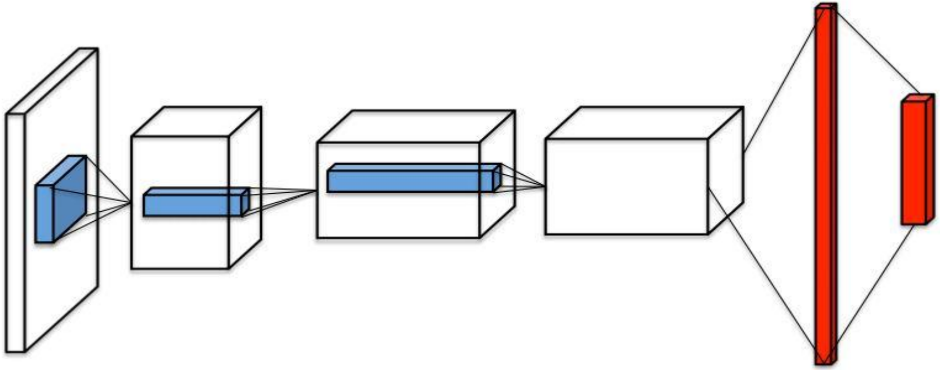
Duelling Networks

Nan Rosemary Ke

Current DQN architecture

- Uses a single network for computing the q-value updates, given a state s and action a (s, a).
- Problems:
    - Existing deep neural network architectures may not be suited for Reinforcement Learning
    - The state value function and the advantage function updates at different rates.

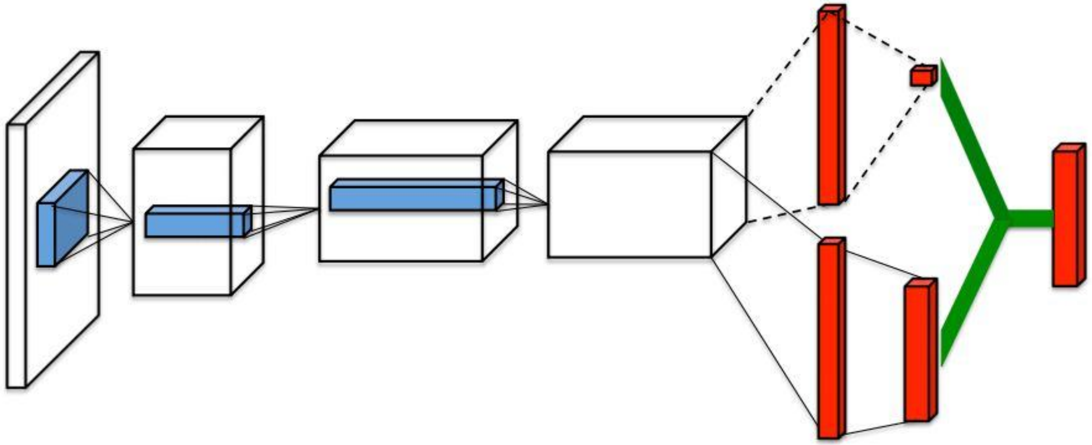


Motivation

- Novel Deep Neural Network architectures build for Model-free RL problems.
- Having separate network approximating the State-value function approximator and the advantage update function.
    - State-value function and advantage-value function updates at different pace.

Duelling Network architecture

- Shared Convolutional feature learning module. Separate state value and advantage function approximator.
    - No extra supervision.
    - Learns which state is valuable, do not have to learn action-value for all action for all states



Duelling network

Definitions

- State value $V(s)$
- Advantage function $A(s, a)$
- Policy π
- Return $R_t = \sum_{\tau=t}^{\infty} \gamma^{\tau-t} r_\tau, \gamma \in [0, 1]$
- Q function $Q^\pi(s, a) = E[R_t | S_t = s, a_t = a, \pi]$
- State-value $V^\pi(s) E_{a\sim\pi(s)} [Q^\pi(s, a)]$

Bellman equation vs Advantage equation

- Bellman equation

$$Q^*(s,a) = \mathbb{E}_{s`}\left[r + \gamma \max_{a`} Q^*(s`,a`) \,|s,a\right]$$

- Advantage equation

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

- $\mathbb{E}_{a\sim\pi(s`)}[A^\pi(s,a)] = 0$

Duelling network - intuition

- Value $V(s)$ computes **how good it is to be in that particular state**
- Q(s,a) computes, given in state s, how **good is action a**.
- A = V - Q computes the **relative importance of each action**

Duelling network separates the computation of the value of the advantage function, since they update at different rates.

Duelling network - Key insights

- For many states
  - Not necessary to compute all action values.
  - In many states, the action has no effect.
- Bootstrapping algorithm
  - Computation of state value is extremely important
  - Bootstrapping: updating estimates based on other estimates (other estimate should be fairly accurate)

Duelling network - Formulation

- $A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s)$
- $V^{\pi}(s) = \mathbb{E}_{a \sim \pi(s)}[Q^{\pi}(s,a)]$
  - $A^{\pi}(s,a) = Q^{\pi}(s,a) - \mathbb{E}_{a \sim \pi(s)}[Q^{\pi}(s,a)]$
- $\mathbb{E}_{a \sim \pi(s)}[A^{\pi}(s,a)] = 0$
- For a deterministic policy, $a^{*} = \arg\max_{a` \in \mathcal{A}} Q(s, a`)$
  - $Q(s,a^{*}) = V(s)$ and $A(s,a^{*}) = 0$

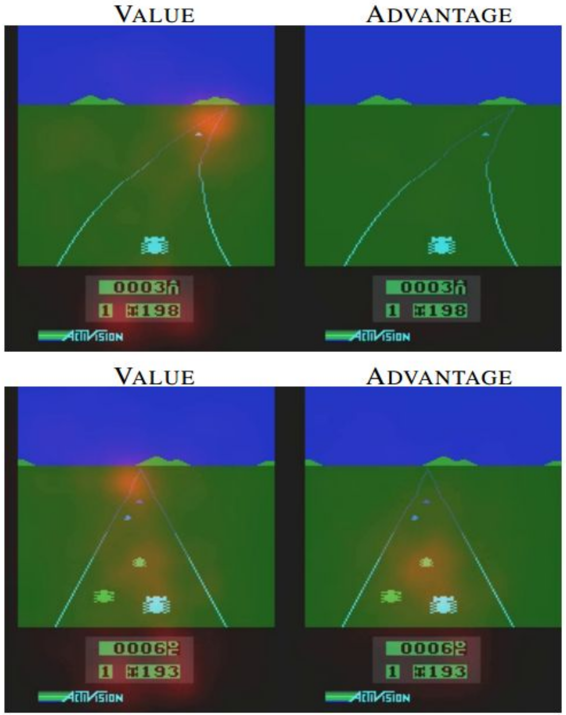
Duelling network - formulation

- Architecture
  - CNN + fully connected layers that output
    - Scalar V(s)
    - Vector A(s, a)
- Tempt to construct aggregation module
  - $$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + A(s, a; \theta, \alpha)$$

Results in the paper

When new car is

Coming, focus on horizon/



Don’t pay attention when there is no car

Focus on the score

Attention on the car immediately at the front. Making its choice of actions relevant.

My Experiments

- Architecture
  - 2 feedforward dense layers (layer size 100)
  - Separate Q and A layer, merges into a single action-value function estimator
- Optimizer:
  - Adam
  - Learning rate: 0.001
- Game : Carpool

My Experiments

Espisodes	Time steps	Reward
1	18	18
2	28	28
3	48	48
4	200	200
5	200	200
6	200	200
7	200	200
8	200	200
9	200	200

My Experiments

Carpole - Reward over episodes