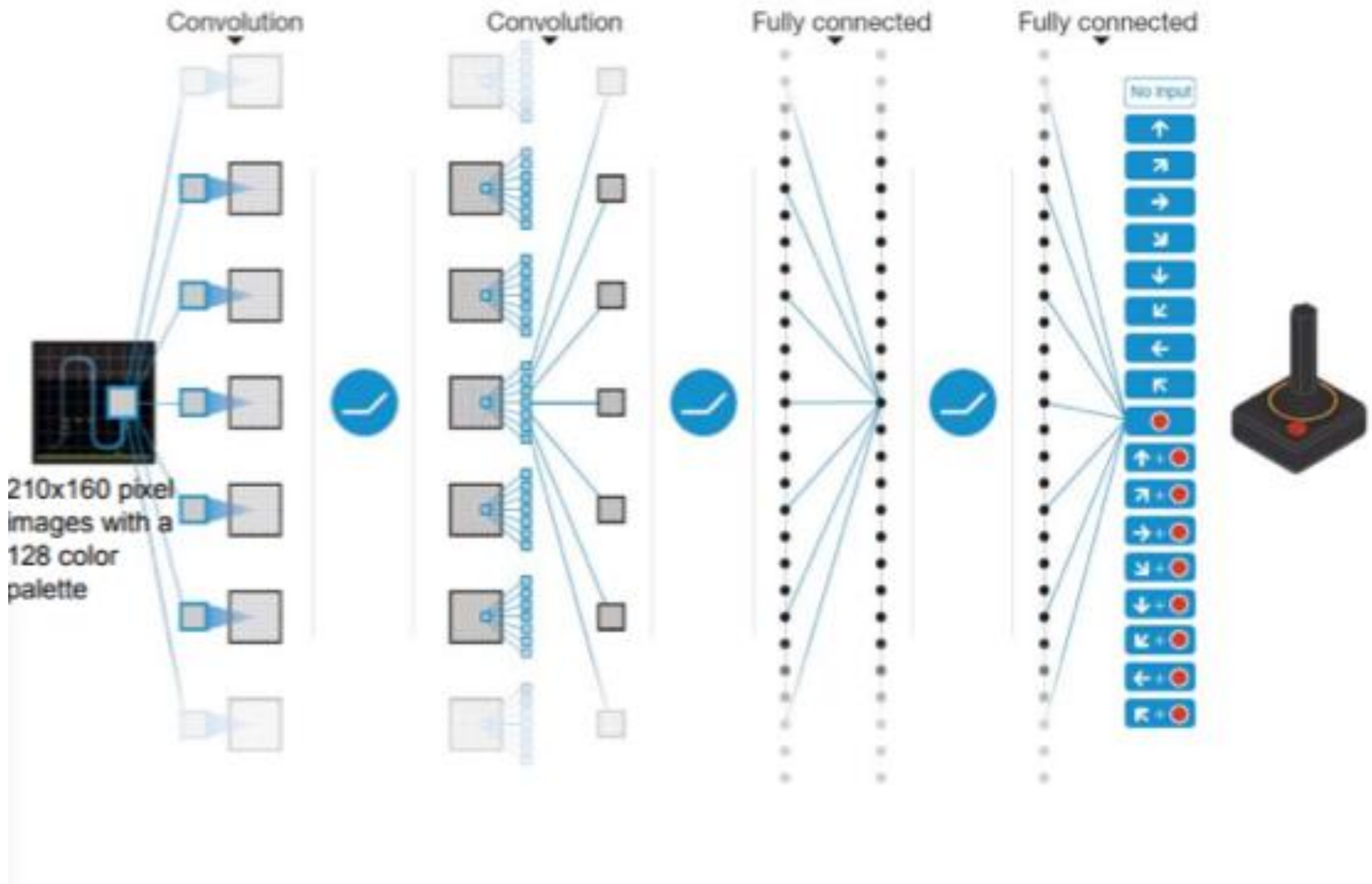
Dueling Network Architectures for Deep Reinforcement Learning

Google DeepMind
ICML 2016

DQN



- The input to the neural network consists of an 84x84x4 image produced by the pre-processing map ϕ
- Input state is stack of raw pixels from last 4 frames

DQN:Experience Replay

To remove correlations, build data-set from agent's own experience

- Take action a_t according to ε-greedy policy

  (Choose "best" action with probability 1- ε, and selects a random action with probability ε)

- Store transition $(s_t, a_t, r_{t+1}, s_{t+1})$ in replay memory $\mathcal{D}$ (Huge data base to store historical samples)
- Sample random mini-batch of transitions (s, a, r, s') from $\mathcal{D}$
- Optimize MSE between Q-network and Q-learning targets, e.g.

$$\mathcal{L}_i(\theta_i) = \mathbb{E}_{s,a,r,s' \sim \mathcal{D}}\left[\left(\underbrace{r + \gamma \max_{a'} Q(s', a'; \theta_i)}_{\text{target}} - Q(s, a; \theta_i)\right)^2\right]$$

DQN:Fixed target Q-network

To avoid oscillations, fix parameters used in Q-learning target

- Compute Q-learning targets w.r.t. old, fixed parameters θ_i^-

$$r + \gamma \max_{a'} Q(s', a'; \theta_i^-)$$

- Optimize MSE between Q-network and Q-learning targets

$$\mathcal{L}_i(\theta_i) = \mathbb{E}_{s,a,r,s' \sim \mathcal{D}} \left[\left(r + \gamma \max_{a'} Q(s', a'; \theta_i^-) - Q(s, a; \theta_i) \right)^2 \right]$$

- Periodically update fixed parameters $\theta_i^- \leftarrow \theta_i$

Improvement

DDQN:Double Deep Q-Network

$$a^* = \max_a Q(s,a,\theta_i)$$

DQN target: $y_i^{DQN} = r + \gamma \max_{a'} Q(s', a'; \theta^-),$

The max operator uses the same values to both select and evaluate an action.

This can therefore lead to overoptimistic value estimates

DDQN target: $y_i^{DDQN} = r + \gamma Q(s', \arg\max_{a'} Q(s', a'; \theta_i); \theta^-).$

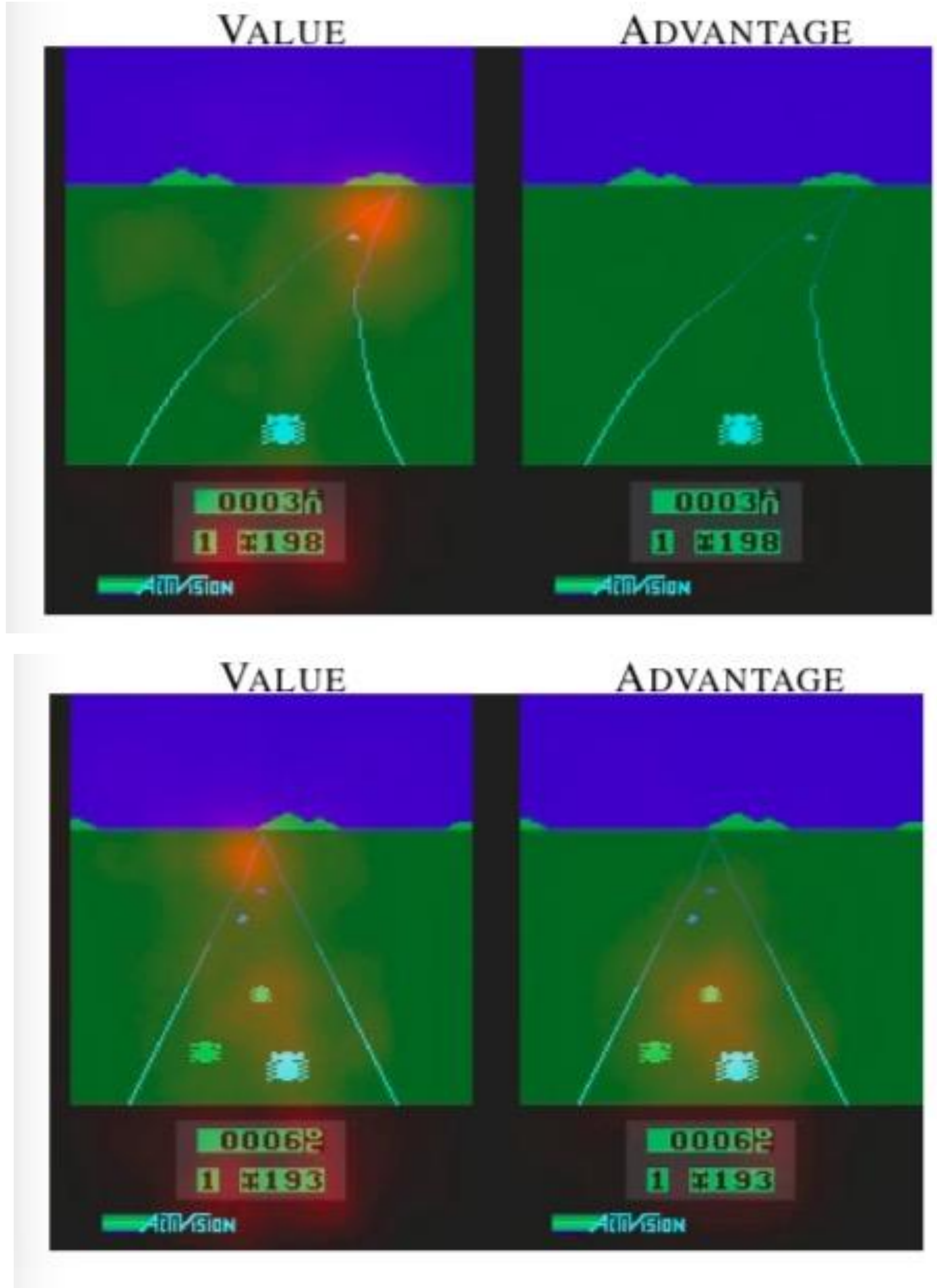
DDQN is the same as DQN, except that the action was chosen from another network.

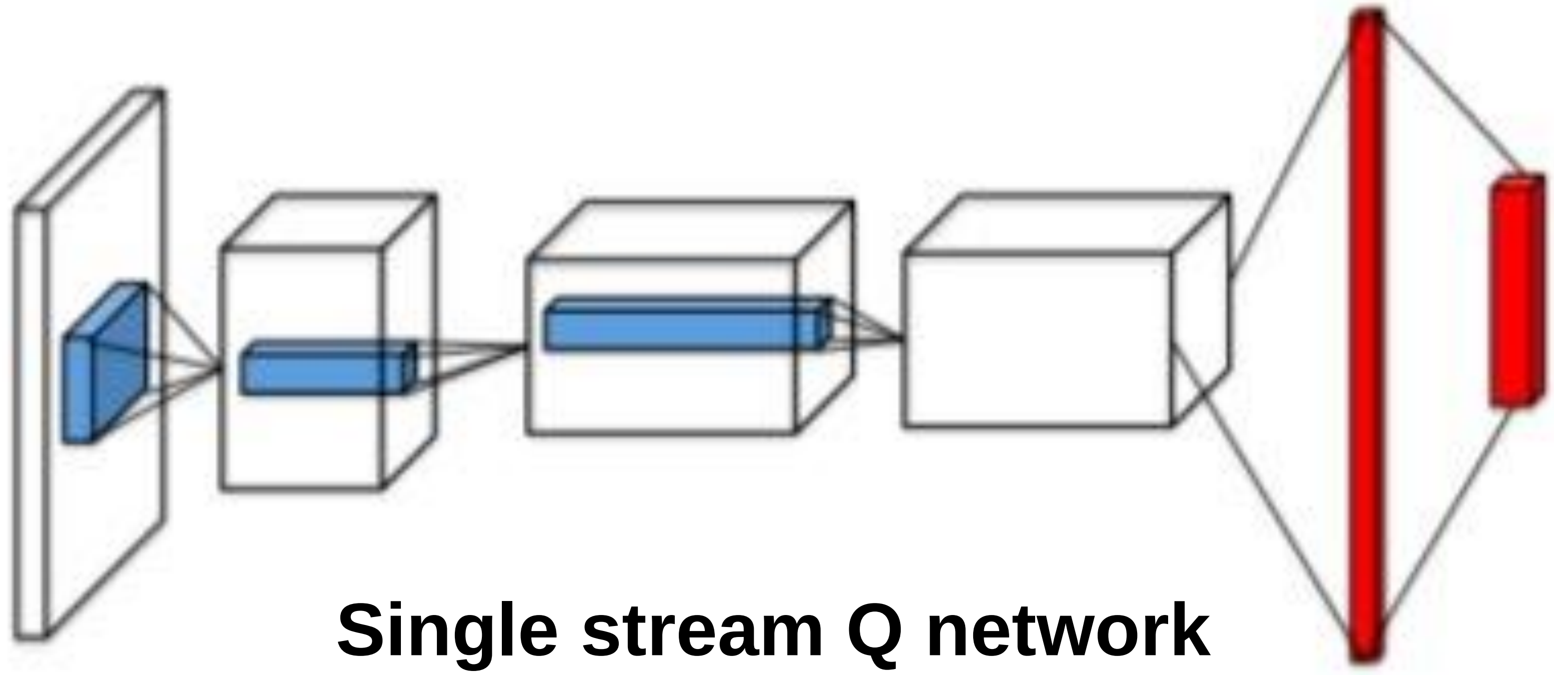
Improvement

Prioritized Replay

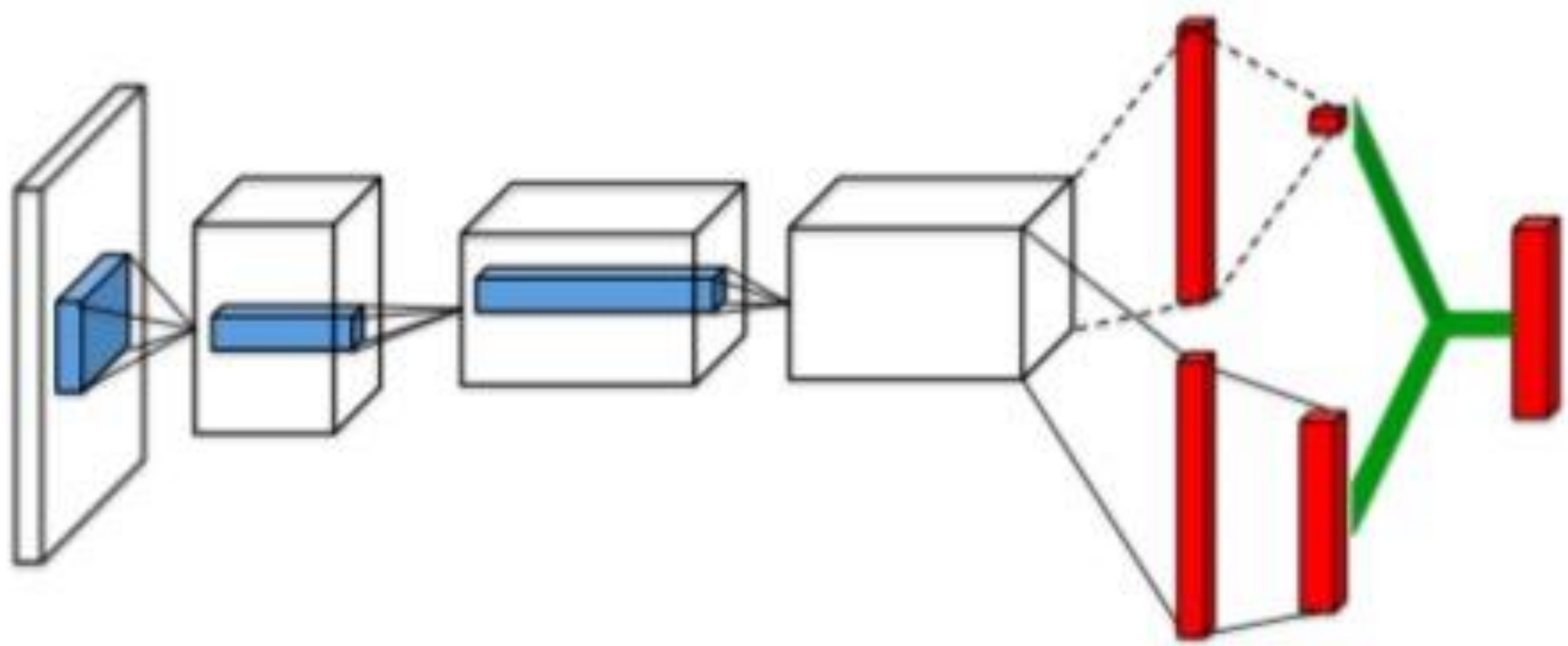
Instead of choosing tuples randomly to update policy, they increase the replay probability of experience tuples that have a high expected learning progress , as measured by TD error

Motivation





Single stream Q network

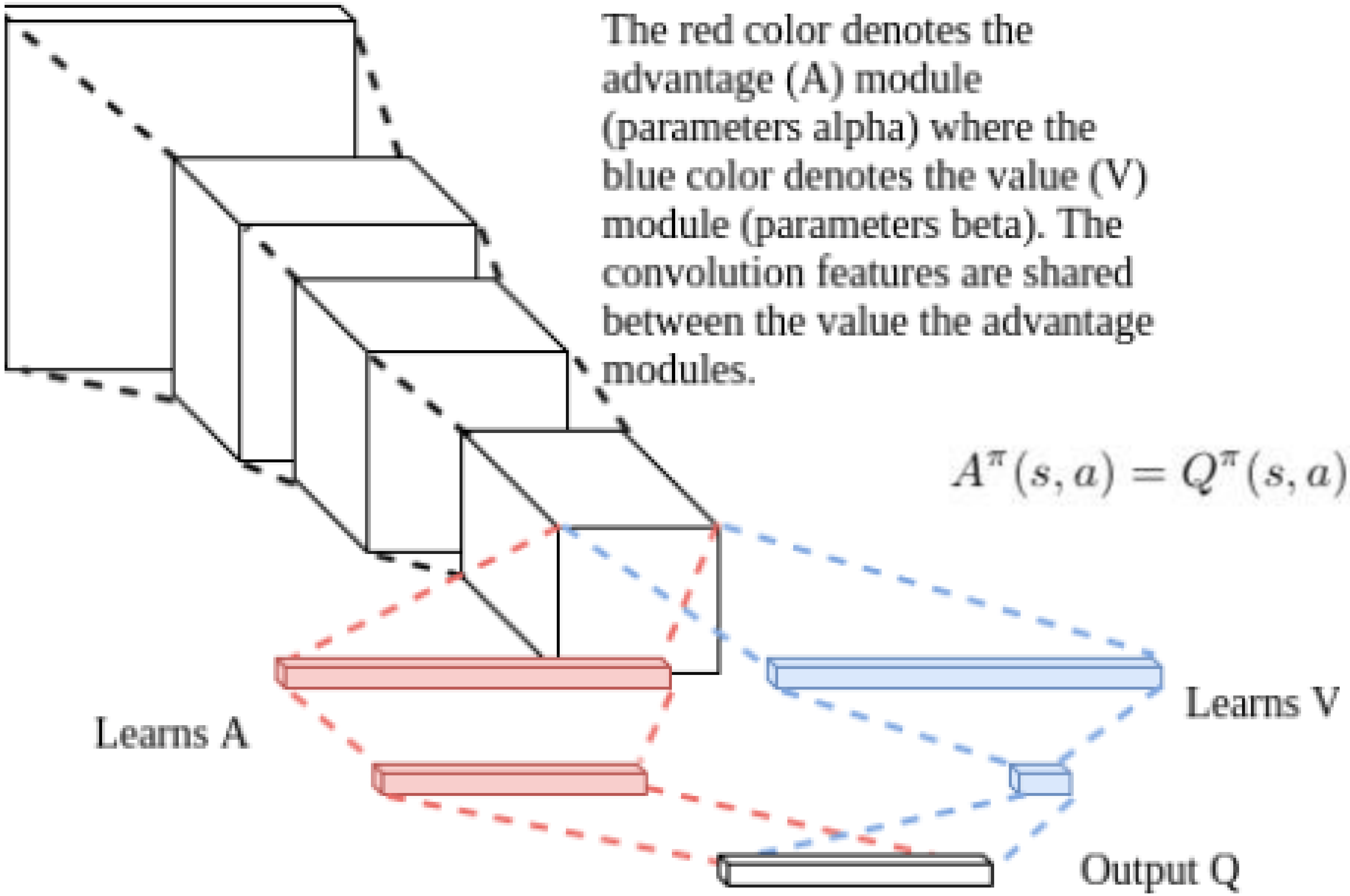


Dueling Q network

Duel Network

The red color denotes the advantage (A) module (parameters alpha) where the blue color denotes the value (V) module (parameters beta). The convolution features are shared between the value the advantage modules.

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



Design of output layer

$$Q(s, a; \theta, \alpha, \beta) = V(s; \theta, \beta) + A(s, a; \theta, \alpha),$$

Issue of identifiability:

given Q, we can not recover V and A uniquely.

we can not say V is a good estimate of state-value function and A is a reasonable estimate of the advantage function since we can distinguish V and A from loss function.

To address this issue of identifiability,

we can force the advantage function estimator to have zero advantage at the chosen action

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

An alternative module replaces the max operator with an average:

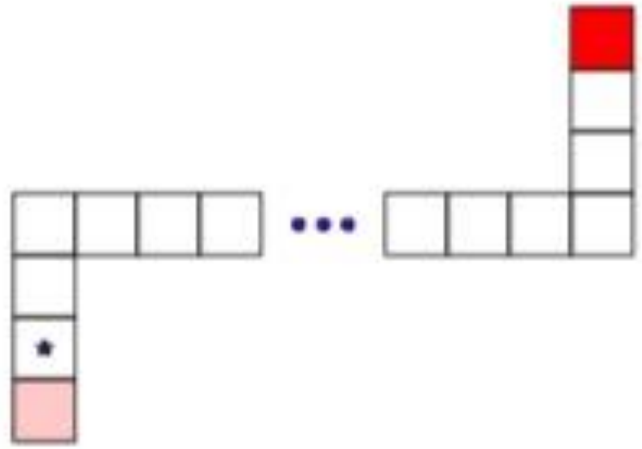
$$Q(s,a;\theta,\alpha,\beta) = V(s;\theta,\beta) + \left(A(s,a;\theta,\alpha) - \frac{1}{|\mathcal{A}|}\sum_{a'} A(s,a';\theta,\alpha) \right).$$

1.increase the stability of optimization

2.the advantage only need to change as fast as the mean ,instead of hav

Experiment:Policy Evaluation

CORRIDOR ENVIRONMENT

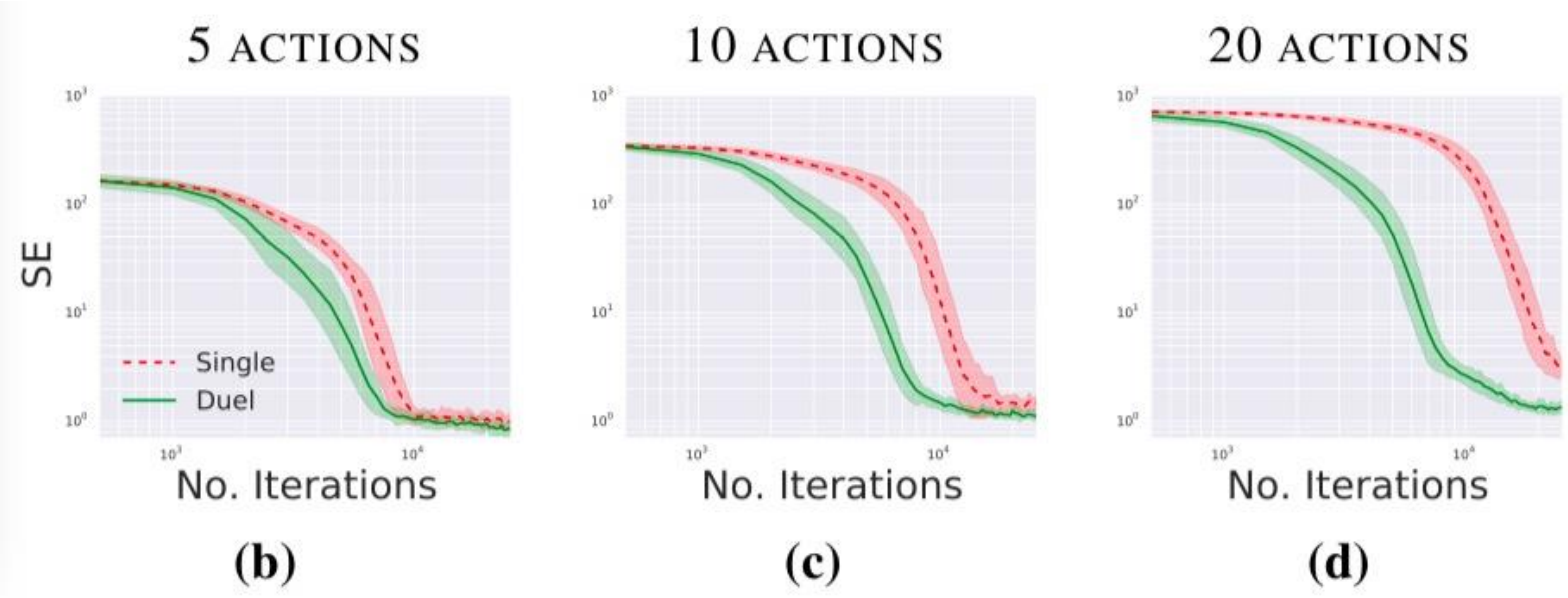


5 actions:go up, down, left, right and no-op

States:The star marks the start state, the two vertical sections both have 10 states while the horizontal section has 50 states.

Rewards:top right red square has the highest reward

Experiment Result:



SE: squared error against true state values.

$$\sum_{s\in\mathcal{S},a\in\mathcal{A}}(Q(s,a;\theta) - \; Q^{\pi}(s,a))^2$$

Experiment:Atari Game Playing

Table 1. Mean and median scores across all 57 Atari games, measured in percentages of human performance.

	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%

Conclusion

1. **Learn state-value function efficiently**

2. **This architecture is robust to reordering of actions which is cause by the noise in Q function.**