Assignment 1: Imitation Learning

Thanapat Trachu

January 1^{st} 2023

Abstract

The goal of this assignment is to experiment with imitation learning, including direct behavior cloning and the DAgger algorithm. In lieu of a human demonstrator, demonstrator, demonstrations will be provided via an expert policy that we have trained for you. Your goals will be to set up behavior cloning and DAgger, and compare their performance on a few different continous control tasks from the OpenAI Gym benchmark suite.

1 Behavior Cloning

This experiment chooses Ant-v2 to demonstrate the agent which perform at least 30% of the expert agent, and Humanoid-v2 to demonstrate the agent which is not.

Question 1.2: Mean and Std of policy's return

This table provide an information about performance of trained policy and expert policy, on Ant-v2 and Humanoid-v2 environment.

	Ant-v2	**Humanoid-v2**
Expert	Mean: 4679.166, Std: 0	Mean: 10714.461, Std: 0
Trained	Mean: 1739.392, Std: 1035.577	Mean: 282.971, Std: 66.83

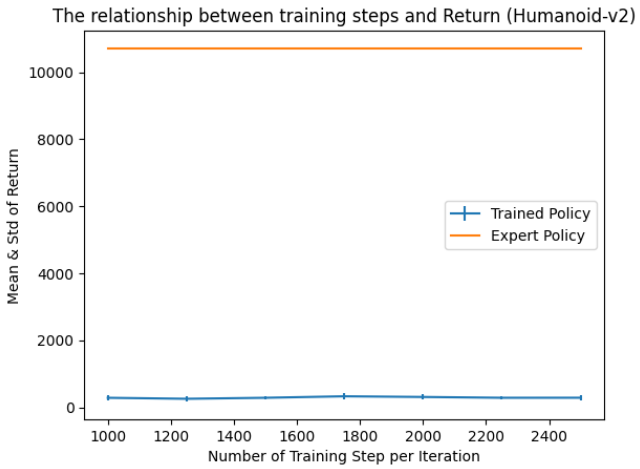
Both environments use the same parameters (default) in term of network size, amount of data, number of training iterations, and etc.

Parameters	**Value**
num_agent_train_steps_per_iter	1000
batch_size	1000
train_batch_size	100
ep_len	1000
eval_batch_size	5000
n_layers	2
size	64

Question 1.3: Experiments with hyperparameters

Since the model with large number of training step was updated by the expert data more than the small number of training step. Therefore, as the the number of training step increased, the performance of the model should also increase.

(a) Ant Environment



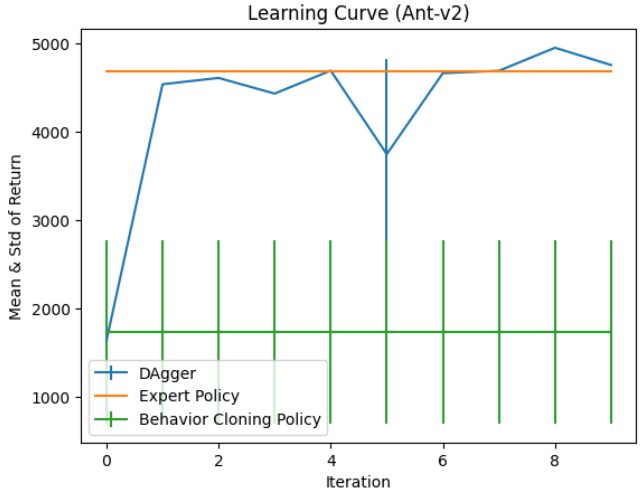
(b) Humanoid Environment

2 DAgger

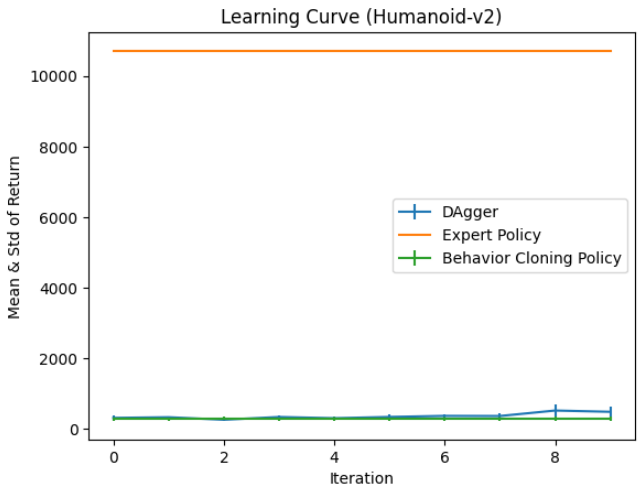
The parameters which are used in this sections is the same as in the previous sections. The only difference is n_iter. The number of iteration is set to 10 for DAgger framework.

Question 2.2: Learning Curve of DAgger

The following figures show the performance of the model in each DAgger iteration.



(a) Ant Environment



(b) Humanoid Environment

Appendix

MLP Policy

The MLP Policy uses reparameterize trick, in order to make the model to be able to find gradient.

$$sampling\_value = mean + std * random\_value$$