DQN

April 25, 2022

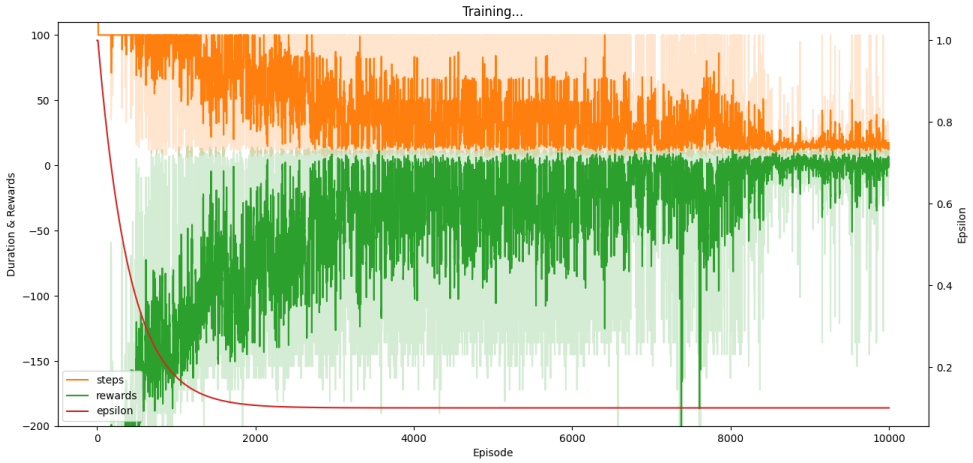
1 Deep Q Learning

[ ]:
```
from DQN_Train import TrainingAgent
import gym
```

1.1 Reference Architecture 1

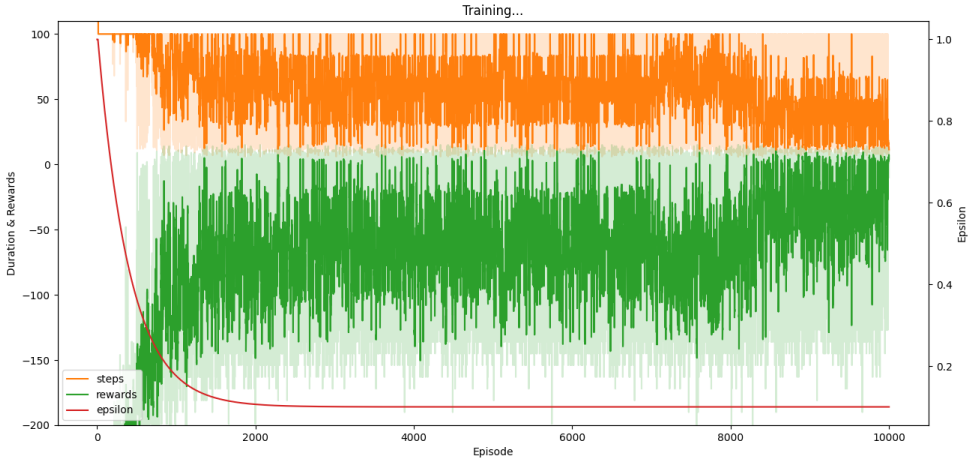
This model was trained to use as a reference to compare future model trained with different parameters

[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                      name="notebook_reference",
                      save=False,
                      architecture=1)
agent.compile()
agent.fit()
```



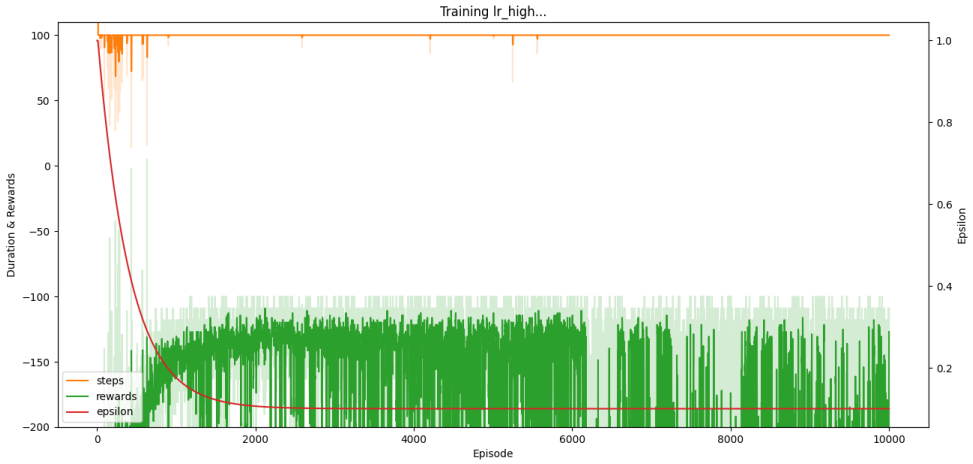
1.2 Constant Learning Rate

```
[ ]: env = gym.make("Taxi-v3").env
     env.seed(42)
     agent = TrainingAgent(env=env,
                           lr_min=0.001,
                           name="notebook_lr_const",
                           save=False,
                           architecture=1)
     agent.compile()
     agent.fit()
```



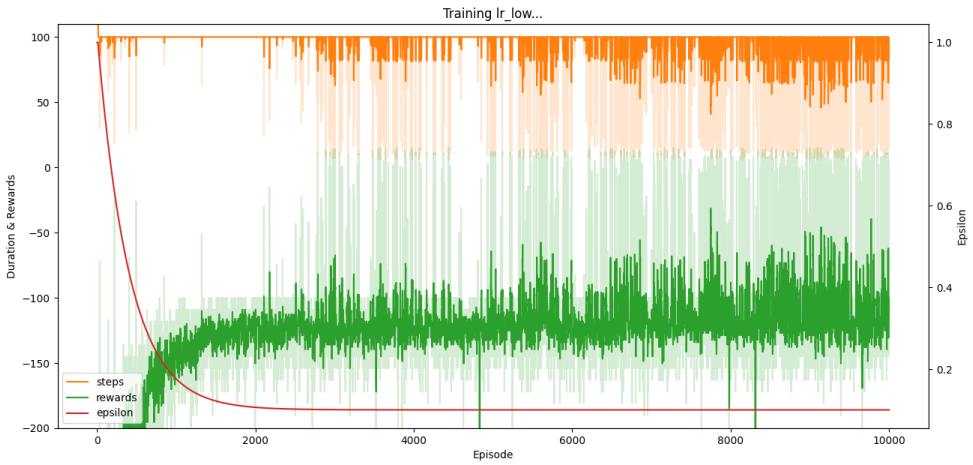
1.3 High Learning Rate

```
[ ]: env = gym.make("Taxi-v3").env
     env.seed(42)
     agent = TrainingAgent(env=env,
                           lr=0.01,
                           lr_min=0.01,
                           name="notebook_lr_high",
                           save=False,
                           architecture=1)
     agent.compile()
     agent.fit()
```



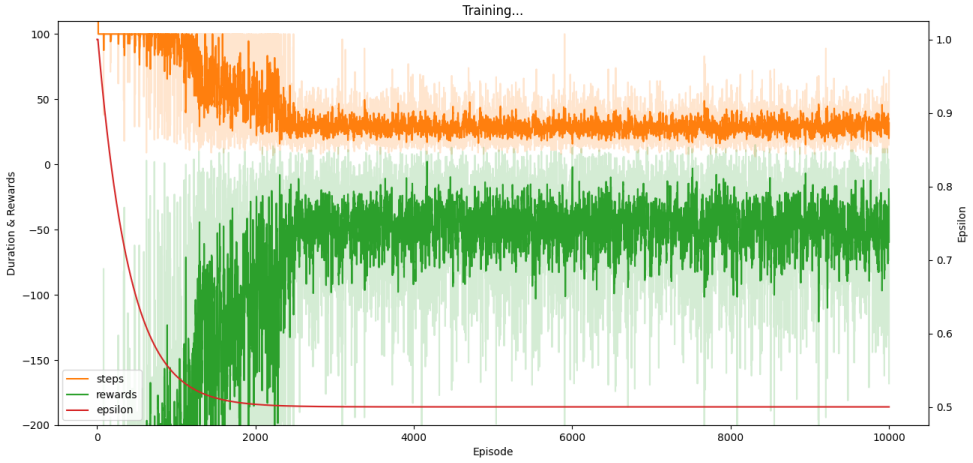
1.4 Low Learning Rate

[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                      lr=0.0001,
                      lr_min=0.0001,
                      name="notebook_lr_low",
                      save=False,
                      architecture=1)
agent.compile()
agent.fit()
```



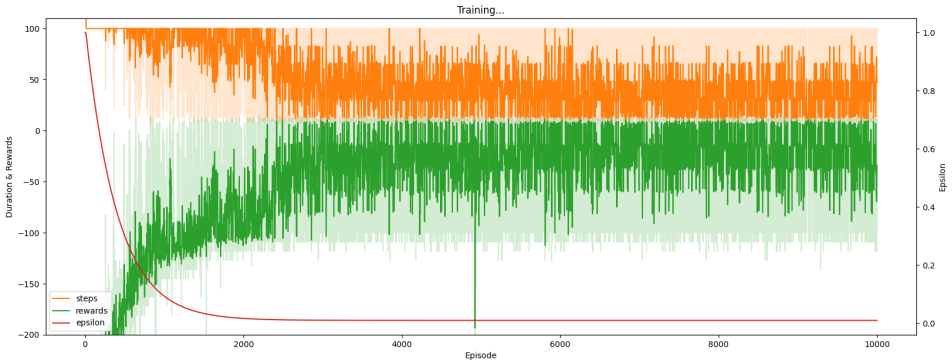
1.5 High Epsilon

```
[ ]: env = gym.make("Taxi-v3").env
     env.seed(42)
     agent = TrainingAgent(env=env,
                           eps_end=0.5,
                           name="notebook_eps_high",
                           save=False,
                           architecture=1)
     agent.compile()
     agent.fit()
```



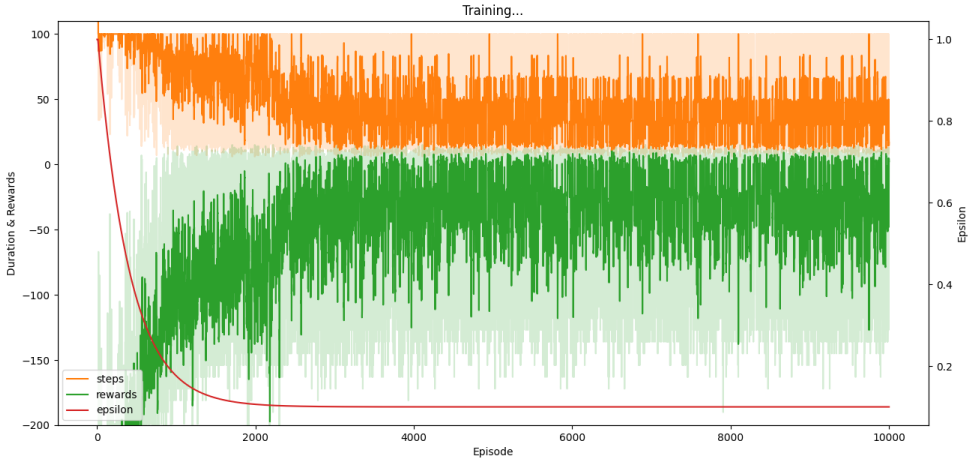
1.6 Low Epsilon

```
[ ]: env = gym.make("Taxi-v3").env
     env.seed(42)
     agent = TrainingAgent(env=env,
                           eps_end=0.01,
                           name="notebook_eps_low",
                           save=False,
                           architecture=1)
     agent.compile()
     agent.fit()
```



1.7 Update Rate High

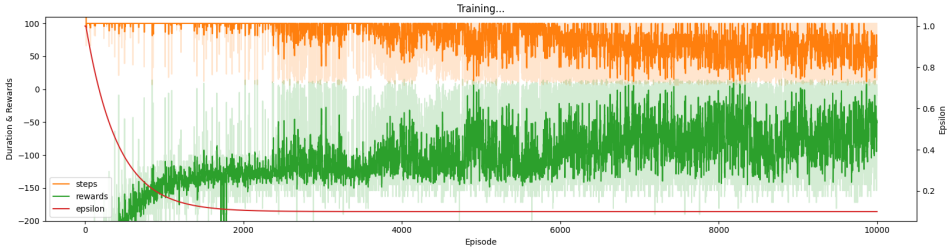
[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                      target_update=2,
                      name="notebook_update_high",
                      save=False,
                      architecture=1)
agent.compile()
agent.fit()
```



1.8 Update Rate Low

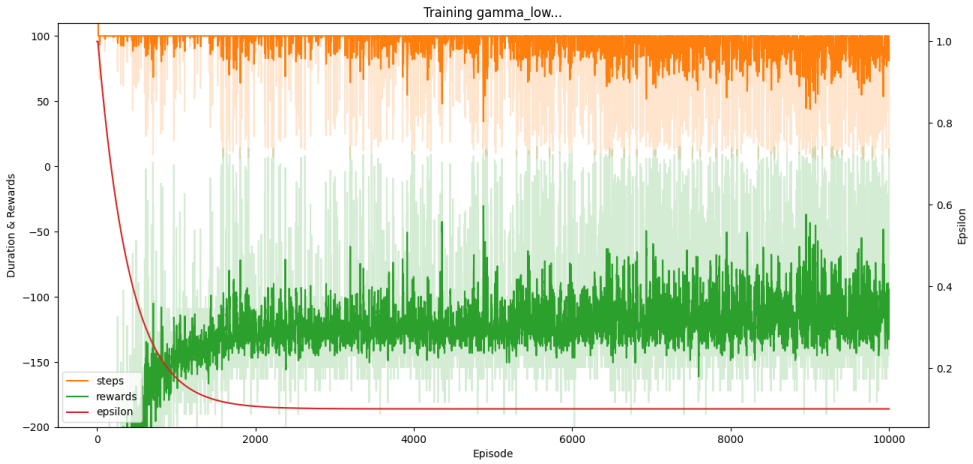
[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                      target_update=200,
                      name="notebook_update_low",
```

```
                        save=False,
                        architecture=1)
agent.compile()
agent.fit()
```



1.9 Gamma Low

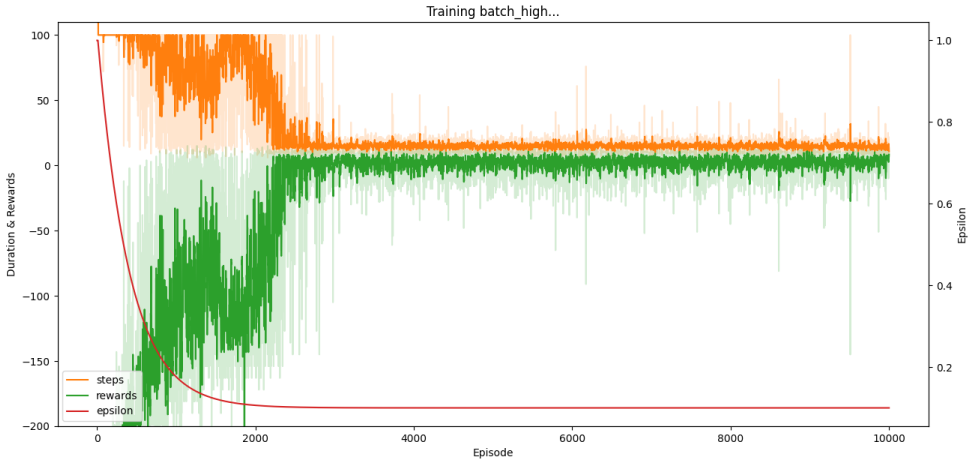
[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                        gamma=0.2,
                        name="notebook_gamma_low",
                        save=False,
                        architecture=1)
agent.compile()
agent.fit()
```



1.10 Batch Size High

[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                        batch_size=512,
```

```
                        name="notebook_batch_high",
                        save=False,
                        architecture=1)
agent.compile()
agent.fit()
```



1.11 Batch Size Low

[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                        batch_size=32,
                        name="notebook_batch_low",
                        save=False,
                        architecture=1)
agent.compile()
agent.fit()
```

1.12 Reference Architecture 2

[ ]:
```
env = gym.make("Taxi-v3").env
env.seed(42)
agent = TrainingAgent(env=env,
                      name="notebook_reference_2",
                      save=False,
                      architecture=2)
agent.compile()
agent.fit()
```