CS 294-112 Deep RL HW1

Nathan Lambert - nol@berkeley.edu

August 30, 2018

Introduction

This homework is a warmup back into Tensorflow with a implementation of behavioral cloining for imitation learning and the DAgger algorithm. All of my code for assignments is found here a ways after the due date: https://github.com/natolambert/deepRL-homework.

1 Set Up:

Instructions as completed.

2 Behavioral Cloning

2.1 Expert Policies

Half Cheetah	mean	4142.998
	std	84.641
Hopper	mean	3777.355
	std	3.106
Ant	mean	4831.173
	std	124.238
Humanoid	mean	10419.300
	std	52.192
Reacher	mean	-3.984
	std	1.835

Table 1: Expert Rollouts

2.2 Behavior Cloning Implementation

Behavior cloning from the expert policies simply entails training a model that maps from observations to actions. Then when rolling out the cloned policy, if

the observations are within the same range the new policy should perform well. Hyper parameters are the size of the network, parameters used, and amount of data etc.

To run my code for this part, enter into terminal the script name followed by the environment and the question part:

```
$ python behave_clone.py Reacher-v2 two
```

Param	Value
depth	3
activation function	ReLU
hidden nodes	200
epochs	200
learning rate	1E-6
batch size	50

Table 2: Network Parameters Table

Half Cheetah	mean	
	std	
Hopper	mean	177.872
	std	33.400
Ant	mean	
	std	
Humanoid	mean	177.772
	std	40.892
Reacher	mean	-9.760
	std	4.0317

Table 3: Behavioral Cloned Rollouts. The training data was the 20 rollouts from *run_expert.py*, with a test/train split of .8. The rollouts can generate different numbers of datapoints

2.3 Hyperparameters

I swept the hidden layer width to see where the network gains the representation capcity for the imitated policy.

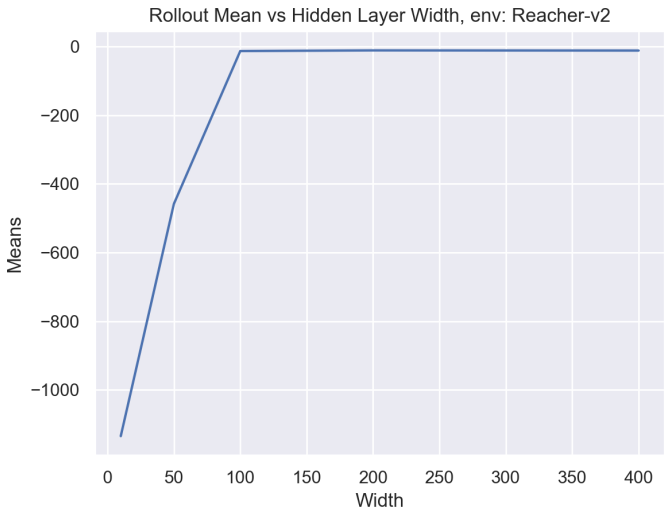

Figure 1: The behavioral clone return verses hidden layer width. All networks had 3 fully connected layers, were trained for 200 epochs with Adam at a learning rate of 1E-6, with a batch size of 60.

3 DAgger

3.1 Implement DAgger

3.2 DAgger Results

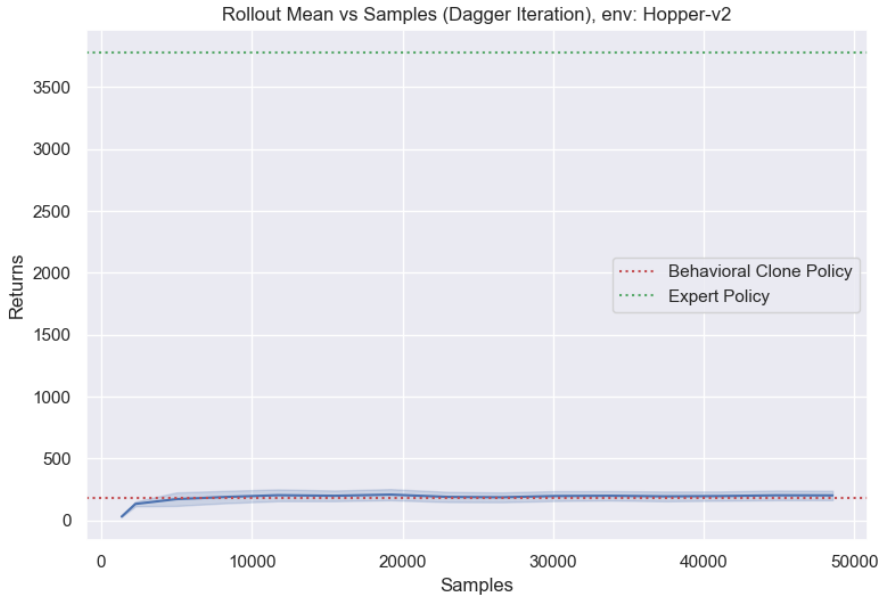

Figure 2: The DAgger implementation surpassing behavioral cloning for the Hopper-v2 task. All networks had 3 fully connected layers of 200 nodes, were trained for 200 epochs with Adam at a learning rate of 1E-6, with a batch size of 60. Note, each samples number is constant for each DAgger iteration. I used 15 iterations.

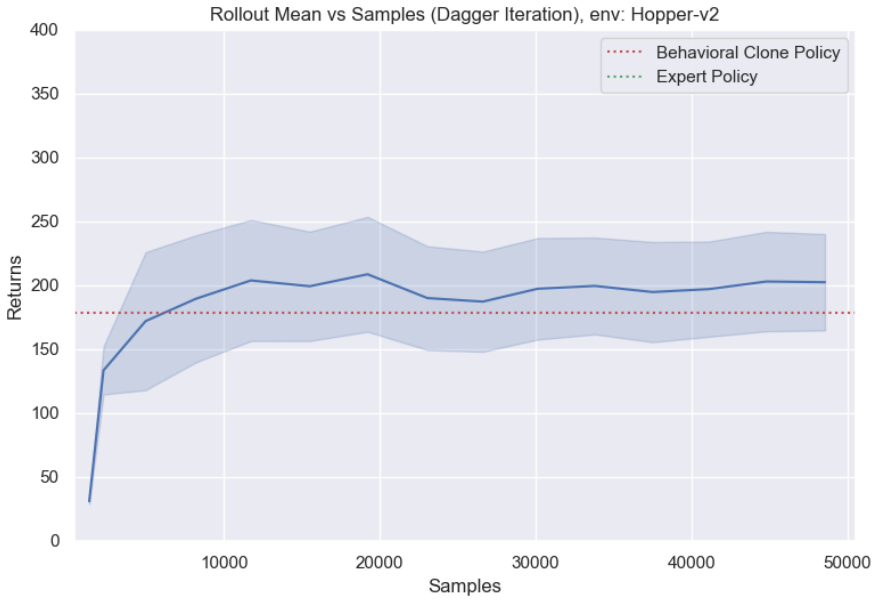

Figure 3: A zoomed version of the above plot on the DAgger values verses the behavioral clone policy.

4 Bonus: Alternative Policy Architectures

5 Discussion

References