

**Details:** Oz Granit, ID 203202684, [ozgranit@gmail.com](mailto:ozgranit@gmail.com)

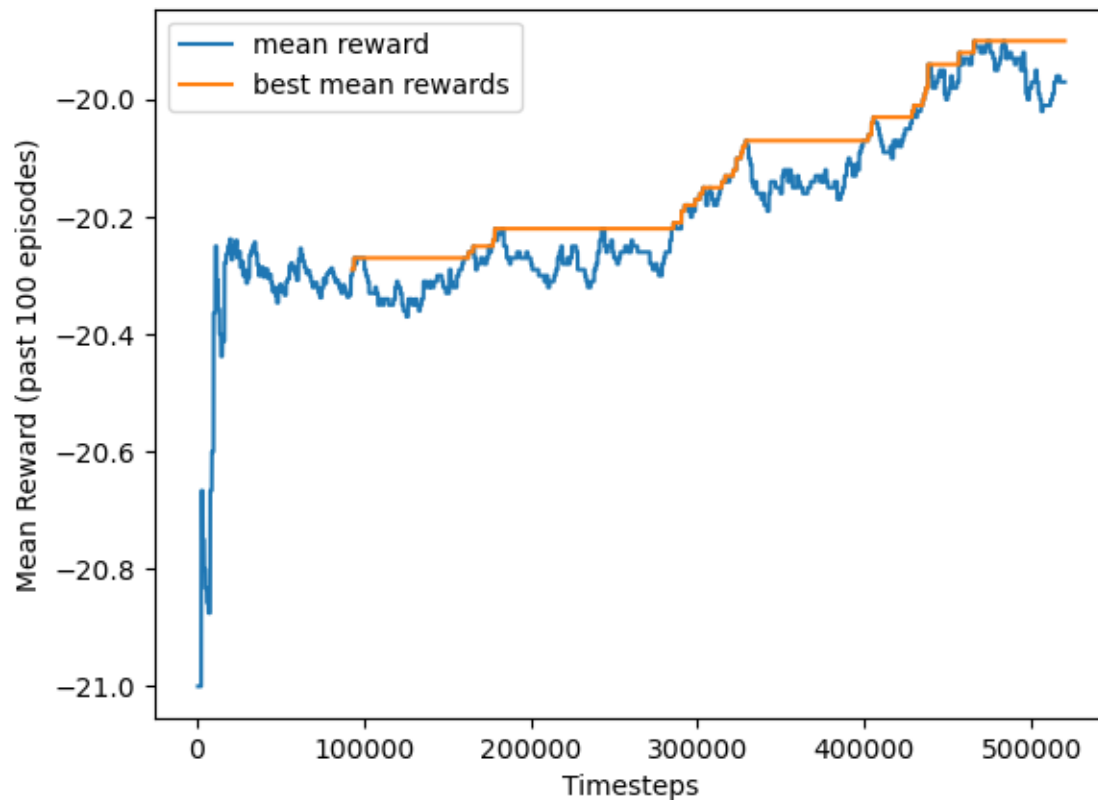
## **Training DQN on Atari-Pong:**

### **First Test:**

Started out by using the default parameters and comparing them to the reference solution in the instruction file, meaning I expected a reward of around -20 to -15 after 500k steps.

The best mean reward after 500000 steps was -19.900, which is low compared to the expected result, so I chose to experiment with the Hyperparameters before training the model for 4M steps.

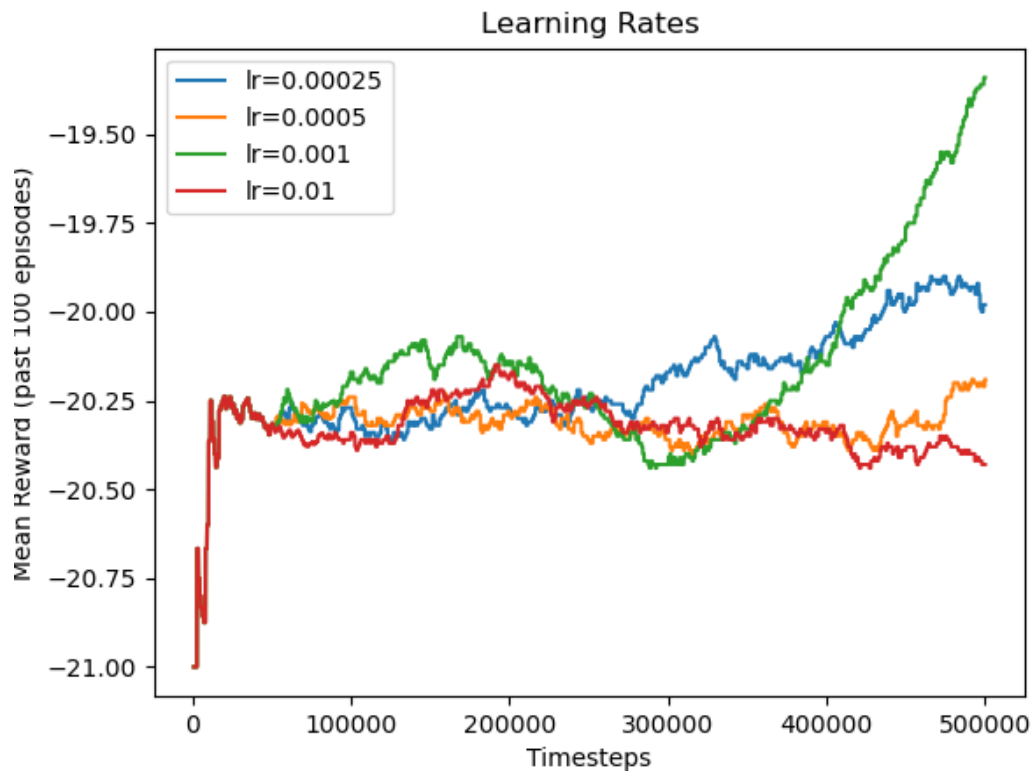
First Results for default parameters:



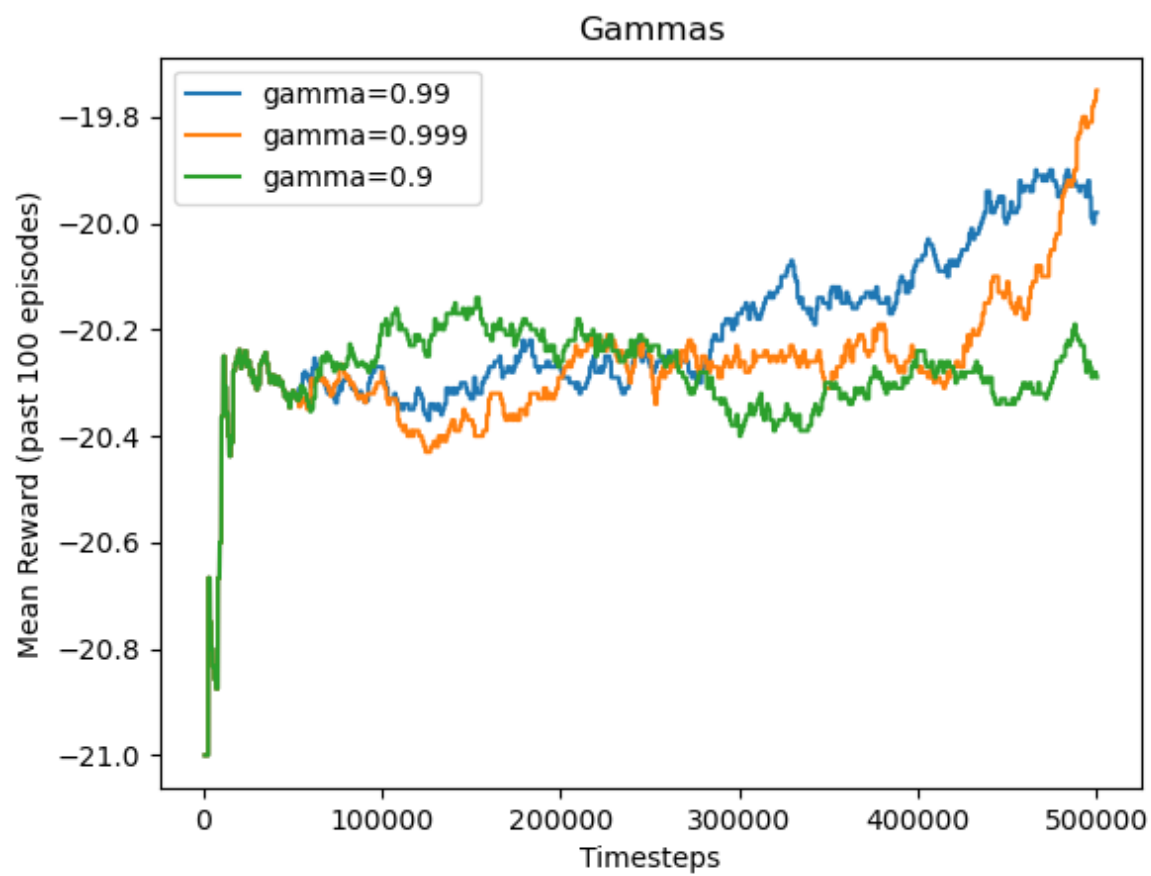
## Testing Hyperparameters:

I Experimented on different gammas, learning rates and exploration schedules,

Tested each parameter for 500,000 steps (due to time limitations, my GPU does not run Cuda). Assuming the best hyperparameters after 0.5M steps will remain best after 4M is too strong of an assumption. however, assuming the lower half of hyperparameters after a short run will not become the best after a long run seems more reasonable, so I continued testing only for the top half.

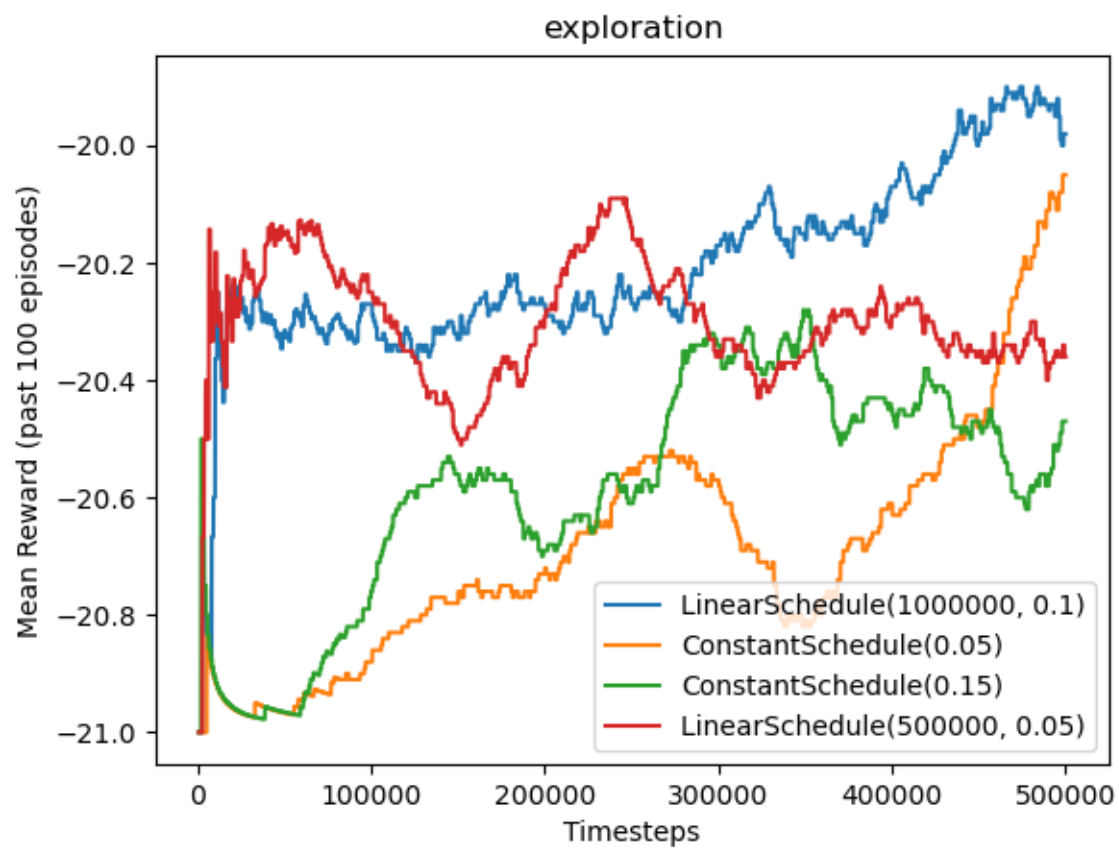


lr=0.00025 is the default, so I use lr=0.001 for further testing.



gamma results were far less conclusive, hardly seemed to have any effect.

Nonetheless, I used gamma=0.999 for further testing, 0.99 being the default value.

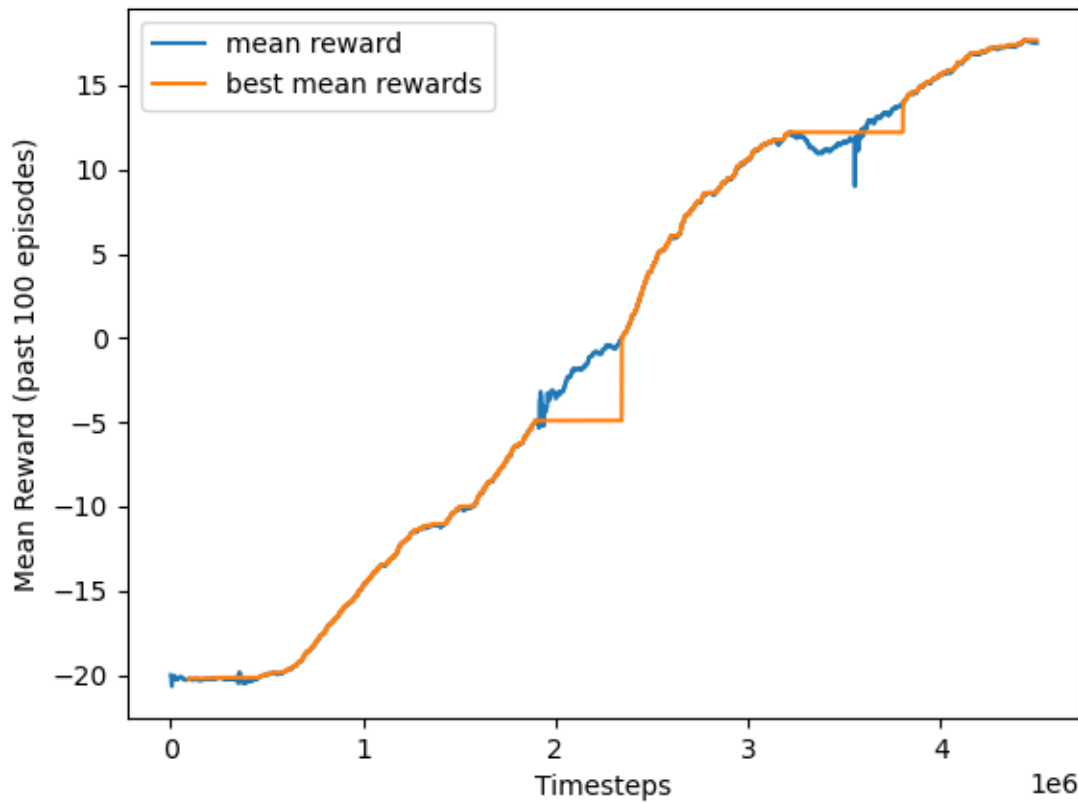


For exploration schedule a longer run was required, 0.5M steps were not enough to see a clear drift.

After those tests I started using a **randomized seed**.

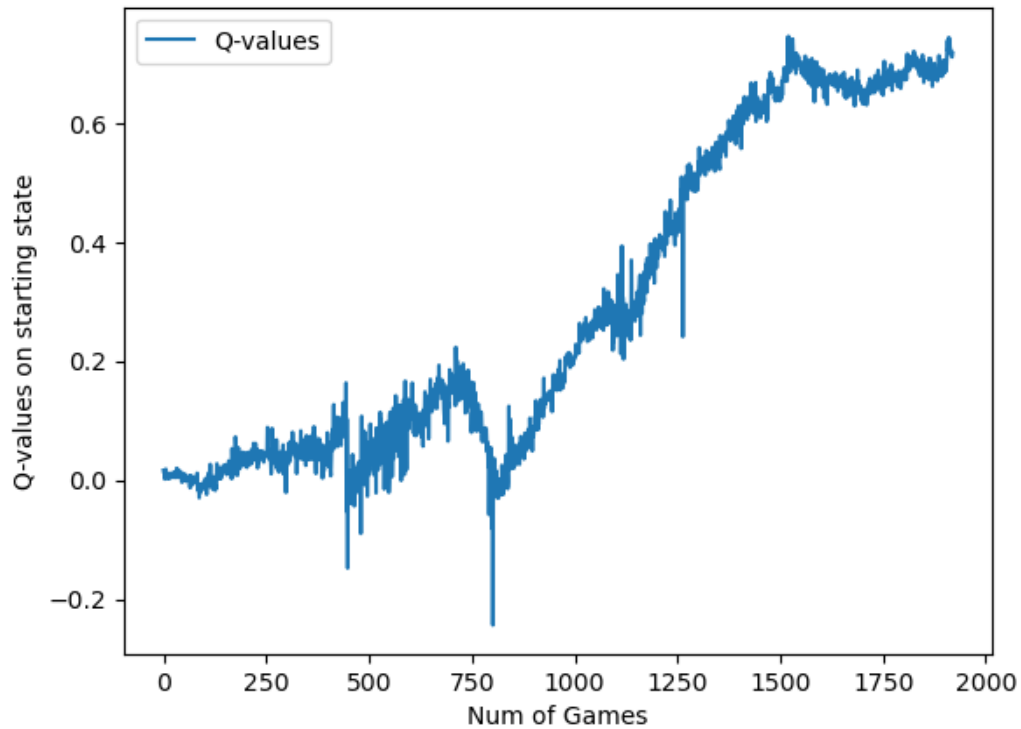
## Comparing Default and Chosen Parameters:

First run of Default Parameters, unfortunately the statistics file was corrupted on save, so I had to repeat the run (could not plot together with chosen parameters). Still, it adds information, so I decided to include the results.



The straight lines in 'best mean reward' are due to a collapse of the process (not enough RAM).

Plotted the Q-values of the initial state representation, to have a better idea of the DQN's estimate of how well it is going to do.



Overall results were decent:

```
Timestep 4550000  
mean reward (100 episodes) 17.460000  
best mean reward 17.680000  
episodes 456  
exploration 0.100000
```



default\_param\_4.5M  
mp4.

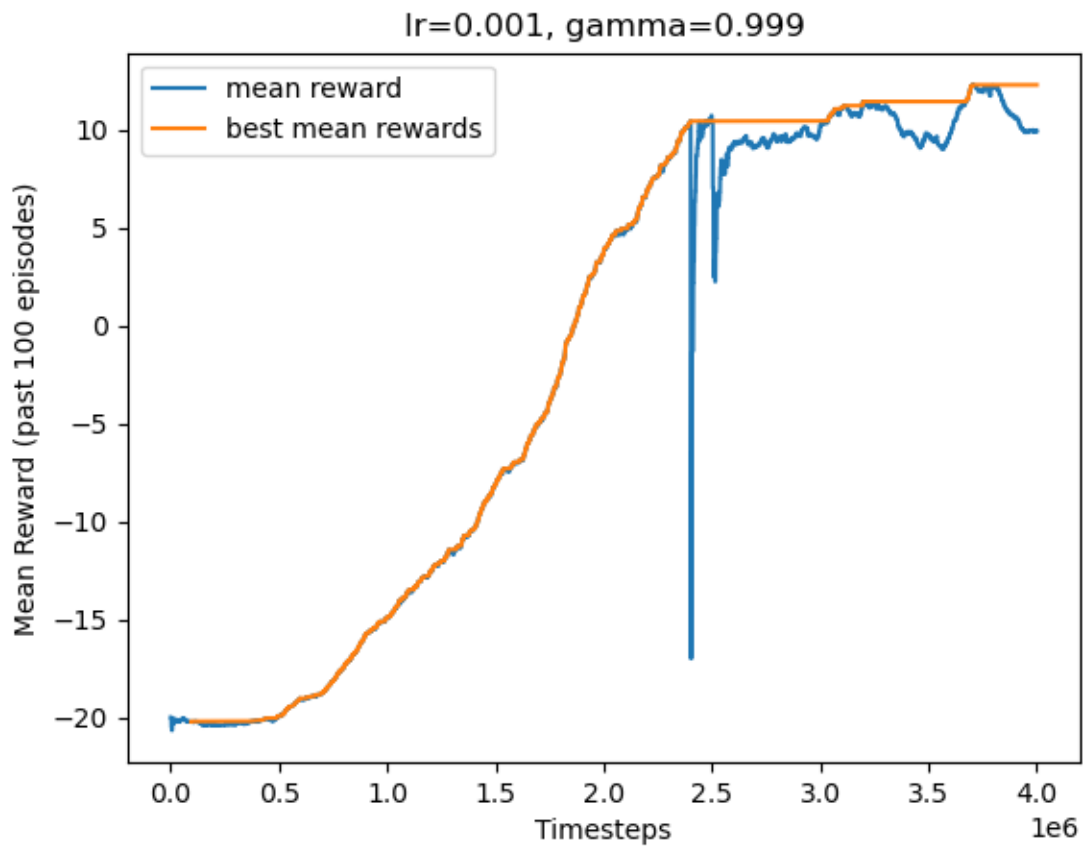


Q\_params.pkl

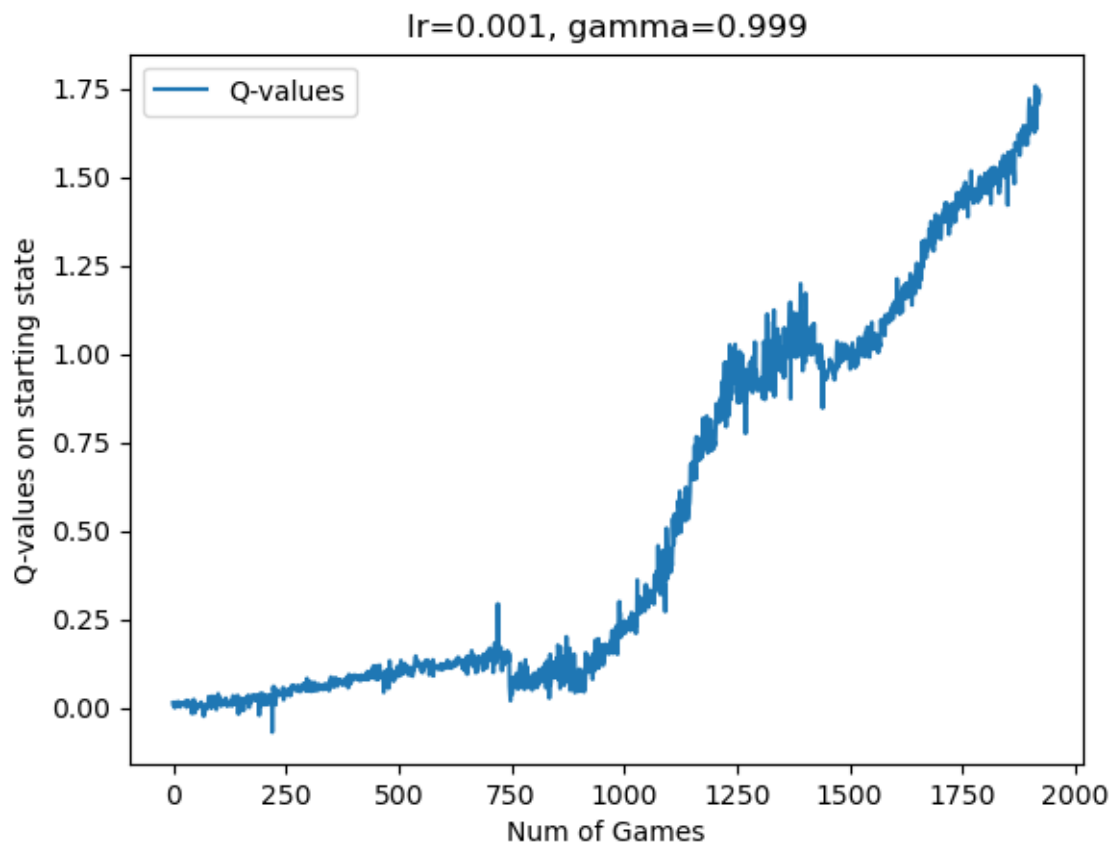


target\_Q\_params.pkl

For the second run I started with the chosen parameters.






All major drops in 'mean reward' came directly after a collapse and reboot of the process, so I assumed (wrongfully) there was a problem with the rebooting procedure or saved Q-parameters. so, I repeated this run as well.



A higher LR and gamma obviously yields higher Q-values.

Overall results were lower, however, keep in mind both runs used different seeds, so we can expect a varied result.

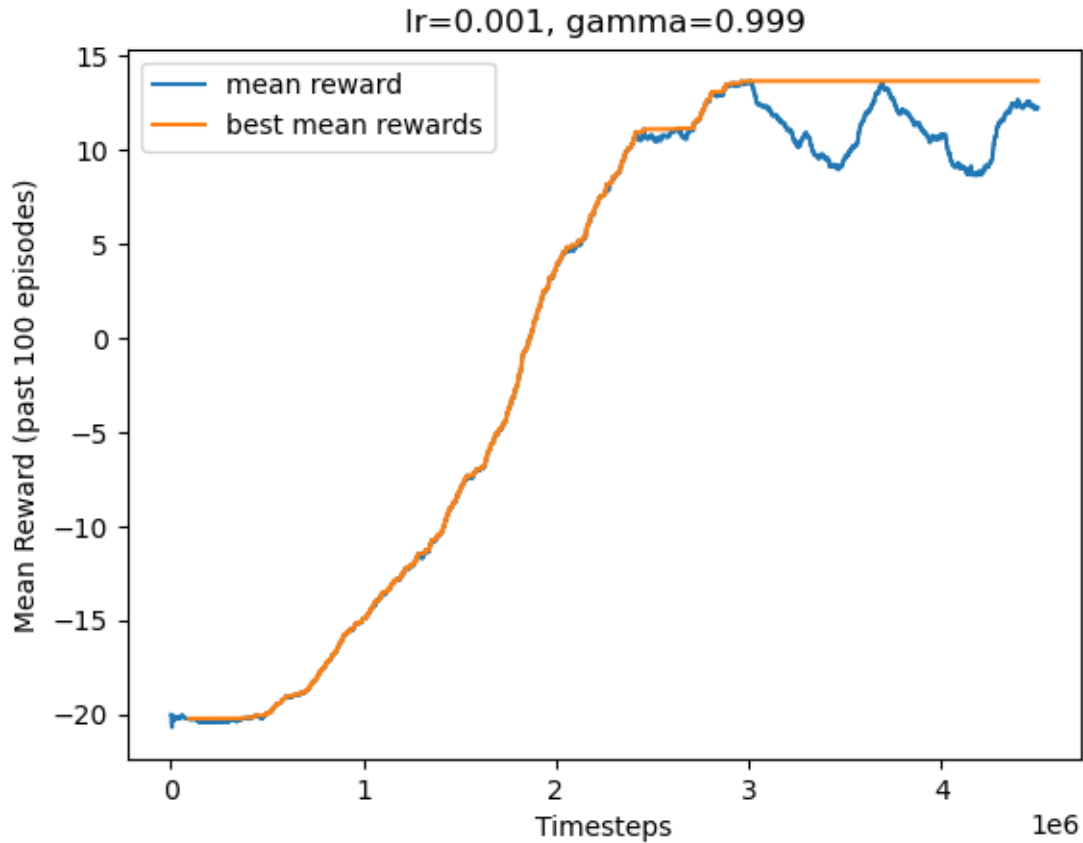
‘Best mean reward’ finished at ~13.5.

    
openai-gym.video.2.6 ,Q\_paramslr=0.001 =target\_Q\_paramslr  
video000512.mp4.51 gamma=0.999.pkl gamma=0.999.pkl ,0

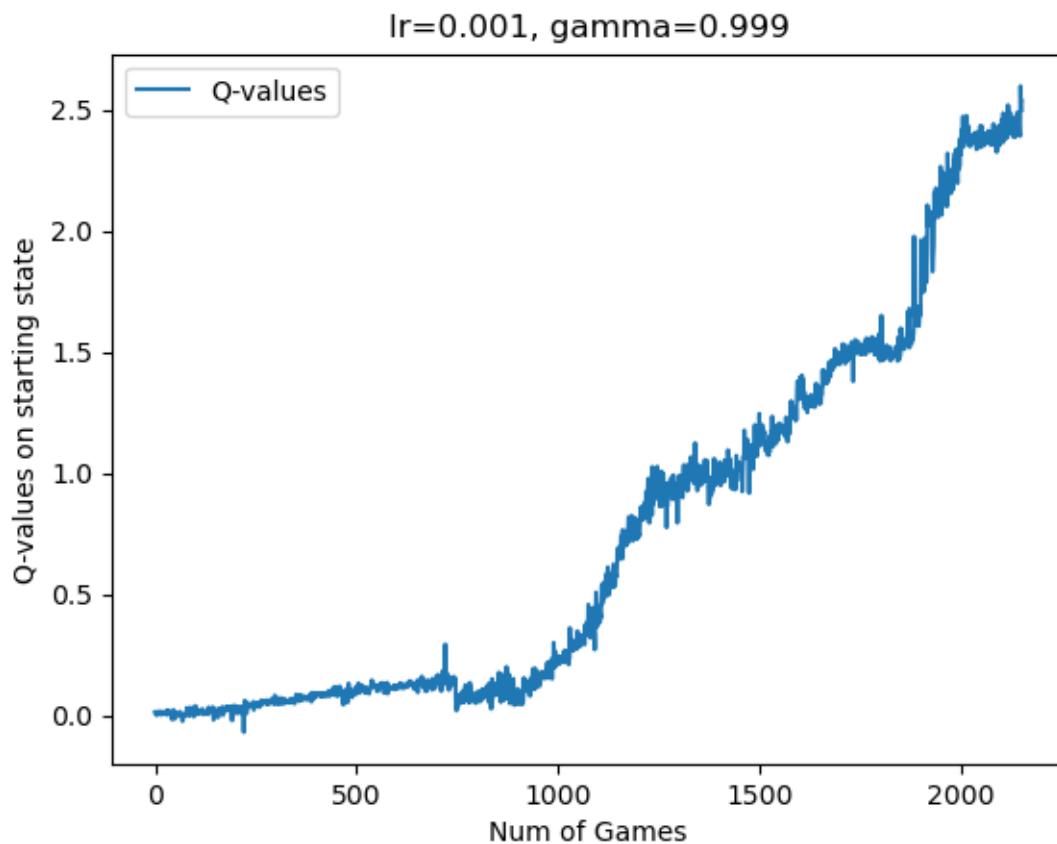


## Re-Comparing Default and Chosen Parameters:

Repeated Run for Chosen Parameters.





Received slightly better performance, also learned the noise in reward was a feature of the high Learning-Rate DQN, not a bug in saving/loading code.



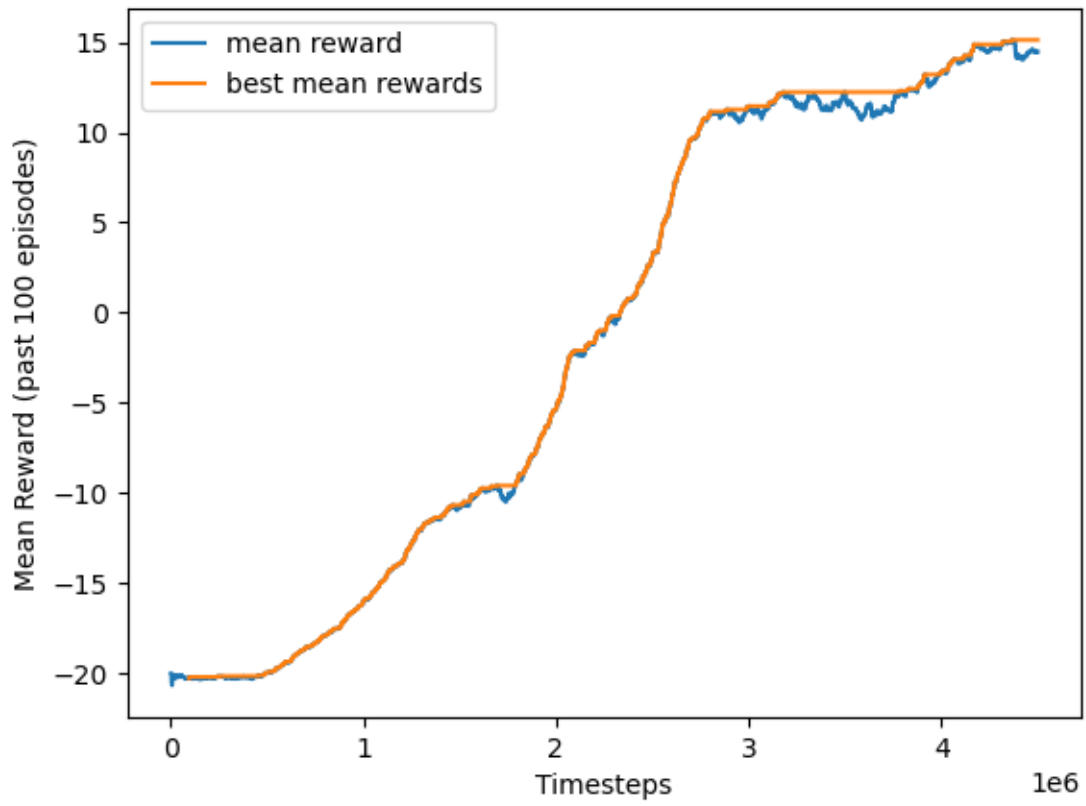
Notice Q-values here are high even for this learning rate and gamma, and continuously climbing, suggesting training for 0.5M-1M more steps could have yielded further improvement.

Overall results:

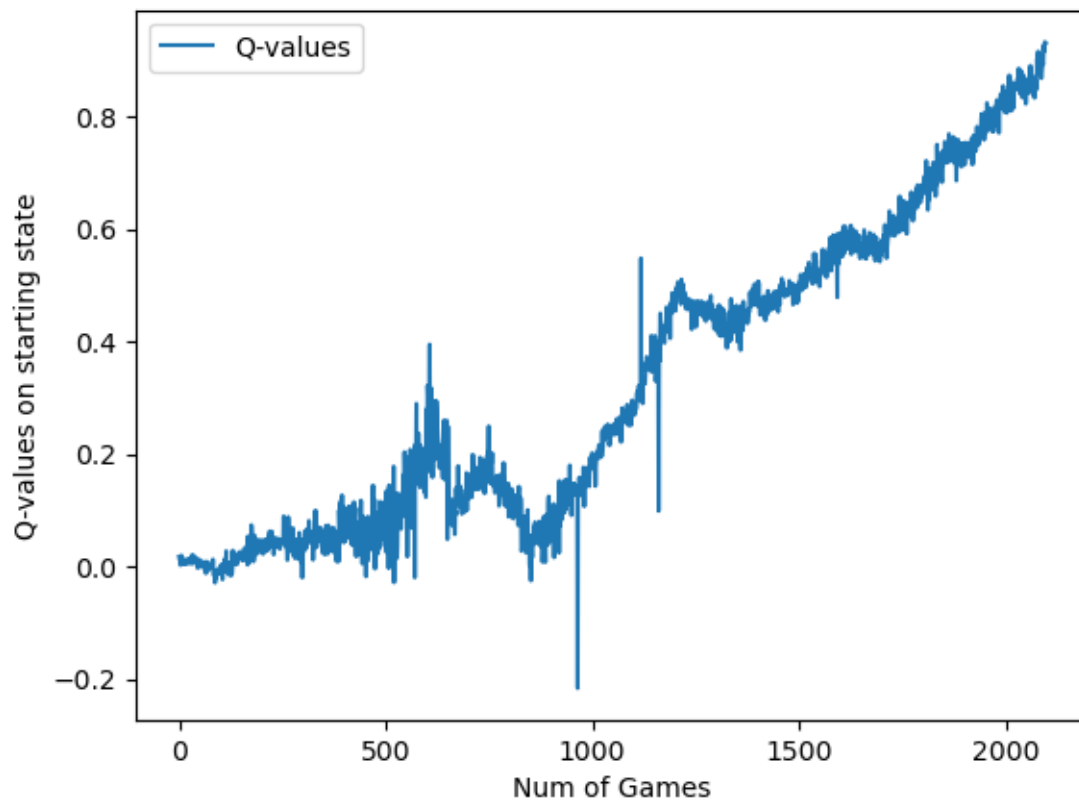
```
Timestep 4500000
mean reward (100 episodes) 12.240000
best mean reward 13.670000
episodes 2204
exploration 0.100000
Saved to copy_statistics lr=0.001, gamma=0.999.pkl
```

   
,Q\_paramslr=0.001 =target\_Q\_paramslr  
gamma=0.999.pkl gamma=0.999.pkl ,0

Repeated Run for Default Parameters.



When the learning rate is small, we see less noise on 'mean reward' measurement.




Naturally, smaller learning-rate and gamma yields smaller Q-values. The increase trend near the end of the run suggests the DQN was still far from converging, plotting the loss function could have given us a better indication of convergence.

Overall results:

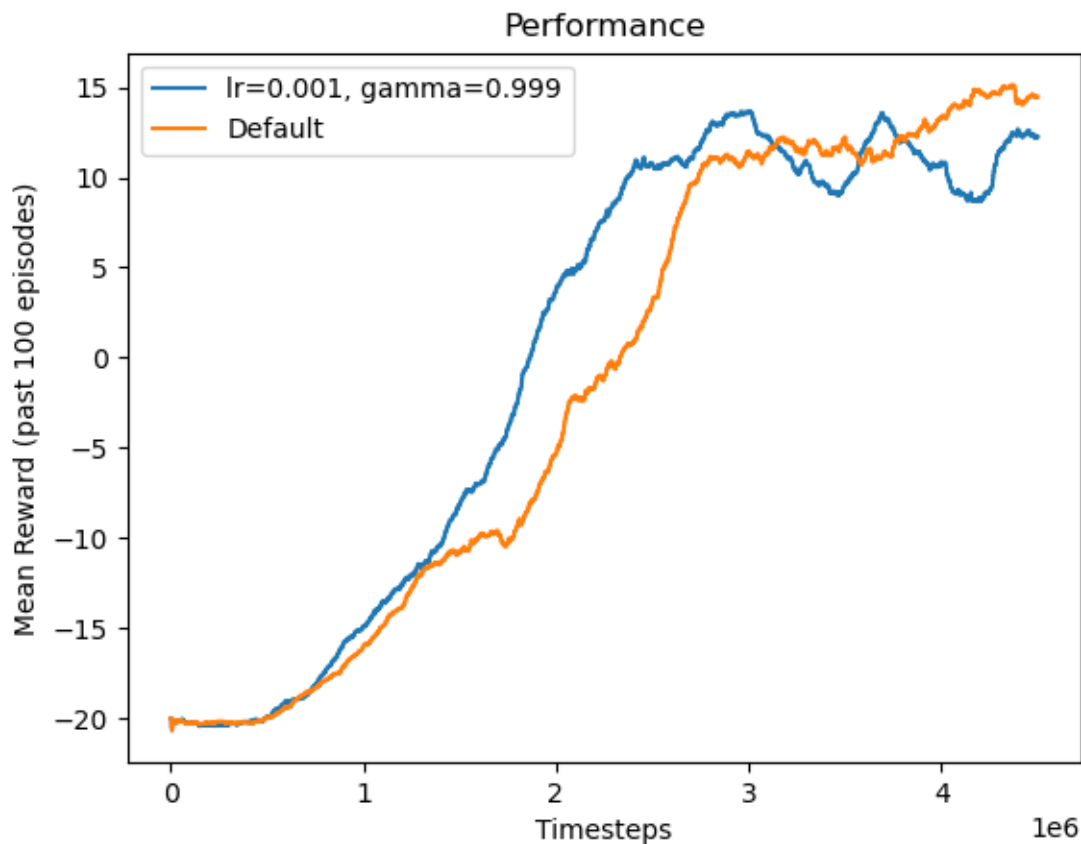
```
Timestep 4500000
mean reward (100 episodes) 14.480000
best mean reward 15.130000
episodes 2150
exploration 0.100000
```

 Q\_params.pkl

 target\_Q\_params.pkl

 openai gym.video.3.7  
video002000.mp4.33

Plotting the two runs together:



The chosen parameters (in blue) showed significantly better performance up to around 2.5M steps, Reaching 'mean reward' of  $\sim 10$  around the time the default's 'mean reward' crosses  $\sim 0$ . At 3M steps, the chosen parameters 'mean reward' begins to diverge, while the default parameters 'mean reward' continues to slowly and steadily climb, bringing it to a slightly better end-result.

Examining the results, it seems there is still room for improvement. Given another week or so, I would have tested a modified exploration schedules and an adjusted `TARGET_UPDATE_FREQ` parameter.

A different exploration schedule can have great effect on the reward at termination, but estimating a schedule requires a long run.

I suspect the loss function diverged on the high Learning-Rate DQN test, and a higher `TARGET_UPDATE_FREQ` (meaning lower update frequency of `Q_target`) could have improved results.

Would have also added the loss function plot to verify convergence of a net.