2 Imitation Learning

2.5 Testing

Using the following values of the parameters, where all the others take on the default values given in the code, we obtain the following average rewards:

1. `python main.py --env="CartPole-v0" --bc_epochs=100 --num_dataset_samples=100`
    - Average reward: 70.4
2. `python main.py --env="CartPole-v0" --dagger --num_dataset_samples=100 --num_rollout_steps=100 --dagger_epochs=10 --dagger_supervision_steps=10`
    - Average reward: 147.1
3. `python main.py --env="MountainCar-v0" --bc_epochs=200 --num_dataset_samples=10000`
    - Average reward: -84.4
4. `python main.py --env="MountainCar-v0" --dagger --num_dataset_samples=1000 --num_rollout_steps=100 --dagger_epochs=10 --dagger_supervision_steps=200`
    - Average reward: -85.8

2.6 Exploration

Below, we tabulate the average rewards that we obtain for BC and DAGGER on each of the possible environments, where we run both algorithms with the same initial dataset. Each table corresponds to a different dataset size used when learning, as determined by the value of the `num_dataset_samples` argument. We also use the following values:

```
1. bc_epochs = 100
2. num_rollout_steps = 100
3. dagger_epochs = 10
4. dagger_supervision_steps = 10
```

Thus, we make it so that BC and DAGGER both learn for the same number of epochs in total, namely, 100. All other arguments take on the default values given in the code.

`num_dataset_samples = 5000`

	CartPole	MountainCar
BC	200	-83.2
DAGGER	200	-82.9

`num_dataset_samples = 1000`

	CartPole	MountainCar
BC	200	-83.7
DAGGER	200	-82.4

`num_dataset_samples = 200`

	CartPole	MountainCar
BC	198.5	-139.0
DAGGER	199.3	-122.5

`num_dataset_samples = 150`

	CartPole	MountainCar
BC	129.9	-159.6
DAGGER	159.2	-140.0

`num_dataset_samples = 100`

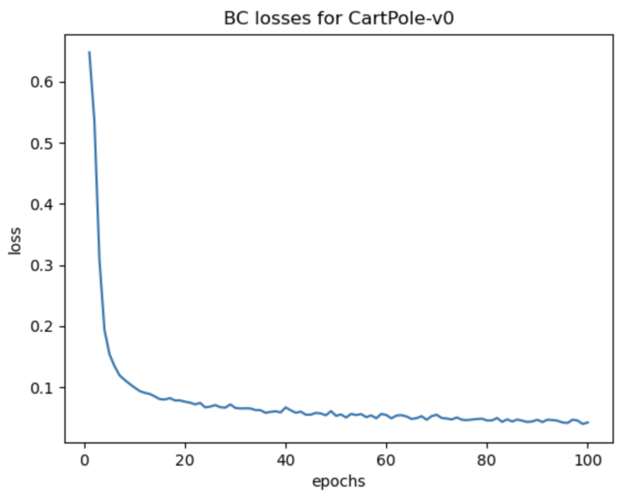
	CartPole	MountainCar
BC	70.7	-200
DAGGER	83.2	-200

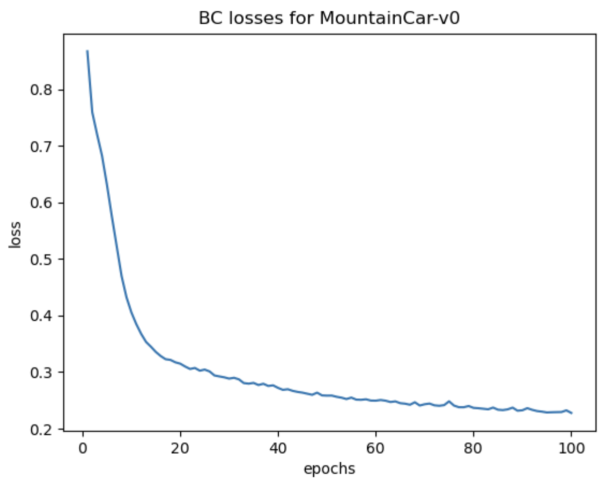
- How does BC perform with a smaller dataset? Is this expected?
  We can see from the data above that BC performs worse with smaller datasets, in the sense that it receives lower rewards with decreasing dataset size. The rewards obtained in CartPole represent how long the agent is able to maintain the pole upright, receiving a reward of +1 for every time step, and so higher rewards correspond to better outcomes, in the sense that the pole is upright for longer, as desired. In MountainCar, the goal is to get the car up the hill as fast as possible, so the negative reward of -1 incurred at every time step is supposed to incentivize shorter duration to crest the hill.
  It is indeed expected that BC gets decreasing rewards with smaller datasets because with smaller datasets, we reduce the coverage of the state space for which we have information on how the expert acts, and whenever the agent arrives at a state for which there is no known information on the expert actions, the agent

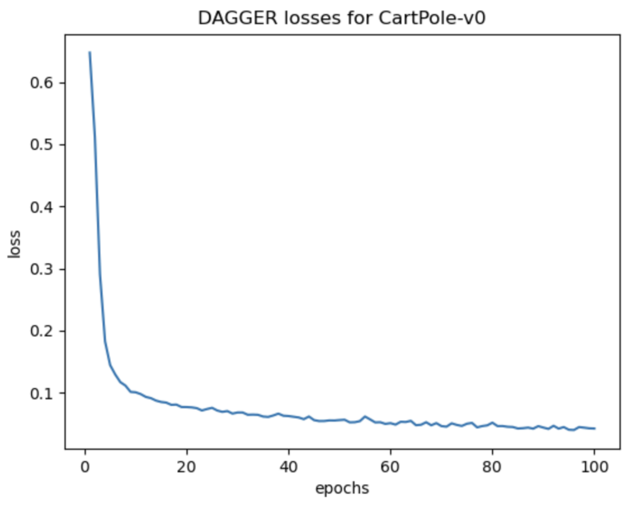
can take any action at random. Therefore, the smaller the dataset size, the larger the number of states for which there will be no expert knowledge, and thus the greater the probability that the agent will act at random and hence not take actions that are optimal or close to optimal.

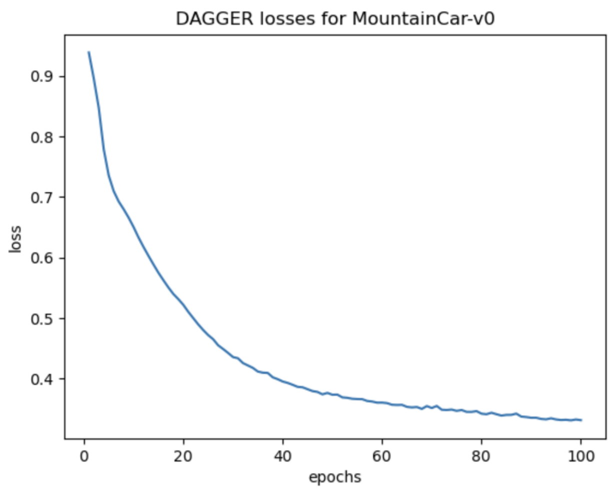
- How do BC and DAGGER perform with the same starting dataset? How quickly does each converge loss-wise?
  From the data above, we can see that DAGGER performs generally better than BC when we start both of them on the same dataset. Specifically, for large dataset size, both of them perform similarly, leading to large positive rewards in CartPole and large, negative rewards in MountainCar. Thus, it is difficult to compare their performance with these large dataset sizes. For this reason, we tried different values of the dataset size, and we see that with decreasing dataset size, DAGGER performs better than BC for both environments. In general, this makes sense since, starting with the same dataset, BC only learns from that specific dataset, whereas DAGGER iteratively rolls out new data and can therefore query the expert policy at more states, compared to BC, and so can adapt better to new circumstances, i.e. it is more robust.
  Finally, we can see from the plots below that both DAGGER and BC converge very similarly, in general, in terms of loss. It is hard to single out a clear, distinctive difference in their convergence that applies to all different dataset sizes, but, broadly, we can say that the convergence speed of BC is usually faster than DAGGER because BC only relies on the provided expert demonstrations, without having to roll out new data iteratively. However, BC seems to plateau or struggle in some cases, which may correspond to scenarios in which the expert knowledge does not cover the states being explored. DAGGER, on the other hand, tends to improve over time as it rolls out more data and refines the policy accordingly, which makes it generally more robust.

`num_dataset_samples = 1000`

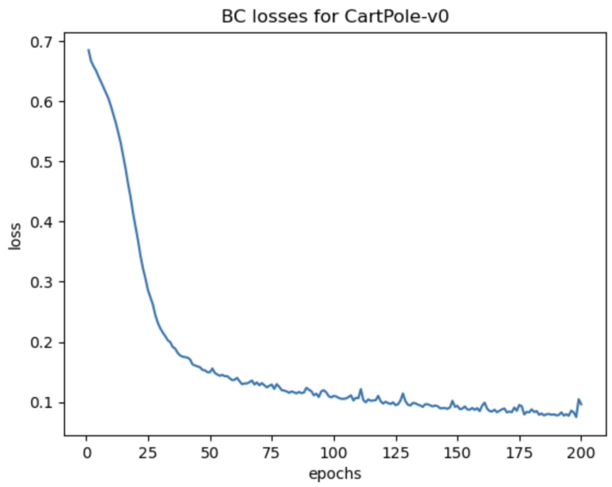


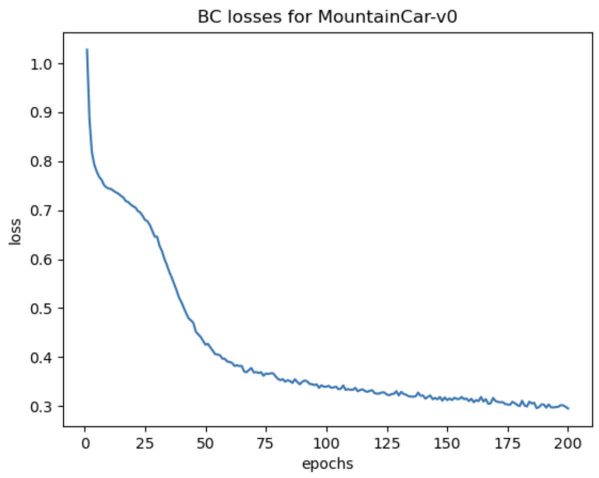


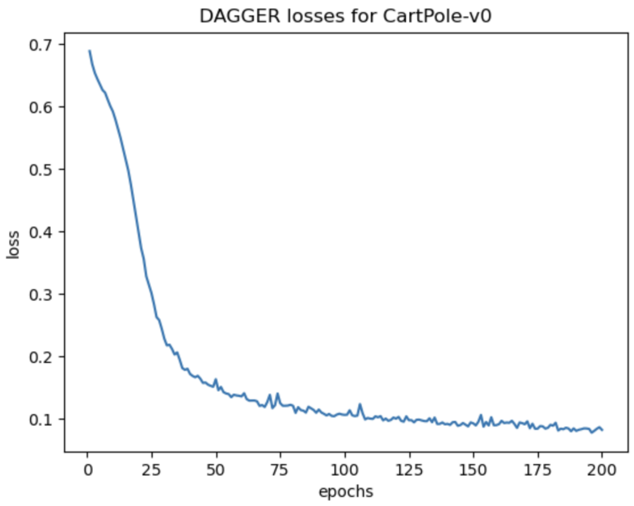


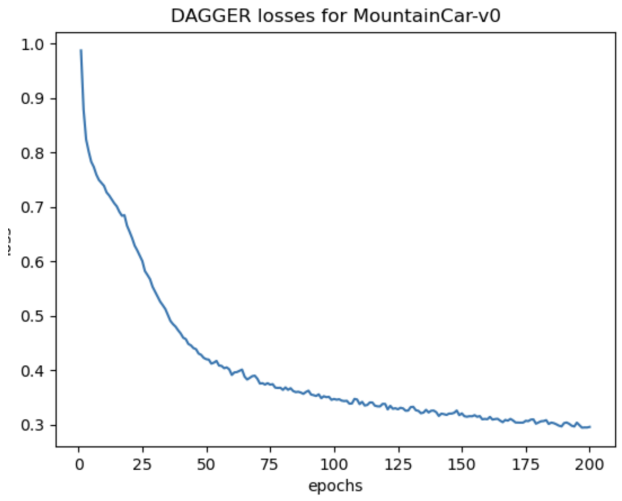


`num_dataset_samples = 100`

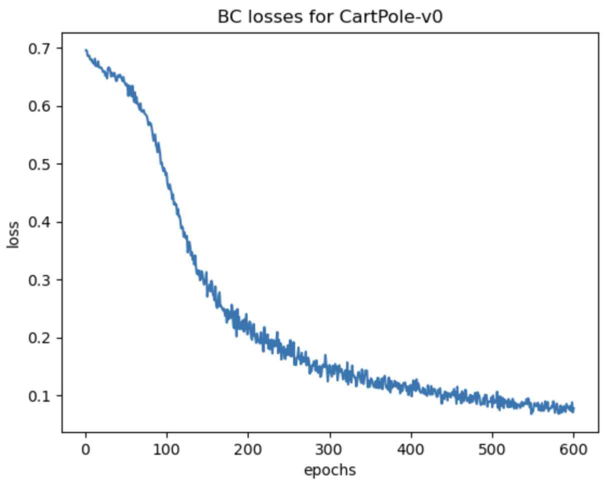


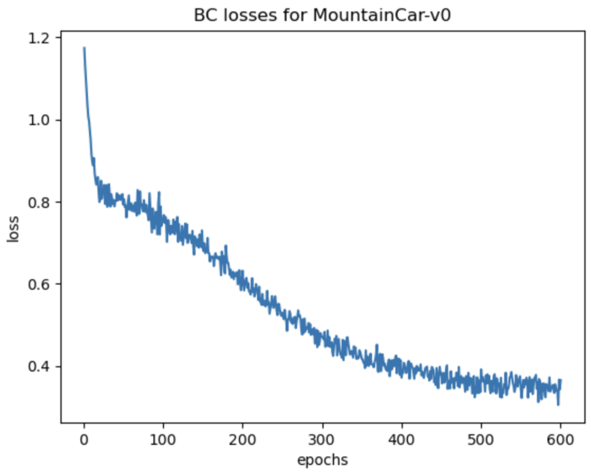


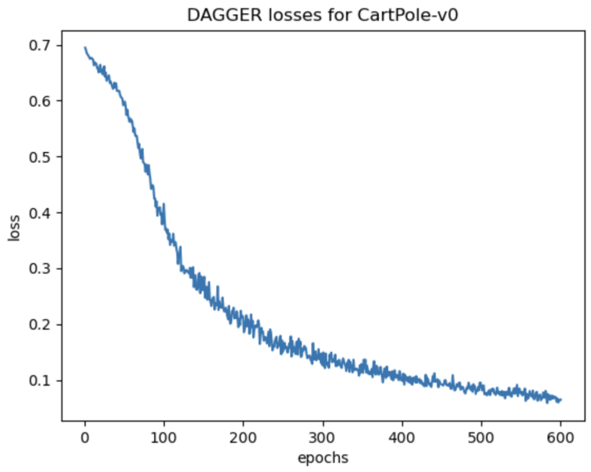


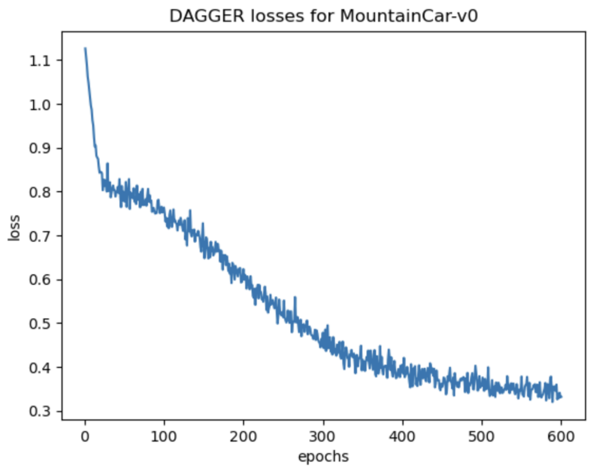


`num_dataset_samples = 50`


BC losses for CartPole-v0
loss
epochs


BC losses for MountainCar-v0
loss
epochs


DAGGER losses for CartPole-v0
loss
epochs


DAGGER losses for MountainCar-v0
loss
epochs