MDPs: SARSA

Using the utility

Data (following policy $\pi(s) = \text{stay}$):

[in; stay, 4, end]	$u = 4$
[in; stay, 4, in; stay, 4, end]	$u = 8$
[in; stay, 4, in; stay, 4, in; stay, 4, end]	$u = 12$
[in; stay, 4, in; stay, 4, in; stay, 4, in; stay, 4, end]	$u = 16$

Algorithm: model-free Monte Carlo

On each (s, a, u):

$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta \underbrace{u}_{\text{data}}$

Using the reward + Q-value

Current estimate: $\hat{Q}_\pi(s, \text{stay}) = 11$

Data (following policy $\pi(s) = \text{stay}$):

[in; stay, 4, end]	$4 + 0$
[in; stay, 4, in; stay, 4, end]	$4 + 11$
[in; stay, 4, in; stay, 4, in; stay, 4, end]	$4 + 11$
[in; stay, 4, in; stay, 4, in; stay, 4, in; stay, 4, end]	$4 + 11$

Algorithm: SARSA

On each (s, a, r, s', a'):

$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta[\underbrace{r}_{\text{data}} + \gamma \underbrace{\hat{Q}_\pi(s',a')}_{\text{estimate}}]$

- Broadly speaking, reinforcement learning algorithms interpolate between new data (which specifies the **target** value) and the old estimate of the value (the **prediction**).
- Model-free Monte Carlo's target was u, the discounted sum of rewards after taking an action. However, u itself is just an estimate of $Q_\pi(s,a)$. If the episode is long, u will be a pretty lousy estimate. This is because u only corresponds to one episode out of a mind-blowing exponential (in the episode length) number of possible episodes, so as the epsiode lengthens, it becomes an increasingly less representative sample of what could happen. Can we produce a better estimate of $Q_\pi(s,a)$?
- An alternative to model-free Monte Carlo is SARSA, whose target is $r + \gamma \hat{Q}_\pi(s', a')$. Importantly, SARSA's target is a combination of the data (the first step) and the estimate (for the rest of the steps). In contrast, model-free Monte Carlo's u is taken purely from the data.

Model-free Monte Carlo versus SARSA

Key idea: bootstrapping

SARSA uses estimate $\hat{Q}_\pi(s,a)$ instead of just raw data u.

u	$r + \hat{Q}_\pi(s', a')$
based on one path	based on estimate
unbiased	biased
large variance	small variance
wait until end to update	can update immediately

- The main advantage that SARSA offers over model-free Monte Carlo is that we don't have to wait until the end of the episode to update the Q-value.
- If the estimates are already pretty good, then SARSA will be more reliable since u is based on only one path whereas $\hat{Q}_\pi(s', a')$ is based on all the ones that the learner has seen before.
- Advanced: We can actually interpolate between model-free Monte Carlo (all rewards) and SARSA (one reward). For example, we could update towards $r_t + \gamma r_{t+1} + \gamma^2 \hat{Q}_\pi(s_{t+1}, a_{t+2})$ (two rewards). We can even combine all of these updates, which results in an algorithm called SARSA(λ), where λ determines the relative weighting of these targets. See the Sutton/Barto reinforcement learning book (chapter 7) for an excellent introduction.
- Advanced: There is also a version of these algorithms that estimates the value function V_π instead of Q_π. Value functions aren't enough to choose actions unless you actually know the transitions and rewards. Nonetheless, these are useful in game playing where we actually know the transition and rewards, but the state space is just too large to compute the value function exactly.

answer in chat

Question

Which of the following algorithms allows you to estimate $Q^*(s,a)$ (select all that apply)?

(a) model-based value iteration

(b) model-free Monte Carlo

(c) SARSA

- Model-based value iteration estimates the transitions and rewards, which fully specifies the MDP. With the MDP, you can estimate anything you want, including computing $Q^*(s,a)$
- Model-free Monte Carlo and SARSA are on-policy algorithms, so they only give you $\hat{Q}_\pi(s,a)$, which is specific to a policy π. These will not provide direct estimates of $Q^*(s,a)$.