RL: Deep

Dueling Networks

Marius Lindauer

Leibniz Universität Hannover

AutoML.org
Automated Machine Learning Hannover

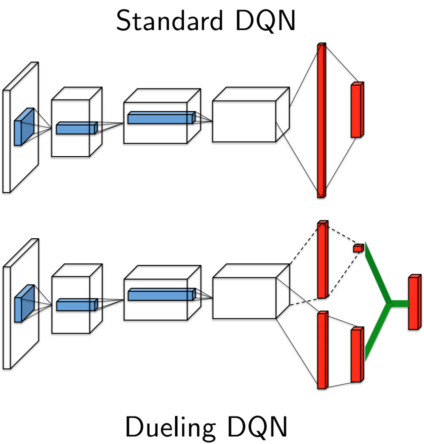
Value & Advantage Function

- Intuition: Features need to accurate represent value may be different than those needed to specify difference in actions
- E.g.
  - Game score may help accurately predict $V(s)$
  - But not necessarily in indicating relative action values $Q(s, a_1)$ vs $Q(s, a_2)$
- Advantage function [Baird 1993]

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

Automated Machine Learning Hannover

Dueling DQN [Wang et al. 2016]



- Above head predicts $V(s)$
- Heads below predicts $A(s, a_1)$, $A(s, a_2)$, ...
- Combination: $Q(s, a_1)$, $Q(s, a_2)$, ...

Dueling DQN [Wang et al. 2016]

- Advantage function [Baird 1993]

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

- Consider a network that outputs $V(s; \mathbf{w}_1, \mathbf{w}_2)$ as well as advantage $A(s, a; \mathbf{w}_1, \mathbf{w}_3)$ where $\mathbf{w}_i$ are the weights of the different parts of the network
- To construct Q could use

$$Q(s, a; \mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3) = V(s; \mathbf{w}_1, \mathbf{w}_2) + A(s, a; \mathbf{w}_1, \mathbf{w}_3)$$

Dueling DQN [Wang et al. 2016]

- Advantage function [Baird 1993]

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

- Consider a network that outputs $V(s; \mathbf{w}_1, \mathbf{w}_2)$ as well as advantage $A(s, a; \mathbf{w}_1, \mathbf{w}_3)$ where $\mathbf{w}_i$ are the weights of the different parts of the network
- To construct Q could use

$$Q(s, a; \mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3) = V(s; \mathbf{w}_1, \mathbf{w}_2) + A(s, a; \mathbf{w}_1, \mathbf{w}_3)$$

- Challenge: There doesn't have to be a unique advantage function

Automated Machine Learning Hannover

Uniqueness

- Consider a network that outputs $V(s; \mathbf{w}_1, \mathbf{w}_2)$ as well as advantage $A(s, a; \mathbf{w}_1, \mathbf{w}_3)$
- To construct Q could use

$$Q(s, a; \mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3) = V(s; \mathbf{w}_1, \mathbf{w}_2) + A(s, a; \mathbf{w}_1, \mathbf{w}_3)$$

- Option 1: Force $Q(s, a) = V(s)$ for the best action suggested by the advantage:

$$\hat{Q}(s, a; \mathbf{w}) = \hat{V}(s; \mathbf{w}) + \left(\hat{A}(s, a; \mathbf{w}) - \max_{a' \in \mathcal{A}} \hat{A}(s, a'; \mathbf{w})\right)$$

  - This helps to force the V network to approximate V
- Option 2: Use mean as baseline (more stable)

$$\hat{Q}(s, a; \mathbf{w}) = \hat{V}(s; \mathbf{w}) + \left(\hat{A}(s, a; \mathbf{w}) - \frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A}} \hat{A}(s, a'; \mathbf{w})\right)$$

  - More stable often because averaging over all advantages instead of the advantage of the current max action.