Brianna Solano
Project 2
Robotics and Computer Vision

Algorithm

The algorithm we are using is provided by PyTorch in their official page "REINFORCEMENT LEARNING (DQN) TUTORIAL". From the tutorial we are able to train a DQN tutorial to play the game CartPole. This algorithm attempts to train a policy such that it tries to maximize the return or reward. It predicts the expected return from taking either action and takes the action that has the larger expected reward. The CartPole task provided us with 4 variables that help represent the environment state which are cart position, velocity, of car, angle of pole, and rotation of pole. This algorithm represents the state as the difference between the current and previous screen patches. The algorithm chooses between two actions to take when completing the task, CartPole, pushing the cart left or right. For every incremental time-step that the algorithm is able to complete without failing (pole falls over too far or the cart moves more than 2.4 units away from center), the algorithm is rewarded +1. The model contains a convolution network that outputs Q(s, left) and Q(s,right). The neural network takes state, which is the difference of the current patch of screen and the previous patch of screen. This network returns two outputs, which is the maximum return of taking either action. Our policy in the program attempts maximize return, or in this case reward.

Plot Comparison

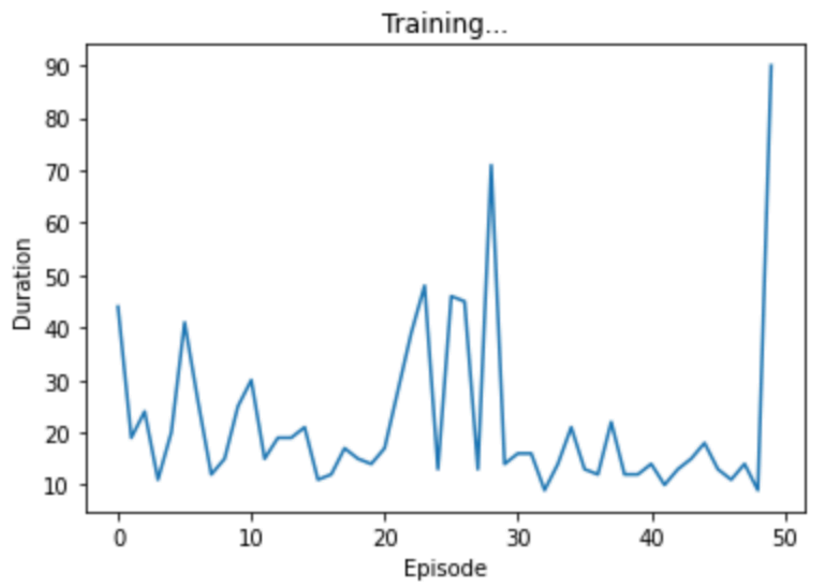

Figure 1: Graph of Duration (Total time before it fails) vs Episodes of agent trained for a short period of time

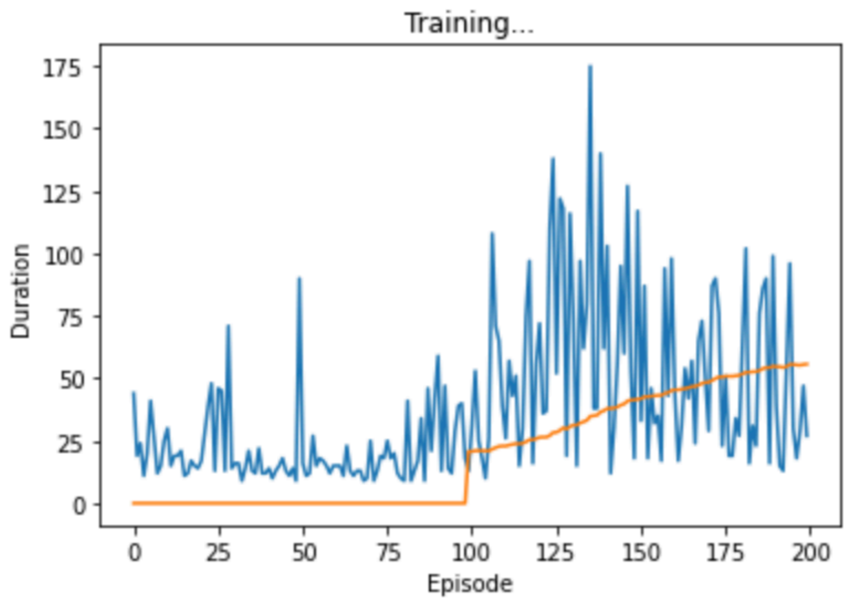

Figure 2: Graph of Duration (Total time before it fails) vs Episodes of agent trained for a long period of time

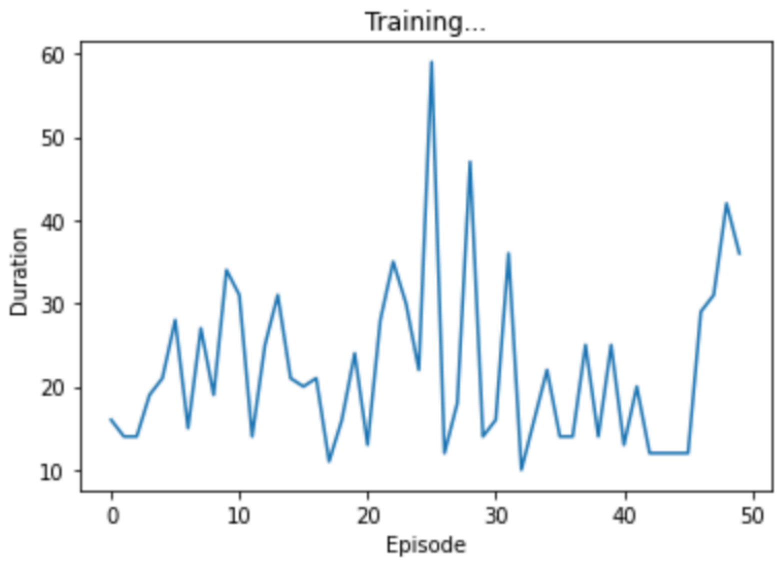

Figure 3: Graph of Duration (Total time before it fails) vs Episodes of random agent