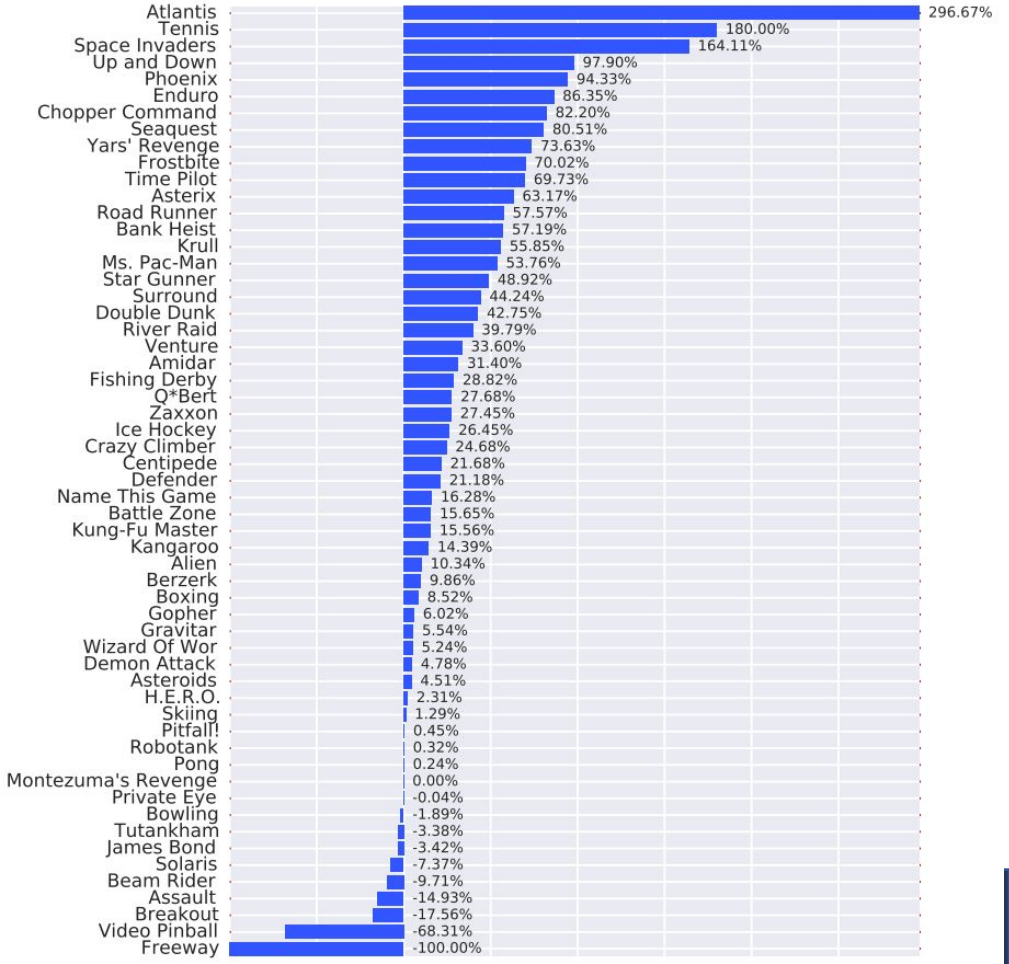
Aggregating module

- Simple add $Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + A(s,a;\theta,\alpha)$
  - Unidentifiable: give a Q, V and A are not uniquely defined
  - Not regulation on A, its expectation should be 0
- Subtract max $Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + A(s,a;\theta,\alpha) - \max_{a' \in A} A(s,a';\theta,a)$
  - When using a greedy policy, Q(s,a*) = V(s)
  - Enforce A to be zero at the chosen action
- Subtract mean $Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + A(s,a;\theta,\alpha) - \frac{1}{|A|}\sum_{a' \in A} A(s,a';\theta,a)$
  - Alternative of subtract max
  - Loss the original semantics of V and A, and off target by a constant
  - But increase stability of optimization, instead of following the optimal advantage, just need to follow the mean
- Take away:
  - Subtract mean is the best, stable + keep relative rank of A

Discussion of results

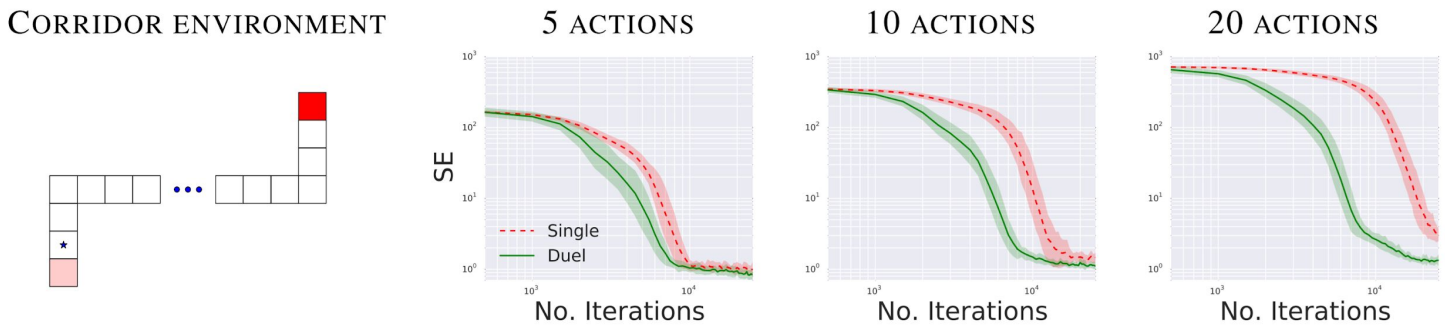
	30 no-ops		Human Starts	
	Mean	**Median**	**Mean**	**Median**
Prior. Duel Clip	**591.9%**	**172.1%**	**567.0%**	**115.3%**
Prior. Single	434.6%	123.7%	386.7%	112.9%
Duel Clip	**373.1%**	**151.5%**	**343.8%**	**117.1%**
Single Clip	341.2%	132.6%	302.8%	114.1%
Single	307.3%	117.8%	332.9%	110.9%
Nature DQN	227.9%	79.1%	219.6%	68.5%

- Outperform Double DQN in most of the settings, got SOTA when using prioritized replay and gradient clip in ALE benchmark
- The performance gain comes with minimal computation cost, as both dueling and single models are using similar amount of parameters.(2x 512 unit layers vs 1024 unit layer)



Discussion of results

- The corridor environment start from one end to red point
- artificially add more useless no-op actions in the action space
- demonstrate an increasing gap between dueling and single stream Q estimator performances



Critique / Limitations / Open Issues

- Double DQN
  - Although both estimation error lower bound and empirical results are presented, these two do not agree with each other. A theoretical analysis of typically relation between error and number of actions will be better
- Dueling DQN
  - The ability to handle no-op actions is only demonstrated by corridor environment, will be interesting to see the behavior on Atari game with expanded action space
  - The idea of saliency map on input frame is similar to attention, there are some publications on attention recurrent DQN[Ivan 2015 DARQN]

Contributions (Recap)

- Double DQN
  - Combining DQN and Double Q-learning to solve overoptimism problems for Q values
  - Provide a solid theoretical analysis of overestimation error bound in tradition Q learning
  - Demonstrate large estimation error in DQN and how DDQN fixs it and improve the performance using Atari games
  -
- Dueling DQN
  - Propose a decoupled estimator architecture for state value and action advantages, to replace previous single stream Q value estimator
  - The new architecture can be used together with many existing RL methods
  - In Atari games, Dueling DQN outperforms DDQN, and with prioritized replay, it is the SOTA in ALE benchmark

References

- Sorokin, Ivan et al. “Deep Attention Recurrent Q-Network.” *ArXiv* abs/1512.01693 (2015): n. Pag
- Hasselt, Hado van et al. “Deep Reinforcement Learning with Double Q-Learning.” AAAI (2015).
- Wang, Ziyu et al. “Dueling Network Architectures for Deep Reinforcement Learning.” ICML (2015).
- Mnih, Volodymyr et al. “Playing Atari with Deep Reinforcement Learning.” ArXiv abs/1312.5602 (2013): n. Pag.
- van Hasselt. Double Q-learning. Advances in Neural Information Processing Systems, 23:2613–2621, 2010.