DM887 Reinforcement Learning Jiawei Zhao

Assignment 2: Deep Q-learning with autoencoder and LSTD

At Q1, I designed the algorithms of the DQN and formatted it in latex.

At Q2, I explored the three required Open AI gym environments and programmed a python class called “*LSTD_DQL_learner*” to suffice the requirements. Both the training mode and evaluation mode are available at the python class.

Comments on the environments: out of the three environments, ”Pendulum-v1" is the only one whose action space isn’t discrete. Therefore, I discretized its continuous 1-D action space “Torque” [-2, 2] into 100 separate actions each representing an interval of a length at 0.04. Furthermore, the variable *torque* at the term *torque^2* at the continuous reward function of ”Pendulum-v1" gets replaced by 100 discrete numbers each representing the median of its interval. Therefore, a child class of “*LSTD_DQL_learner*”, i.e. “*LSTD_DQL_pendulum_learner*” was designed with several methods at its parent class being overridden.

At Q3, with regard to each environment, I tried an encoding dimension that is two times as large as the number of observations at the env, as well as an encoding dimension that is the ceiling integer of half the number of observations at the env. I chose the encoding dimension which results in a better *gradient descent time step - total episode reward* curves. As per my experiments, halving the encoding dimension yields a better learning curve, possibly due to the less sparse matrices representing the weights of the autoencoder layers.

Unfortunately, my DQN fails to solve the environment “MountainCar-v0” in spite of numerous attempts on different training parameters, while it occasionally solves “Acrobot-v1”, possibly due to the low dimension of observation spaces of “MountainCar-v0”.

With regards to the learning curve of ”Pendulum-v1" by “*LSTD_DQL_pendulum_learner*”, in spite of an extremely large number of discretized actions and numerous attempts to fine-tune the training parameters, the observed improvement of total episode rewards at the evaluation mode has been measly. Moreover, the autoencoder weight update phases are observed to worsen the total episode rewards at evaluation over the entire training process, while LSTD weight updates are more likely to improve the total episode rewards at evaluation slightly.