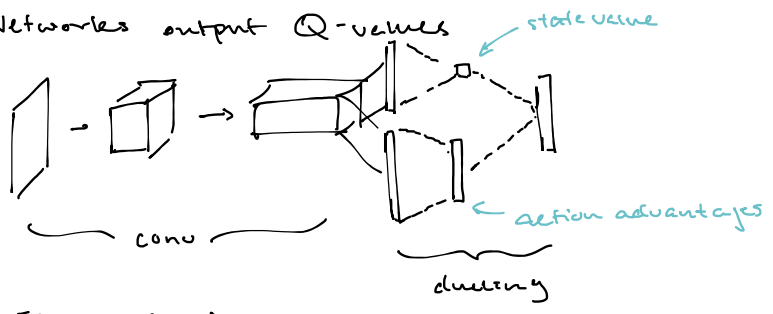
DUELING NETWORK ARCHITECTURES FOR DEEP REINFORCEMENT LEARNING

▶ model-free RL

* Factor state value function V & state dependent actor advantage function A into two separate streams.
* Networks output Q-values



* $V(s)$, $A(s,a)$

  Advantage updating converged faster than Q-learning

2. Background

video frame

* Agent gets input (M frames) of $s_t = (x_{t-M+1}, \dots, x_t) \in S$

* maximize expected discounted return.

$$\mathbb{E}\left[R_t = \sum_{\tau=t}^{\infty} \gamma^{\tau - t} r_\tau \right]$$

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

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

$$Q^\pi(s,a) = \mathbb{E}_{s'}\left[r + \gamma \mathbb{E}_{a' \sim \pi(s')} \left[Q^\pi(s',a') \right] \mid s, a, \pi \right]$$

(f: perform an action)

For optimal Q value, Q^*

(w/ deterministic policy)

$$a = \mathop{\text{argmax}}_{a' \in A} Q^*(s,a') \Rightarrow \mathbb{E}_{a' \sim \pi(s')}\left[Q^*(s',a') \right] = \max_{a' \in A} Q^*(s',a')$$

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

* Advantage function (remove state's influence on the value of actions.)

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

$$\mathop{\mathbb{E}}_{a \sim \pi(s)}\left[A^\pi(s,a) \right] = 0$$ ☑ PROVE THIS!

(a depends on s.)

2.1 DQN

* NN := $Q(s,a;\theta)$.
* At iteration i, loss function is

$$L_i(\theta_i) = \mathop{\mathbb{E}}_{s,a,r,s'}\left[\left(y_i^{DQN} - Q(s,a;\theta_i) \right)^2 \right]$$

$$y_i^{DQN} = r + \gamma \max_{a'} Q(s',a';\theta^-)$$ ← target is the value of the best action/state in the target network

(target network (frozen weights).)

$$\nabla_{\theta_i} L_i(\theta_i) = \mathop{\mathbb{E}}_{s,a,r,s'}\left[\left(y_i^{DQN} - Q(s,a;\theta_i) \right) \nabla_{\theta_i} Q(s,a;\theta_i) \right]$$

(standard NN technique)

$(s,a,r,s') \sim \mathcal{U}(D)$ (uniform; D: dataset of experiences.) ≫ reduces correlation among samples.

2.2 Double DQN [van Hasselt et al (2015)]

* Q-learning & DQN, 'max' operator uses same values to both select & evaluate an action.

$$y_i^{DDQN} = r + \gamma Q\left(s', \mathop{\text{argmax}}_{a'} Q(s,a;\theta_i); \theta^- \right)$$

(evaluate; select; evaluate best action in particular.)

y_i^{DQN} selected the action & evaluated it — bias!

2.3 Prioritized Replay [Schaul 2016]

* increase replay probability of experience tuples that have high expected learning progress (proxy: absolute TD-error).

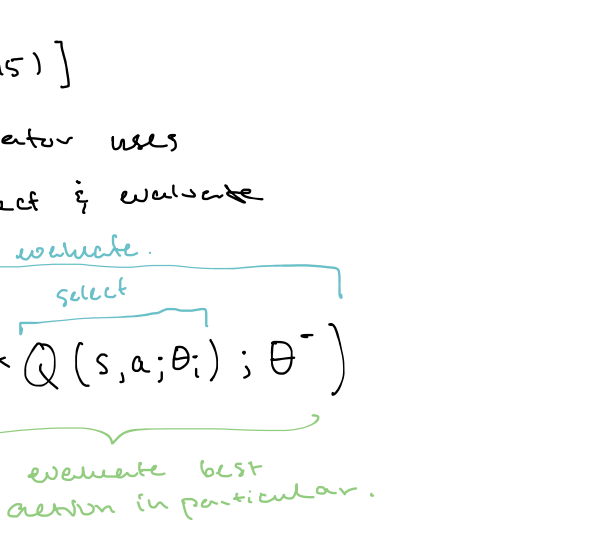
3 Dueling Network Architecture

* evaluating value of action choice doesn't matter for many states.

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

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

$$\Rightarrow \mathop{\mathbb{E}}\left[A^\pi(s,a) \right] = 0.$$



* $V(s;\theta,\beta)$ → β is fully connected

  $A(s,a;\theta,\alpha)$ → α is fully connected

* $Q(s,a;\theta,\alpha,\beta) =$

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

bad b/c we can't identify V & A uniquely ($V+\delta$, $A-\delta$) $\forall \delta$.

remedy; subtract out $\left[\max_{a' \in A} A(s,a';\theta,\alpha) \right]$ (chosen action) → now $Q(s,a^*;\theta,\alpha,\beta) = V(s;\theta,\beta)$.

if we subtract (mean) → $\frac{1}{|A|} \sum_{a' \in A} A(s,a';\theta,\alpha)$

instead, Q will be off-target, but more stable.

4. Experiments

4.1 Policy Evaluation