Report of Deep reinforcement learning Nano-degree Project 1: Navigation

2019-2-17

In this report, I would like to briefly summarize the DRLN project: Navigation. The typical Deep Q-Network Architecture (DQN) algorithm was applied in this project and the environment can be solved in 300-400 episode. Some details are described in the following.

Environment

In this environment, the subject is to navigate an agent to wonder around in a land of bananas, while collecting as much as yellow bananas and avoiding the violet bananas. The environment space is continuous and was discretized into 37 dimensions, the agent's action space is 4. The requirement of this project is to train the agent to achieve more than 13 points as least in the test run.

DQN

I borrowed the typical DQN framework of the former lessons, and setup 3 fully connected layers, with the ReLU function as the activation function. Since we are working on a small state and action space, I assume that there is no need to have 4 layers or more which complexity is too high.

With 3 fully connected layers, I tried the layer size of 64/64/32、128/64/32 and 265/128/64, they all seem to solve the problem with limited episodes, thus a typical 128/64/32 setup was applied here.

Tuning ϵ

During the experiments, I noticed that if ϵ decays a little faster, this could help the agent to learn faster. It seems in this environment the agent does not need to spend too much episodes in exploration. This may due to the simplicity of the environment. I tried out some combinations of (eps_start, eps_end, eps_decay) and settled for the combination:

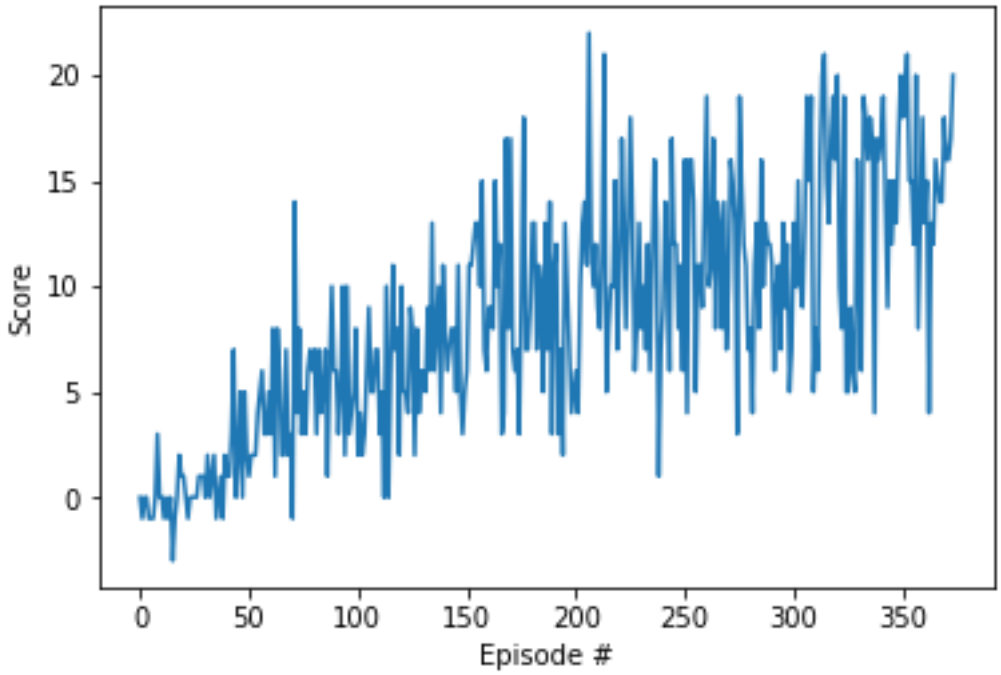
- eps_start=1.0

- eps_end=0.005
- eps_decay=0.985

Where the ϵ decays faster and also ends up small.

Results

Under the setup above, The problem was resolved at the 374th episodes, and has a average score of 13.01. At this point , ϵ has already decayed to 0.005. The Training process is shown in the figure below:



During the training, it is interesting to notice an interesting scene that shows the growing knowledge of an agent. When an agent is not well trained enough, it knows the idea of getting yellow bananas and avoiding violet bananas but did not know how to go around a violet one. Thus, this agent easily got stuck in front of a violet banana and just goes back and forth. This was solved by not setting the eps_decay too large and not having too small neural network..