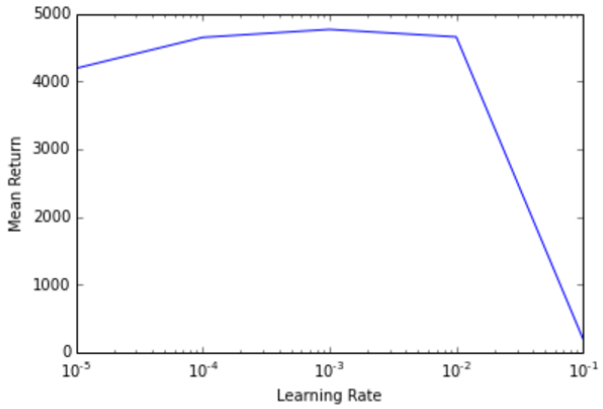
Yifat Amir
HW1
Deep RL

| Task | My Mean of Returns | My SD of Returns | Expert Mean of Returns | Expert SD of Returns |
|---|---|---|---|---|
| Ant | 4852.50172309 | 74.2398030029 | 4782.16091724 | 132.555603119 |
| Walker2d | 4995.4266902 | 1242.65553931 | 5534.92655192 | 44.5002486121 |
| Hopper | 1722.43819117 | 1216.78097727 | 3777.9972139 | 3.20571253539 |
| Reacher | -4.86192835166 | 1.69605609577 | -3.89877452701 | 1.81825232762 |
| Humanoid | 10385.5954084 | 52.9418006081 | 10411.3670062 | 64.3865538357 |
| HalfCheetah | 4049.21748582 | 92.0304040342 | 4158.39699685 | 46.9716858278 |

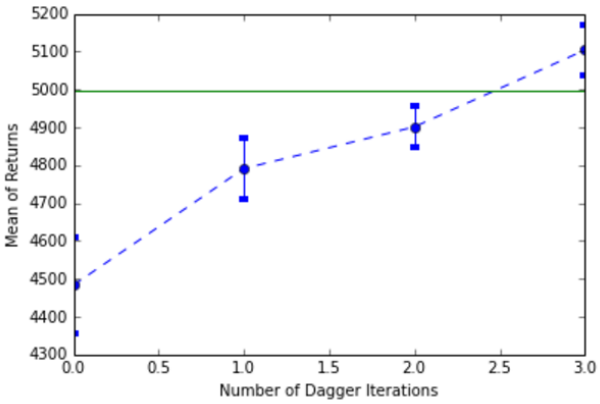
All tasks were run on a NN with two hidden layers, tanh activation, 20000 sample points, 20000 training iterations, batch size of 100, and learning rate of 0.001. Most tasks achieve comparable performance, except Hopper.




```
print creature
```

```
Ant
```

The graph above shows how performance varies with the value of the learning rate. I chose to experiment with this hyperparameter because it can greatly affect the optimization of the loss function. If the learning rate is too high, it is easy to overshoot a local optimum in the cost function. If the learning rate is too low, then it is hard to learn anything given a fixed number of training iterations.




```
print creature
```

```
Walker2d
```

The task above was run on a NN with two hidden layers, tanh activation, 20000 sample points, 10000 training iterations, batch size of 100, and learning rate of 0.001. There were 4 DAgger iterations.