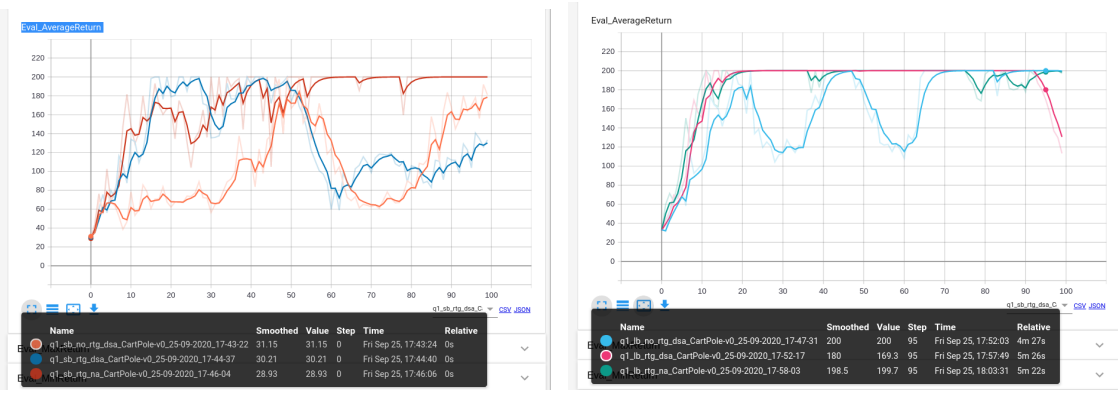
CS285 HW 2 by Tomas Santos

1 CartPole



(a) Small Batches (b) Large Batches

Figure 1: CartPole Experiment

Which value estimator has better performance without advantage-standardization: the trajectorycentric one, or the one using reward-to-go?

The plots with names containing "dsa" have advantage normalization disabled. In figure (a) this included the orange and blue plots. The blue plot which represent the "reward to go" trajectory, trains much more quickly initially but then fails to converge and drops off later. The orange plot trains more slowly and also stays at the peak for less time. Although it seems to get a better recovery the second time around, I think the performace of the blue plot which uses reward to go is better. When we look at the larger batches in figure (b) it's super clear that the reward to go dominates the trajectory centric learner. Thus I conclude that without advantage normalization reward to go has a performance that is confidently superior.

Did advantage standardization help?

The beneficial effect of normalization is most visible with small batches. The red plot is clearly the best performer. In figure (b) the green plot which represents the standardized advantaage performs nearly as good as the red plot. I think that they're close enough to call a tie.

Did the batch size make an impact?

An increase in batch size led to performace enhancements across the board. I would argue it was the most important parameter.

Provide the exact command line configurations (or #@params settings in Colab) you used to run your experiments, including any parameters changed from their defaults.

Figure 2: CartPole section of cs285/scripts/run_experiments.ipynb

2 Inverted Pendulum

Given the b* and r* you found, provide a learning curve where the policy gets to optimum (maximum score of 1000) in less than 100 iterations.

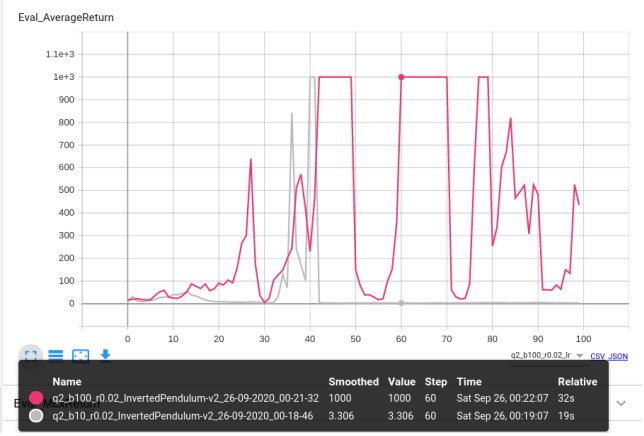


Figure 3: Pendulum Experiment

I included two runs because although using a batch size of 10 and a learning rate of 0.02 (the grey line) technically reaches 1000, it barely manages and then it crashes and never comes close again. The pink line on the other hand (b=100, lr=.02) reached 1000 mark pretty consistently relative to how much fluctuation there was for every run in the pendulum experiement.

Provide the exact command line configurations you used to run your experiments

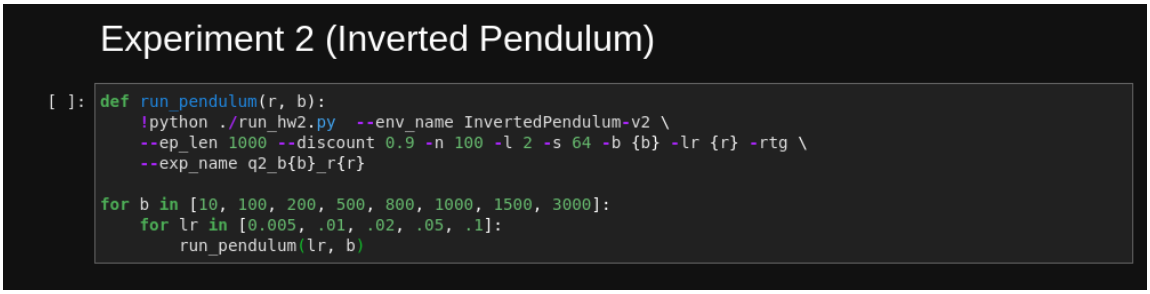


Figure 4: Pendulum section of cs285/scripts/run_experiments.ipynb

3 Lunar Landing

Plot a learning curve for the above command. You should expect to achieve an average return of around 180 by the end of training.

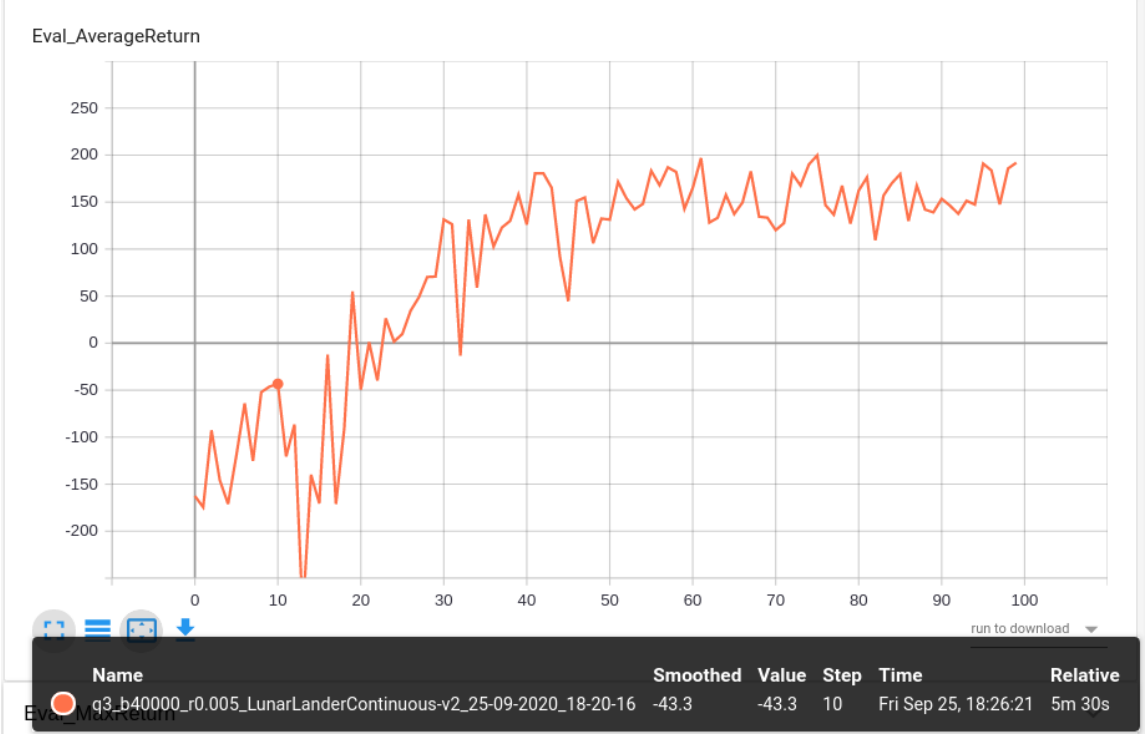


Figure 5: Lunar Landing Experiment

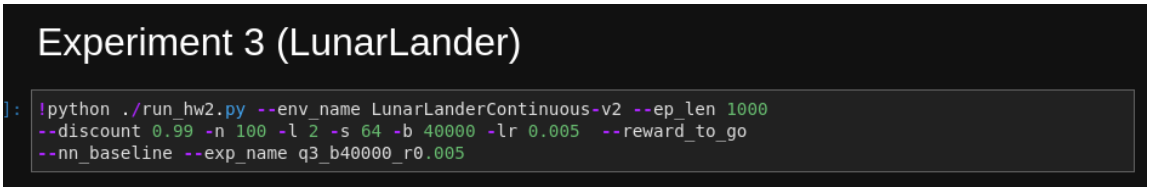
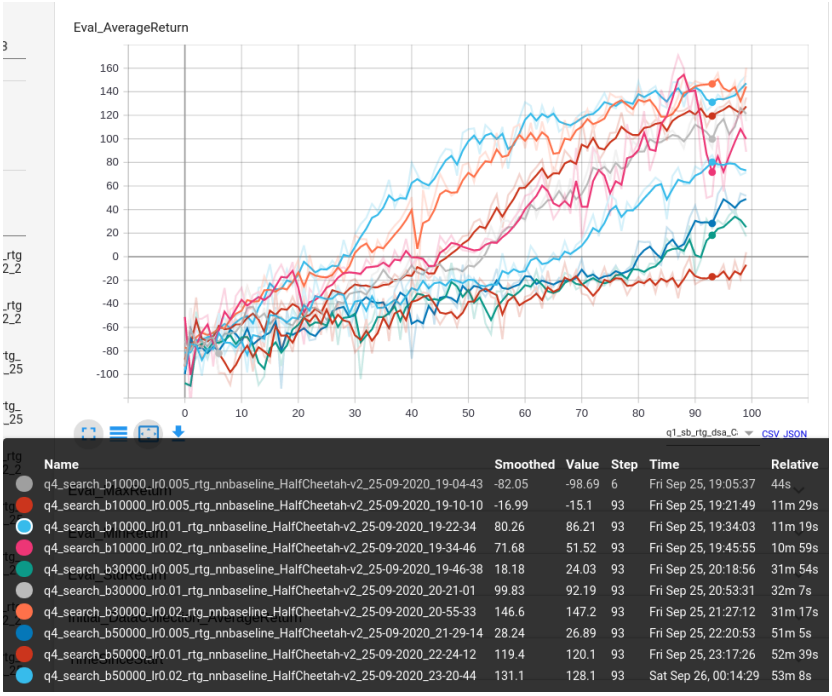


Figure 6: Lunar Landing section of cs285/scripts/run_experiments.ipynb

4 HalfCheetah



Name	Smoothed	Value	Step	Time	Relative
q4_search_b10000_lr0.005_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_19-04-43	-82.05	-98.69	6	Fri Sep 25, 19:05:37	44s
q4_search_b10000_lr0.005_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_19-10-10	-16.99	-15.1	93	Fri Sep 25, 19:21:49	11m 29s
q4_search_b10000_lr0.01_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_19-22-34	80.26	86.21	93	Fri Sep 25, 19:34:03	11m 19s
q4_search_b10000_lr0.02_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_19-34-46	71.68	51.52	93	Fri Sep 25, 19:45:55	10m 59s
q4_search_b30000_lr0.005_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_19-46-38	18.18	24.03	93	Fri Sep 25, 20:18:56	31m 54s
q4_search_b30000_lr0.01_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_20-21-01	99.83	92.19	93	Fri Sep 25, 20:53:31	32m 7s
q4_search_b30000_lr0.02_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_20-55-33	146.6	147.2	93	Fri Sep 25, 21:27:12	31m 17s
q4_search_b50000_lr0.005_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_21-29-14	28.24	26.89	93	Fri Sep 25, 22:20:53	51m 5s
q4_search_b50000_lr0.01_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_22-24-12	119.4	120.1	93	Fri Sep 25, 23:17:26	52m 39s
q4_search_b50000_lr0.02_rtg_nnbaseline_HalfCheetah-v2_25-09-2020_23-20-44	131.1	128.1	93	Sat Sep 26, 00:14:29	53m 8s

Figure 7: Cheeta finding batch size and learning rate

Provide a single plot with the learning curves for the HalfCheetah experiments that you tried. Describe in words how the batch size and learning rate affected task performance.

The learning rate of 0.02 was the most effective. At one point in the graph, the 3 highest values all have learning rate of 0.02. The lowest runs all had the smallest learning rate so there is definitely some correlation between learning rate and progress. Seems like the larger the learning rate the better the learner did. The same goes for batch sizes, the larger the better.

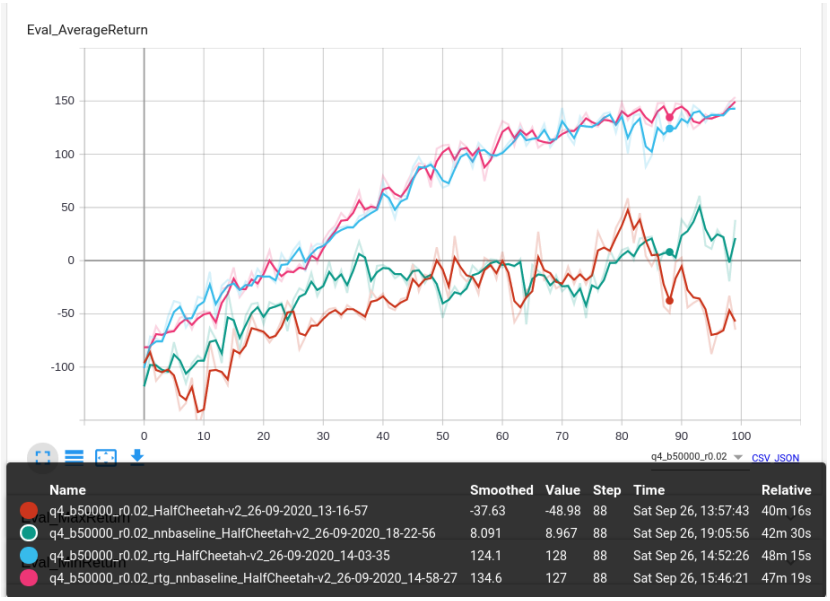


Figure 8: Cheeta Additional Experiments

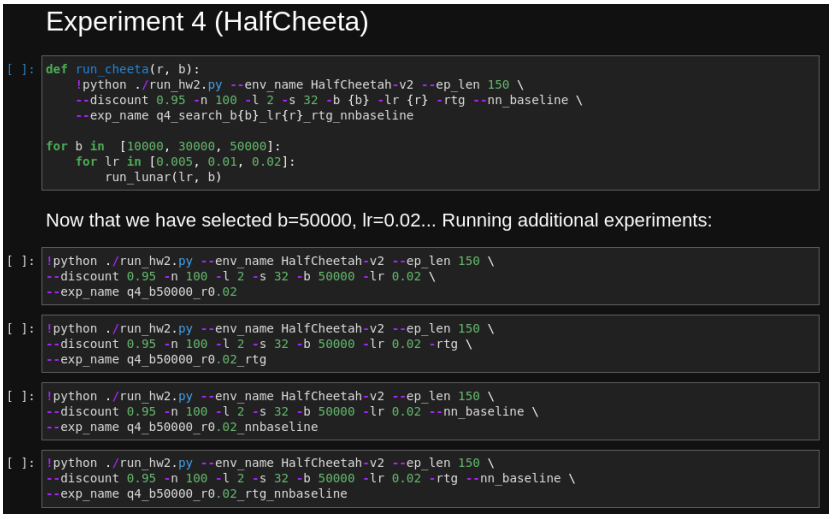


Figure 9: Cheeta section of cs285/scripts/run_experiments.ipynb