CS 277: Control and Reinforcement Learning (Winter 2021)
Assignment 3

Due date: Tuesday, February 16, 2021 (Pacific Time)

Roy Fox
https://royf.org/crs/W21/CS277/

In the following questions, a formal proof is not needed (unless specified otherwise). Instead, briefly explain informally the reasoning behind your answers.

Part 1 Variance of advantage estimators (50 points + 15 bonus)

This part studies the variance of several estimators of the advantage function $A_\pi(s,a) = Q_\pi(s,a) - V_\pi(s)$. For this purpose, you will need to find a process that is, in a sense, "worst-case" for this variance. To simplify things, we will eliminate all the bias by making the following assumptions:

- We have access to the true state-value function V_π of the policy of interest π; and
- The experience (s,a,r,s') is sampled by on-policy, i.e. by rolling out π.

We will also assume that $r(s,a)$ is a deterministic function of the state s and, optionally, the action a; and that $V(s) \in [-1,1]$ (otherwise the variance can be made arbitrarily large just by growing the range of V).

We start by finding some properties that such a "worst-case" process must satisfy.

1. Consider the advantage estimator $\hat{A}(s,a;r,s') = r + \gamma V_\pi(s') - V_\pi(s)$. For given state s and action a, how should $V(s') \in [-1,1]$ be distributed such that $\hat{A}(s,a;r,s')$ will have the maximal conditional variance $\operatorname{Var}[\hat{A}(s,a;r,s')|s,a]$ that it possibly can? (5 points)
2. For the distribution that you found in the previous question, what is $\mathbb{E}[V(s')|s,a]$? Recall that, by the recursive definition for V_π, $\mathbb{E}_{a|s\sim\pi}\mathbb{E}_{s'|s,a\sim p}[\hat{A}(s,a;r,s')] = 0$. What can you conclude about $r(s,a)$? (5 points)

With these properties, we turn to find a dynamic process (an MDP + a policy) that has these properties, and therefore has "worst-case" variance in this sense. Don't worry if the process is very degenerate. In fact, try to make it as simple as possible. You may even find an MDP that is so simple that the policy doesn't make any difference.

3. Suggest a two-state dynamic process (MDP + policy) that has the worst-case conditional variance of $\hat{A}(s,a;r,s')$ in each state and action. Specify $p(s'|s,a)$, $r(s,a)$, and $\pi(a|s)$ for each s, a, and s'. (10 points)
4. In the process that you proposed in the previous question, what is the conditional variance, given s_t and a_t, of each of the following advantage estimators on on-policy experience $(s_t, a_t, r_t, s_{t+1}, \ldots)$?
   (a) $\hat{A}^{\text{MC}}(s_t,a_t) = \sum_{t' \geqslant t} \gamma^{t'-t} r_{t'} - V(s_t)$. (15 points)
   (b) $\hat{A}^n(s_t,a_t) = \sum_{t'=t}^{t+n-1} \gamma^{t'-t} r_{t'} + \gamma^n V(s_{t+n}) - V(s_t)$; which n minimizes the conditional variance under the question's assumptions? (15 points)
   (c) **Bonus:** $\hat{A}^\lambda(s_t,a_t) = (1-\lambda)\sum_{n\geqslant 1} \lambda^{n-1} \hat{A}^n(s_t,a_t) = \sum_{t' \geqslant t} (\lambda\gamma)^{t'-t} \hat{A}^1(s_{t'}, a_{t'})$; Hint: careful, these $\hat{A}$ aren't conditionally independent. (15 points)

Part 2 Advantage Actor–Critic (25 points)

In this part you'll implement several actor–critic algorithms, starting with Advantage Actor–Critic (A2C; `https://arxiv.org/abs/1602.01783`). Note: in RLlib it's easy to distribute the algorithm's execution using Ray, by setting the `num_workers` in the Tune configuration. If the distribution is asynchronous (rather than synchronous, which is the default), this would then be the A3C algorithm.

Download the code at `https://royf.org/crs/W21/CS277/A3/a2c.py`. In the function `actor_critic_loss`, write TensorFlow code that calculates a loss with 3 terms:

- An actor loss: a policy-gradient loss with pre-computed advantage estimates (`advantages`);
- A critic loss: a temporal-difference loss, the square error between the pre-computed value targets (`value_targets`) and the critic values, weighted by `critic_loss_coeff`; and
- A negative-entropy loss on the actor policy, weighted by `entropy_loss_coeff` (i.e. a slight push to *maximize* entropy). First try without it, and then add it and compare. Hint: `action_dist.entropy()` can come in handy.

In the function `postprocess_advantages`, recall that `sample_batch` is part of a single trajectory, but in this assignment we will **not** assume that it's the entire episode. The batch contains tuples (s_t, a_t, r_t, s'_t) for some consecutive steps $t = t_1, \dots, t_2$ in a trajectory. Write code that calculates the scalar `last_value_pred`, defined as $V_\phi(s'_{t_2})$, i.e. the critic's prediction of the expected return following the `next_obs` s'_{t_2} at the end of the batch in the sample. Useful: (1) `policy._value`, a function that gets an array of observations and returns a same-size array of value predictions; and (2) `dones`, a boolean array indicating episode termination in each time step (hint: why is this useful here?).

Also write NumPy code that calculates for each step the discounted one-step advantages $\hat{A}^1(s, a)$ and value targets $V_\phi(s)$ for the critic's TD-learning.

Run your code on the `CartPole-v1` environment for 1000000 time steps.

Part 3 Generalized Advantage Estimation (25 points)

Recall the definition of the GAE as $\hat{A}^\lambda(s_t, a_t) = \sum_{t' \geqslant t} (\lambda\gamma)^{t'-t} \hat{A}^1(s_{t'}, a_{t'})$. Hint: another useful expression for it was given in Lecture 5. Create a copy of `a2c.py` called `gae.py`, and change it to use $\hat{A}^\lambda$ as the advantage estimates (see `https://arxiv.org/abs/1506.02438`, also see question 4c in Part 1). Useful: recall the helper function `ray.rllib.evaluation.postprocessing.discount_cumsum`.

Run your code on `CartPole-v1` with a variety of λ values. Tip: by setting the name of the `trainer` to include the value of λ, you can easily see it later in TensorBoard.

Visualize the results in TensorBoard, and attach the resulting plots. Briefly discuss the results, including:

- What was the best value of λ in your experiments?
- What happens as $\lambda \to 0$?
- What happens as $\lambda \to 1$ in theory? What happens in practice?