Function-Approximated Q-Learning Demo

Rishikesh Vaishnav

July 1, 2018

Basic Implementation

Code

- The code for this project is available at: .

Implementation Details

- This algorithm implements function approximation with Q-learning where the behavioral policy is ϵ-greedy w.r.t. the current state-value approximation $\hat{q}(s, a; \theta)$.
- Although not theoretically proven to converge, this algorithm has been known to converge empirically.
- Each state-action pair is converted to the feature vector $x(s, a)$. Letting S_{obs} and S_{act} be the size of the observation and action spaces, respectively, the size of the vector is $S_{obs} \times S_{act}$, where all features are 0 except for the S_{obs} features starting at index $S_{obs} \times a$, which are set to the environment's parameterization of s.
  - In this case, $S_{obs} = 4$ and $S_{act} = 2$.
- The action-value function $\hat{q}(s, a; \theta)$ performs a parameterized linear mapping of feature vectors:

$$\hat{q}(s, a; \theta) = \theta^T x(s, a)$$

- The gradient of the action-value function is:

$$\nabla_\theta \hat{q}(s, a; \theta) = x(s, a)$$

- Simplifying the update rule:

$$\begin{aligned}\theta_{t+1} &= \theta_t + \alpha \left(R_{t+1} + \gamma \max_a \hat{q}(S_{t+1}, a, \theta_t) - \hat{q}(S_t, A_T, \theta_t) \right) \nabla_\theta \hat{q}(S_t, A_t; \theta) \\ &= \theta_t + \alpha \left(R_{t+1} + \gamma \max_a \hat{q}(S_{t+1}, a, \theta_t) - \hat{q}(S_t, A_T, \theta_t) \right) x(S_t, A_t)\end{aligned}$$

Results

- The results can be summarized as follows:
  -