HW3_gagne

Chris Gagne SSID=25952284

October 4, 2017

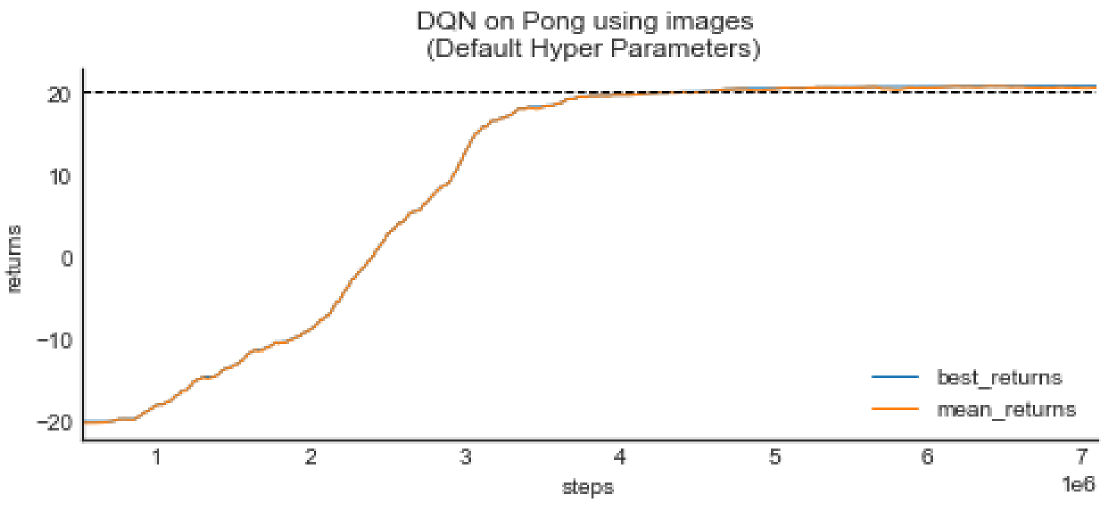
1 Question 1: basic Q-learning performance.

1.0.1 Implentation:

- The code for running the DQN can by found in dqn.py
- The result was obtained on AWS on a g2.2xlarge EC2 instances type.
- I used Huber loss instead of mean square error.
- I also used tf.stop_gradient for the target network, though I don't think this was necessary.

1.0.2 Results:

- My average return did not increase as fast as the reference solution by 1m steps, but that could have been my random seed. However, it did get to optimal performance by 3.5m.
- I'm not sure why my best and mean returns are so close. They are not identical.



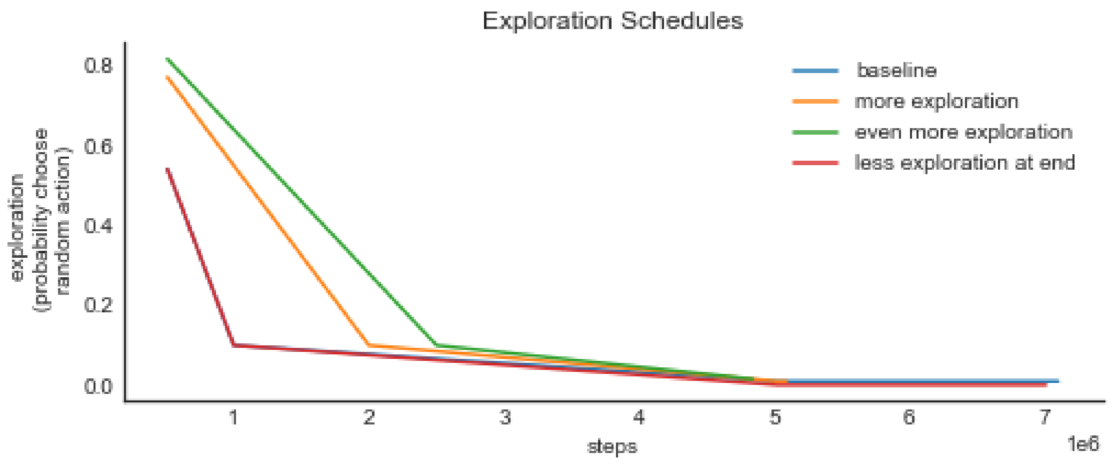
2 Question 2: experimenting with hyperparameters.

2.0.1 Exploration

For this question, I experimented with differnent exploration schedules and compared the performance to the default schedule. I also ran my DQN algorithm on the Pong with full images. Each round until convergence took ~10+ hrs.

2.0.2 Exploration Schedules

- baseline goes to 0.1 after 1m steps, then down to 0.01 after 5m
- more exploration goes to 0.1 after 2m steps, then down to 0.01 after 5m
- even more exploration goes to 0.1 after 2.5m steps, then down to 0.01 after 5m
- less exploration at end goes to 0.1 after 1m steps like baseline, but goes down to 0.001 after 5m steps.

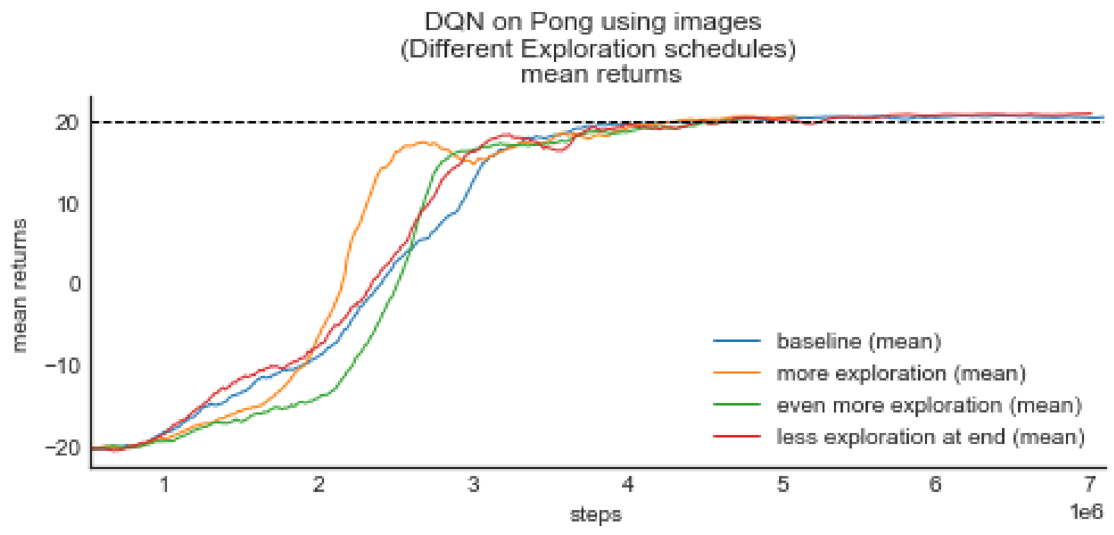


2.0.3 Results:

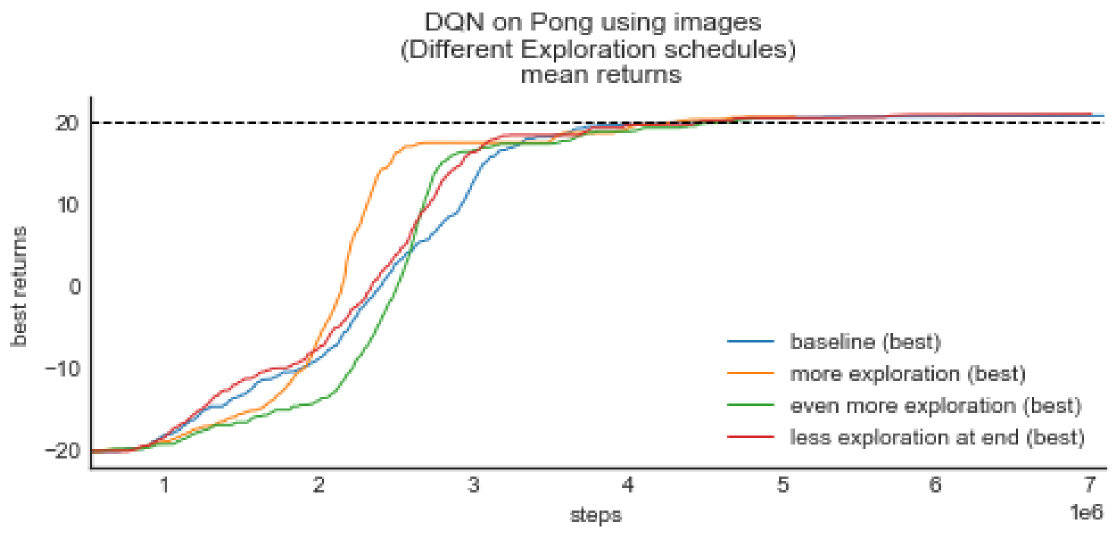
- Reducing the exploration to 0.001 from 0.01 at after the algorithm had converged had almost no effect on the mean or best returns (red v blue). In hindsight, this was a dumb schedule to try. But, at least it gives me a sense of the variability in returns for the default exploration schedule.
- increasing the exploration so that it continued to explore until 2m steps (orange line), yielded returns that were worse until the 2mth step, but then gained ground more quickly after that. Interestingly, although this schedule reached 'good' performance faster than the default schedule, it achieved asymptotic performance around the same time.
- finally, increasing the exploration even more so that it continued to explore until 2.5m steps yielded returns that were worse until the 2.5mth step, and then faster gains after that. However, by 2.5m steps, the default had already achieved good performance, so there was not much more room to improve.

- In all, this indicates that increasing exploration early could lead a quicker acension to 'good' performance, but the fine tuning required for perfect performance seemed to require similar amounts of subsequent training after the exploratory period.

```
<matplotlib.figure.Figure at 0x10243cdd8>
```



```
<matplotlib.figure.Figure at 0x119e93c88>
```


DQN on Pong using images
(Different Exploration schedules)
mean returns
best returns
steps
1e6
baseline (best)
more exploration (best)
even more exploration (best)
less exploration at end (best)