[SOLUTION TEMPLATE] Assignment 2: Policy Gradients

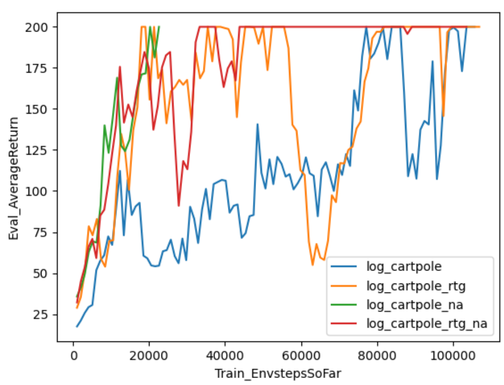
Due September 25, 11:59 pm

3 Policy Gradients

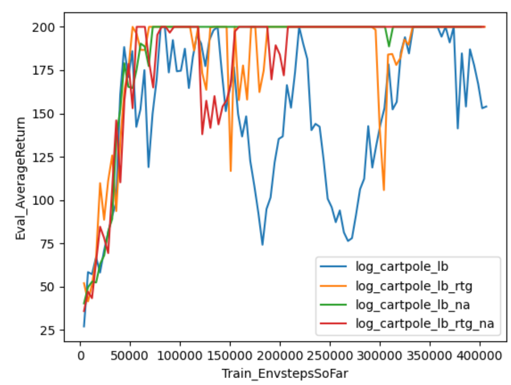
- Create two graphs:
  - In the first graph, compare the learning curves (average return vs. number of environment steps) for the experiments prefixed with `cartpole`. (The small batch experiments.)
  - In the second graph, compare the learning curves for the experiments prefixed with `cartpole_lb`. (The large batch experiments.)

  For all plots in this assignment, the x-axis should be number of environment steps, logged as `Train_EnvstepsSoFar` (*not* number of policy gradient iterations).

  My Solutions:



(a) Small batch experiments



(b) Large batch experiments

Figure 1: Learning curves for different experiments

- Answer the following questions briefly:
  - Which value estimator has better performance without advantage normalization: the trajectory-centric one, or the one using reward-to-go?
  - **My Solutions:** Without advantage normalization, using reward-to-go is better; with advantage normalization, the two ways have similar performance.
  - Did advantage normalization help?
  - **My Solutions:** Yes.
  - Did the batch size make an impact?
  - **My Solutions:** Yes, the large batch size has better performance.
- Provide the exact command line configurations (or `#@params` settings in Colab) you used to run your experiments, including any parameters changed from their defaults.

  My Solutions:

```
python cs285/scripts/run_hw2.py --env_name CartPole-v0 -n 100 -b 1000 \
       --exp_name cartpole \
       > log_cartpole.log
```

I haven't change any parameters from their defaults.

4 Neural Network Baseline

- Plot a learning curve for the baseline loss.

  My Solutions:

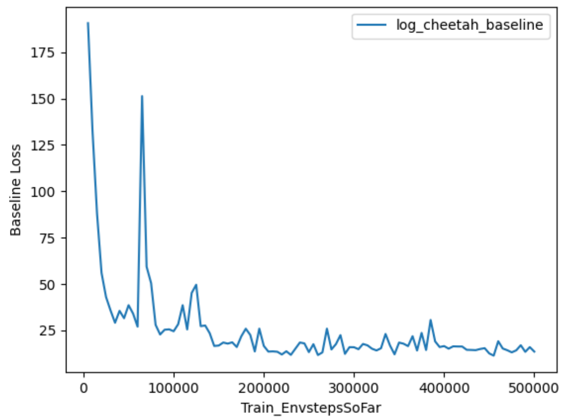


Figure 2: Learning curve for the baseline loss

- Plot a learning curve for the eval return. You should expect to achieve an average return over 300 for the baselined version.

  My Solutions:

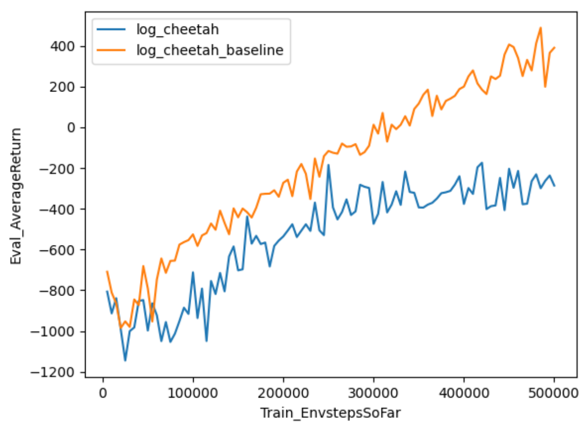
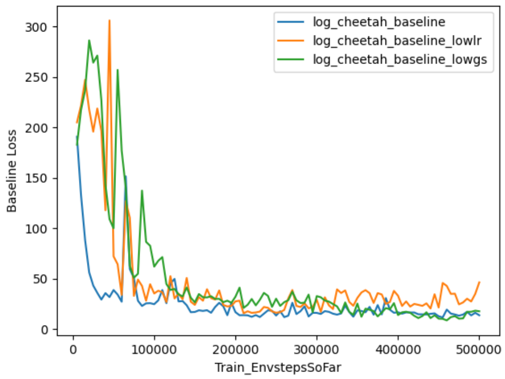


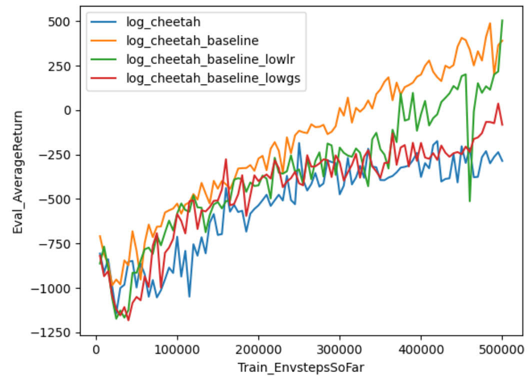
Figure 3: Learning curve for eval return

- Run another experiment with a decreased number of baseline gradient steps (`-bgs`) and/or baseline learning rate (`-blr`). How does this affect (a) the baseline learning curve and (b) the performance of the policy?

  My Solutions:



(a) Change of baseline loss



(b) Change of policy performance

Figure 4: Learning curves for different experiments

From the figures, we can see that the decreasing the learning rate of the baseline from 0.01 to 0.004 doesn't matter much to the final performance of the policy, but it harms the final baseline loss. On the other hand, decreasing the number of baseline gradient steps from 5 to 2 doesn't affect the final baseline loss much (only the convergence rate becomes slower), but it the performance of the policy is strongly harmed.

- **Optional:** Add `-na` back to see how much it improves things. Also, set `video_log_freq 10`, then open TensorBoard and go to the "Images" tab to see some videos of your HalfCheetah walking along!

5 Generalized Advantage Estimation

- Provide a single plot with the learning curves for the `LunarLander-v2` experiments that you tried. Describe in words how λ affected task performance. The run with the best performance should achieve an average score close to 200 (180+).
- Consider the parameter λ. What does $\lambda = 0$ correspond to? What about $\lambda = 1$? Relate this to the task performance in `LunarLander-v2` in one or two sentences.

6 Hyperparameter Tuning

1. Provide a set of hyperparameters that achieve high return on `InvertedPendulum-v4` in as few environment steps as possible.
2. Show learning curves for the average returns with your hyperparameters and with the default settings, with environment steps on the x-axis. Returns should be averaged over 5 seeds.

7 (Extra Credit) Humanoid

1. Plot a learning curve for the Humanoid-v4 environment. You should expect to achieve an average return of at least 600 by the end of training. Discuss what changes, if any, you made to complete this problem (for example: optimizations to the original code, hyperparameter changes, algorithmic changes).

9 Survey

Please estimate, in minutes, for each problem, how much time you spent (a) writing code and (b) waiting for the results. This will help us calibrate the difficulty for future homeworks.

- **Policy Gradients:**
- **Neural Network Baseline:**
- **Generalized Advantage Estimation:**
- **Hyperparameters and Sample Efficiency:**
- **Humanoid:**
- **Humanoid:**
- **Analysis – applying policy gradients:**
- **Analysis – PG variance:**
- **Analysis – return-to-go:**
- **Analysis – importance sampling:**