Introduction to Reinforcement Learning

Project 2: Continuous Control

Jose Lara | jose.lara.l@outlook.com

Problem Statement

The goal of this project is to train an agent to navigate the Reacher Unity environment to maximize rewards. The environment involves a controlling a double-joined arm that can move to target locations around the environment. A reward of +0.1 is provide for each step that the agent's end-effector is within the goal location. The observation space consists of 33 variables, and four actions corresponding to the torque applicable to the two joints. Two environment options were given in this assignment:

1. Solve environment with 1 agent (ML Unity Reacher Environment)
2. Solve environment with 20 identical agents (ML Unity 20 Reacher Environment)

Using a Deep Q Network, based on PyTorch library, and a DDPG algorithm, a final set of optimized weights generated through training must provide an optimal solution. The RL agent based of these weights must be able to average a reward of +30 over 100 consecutive episodes.

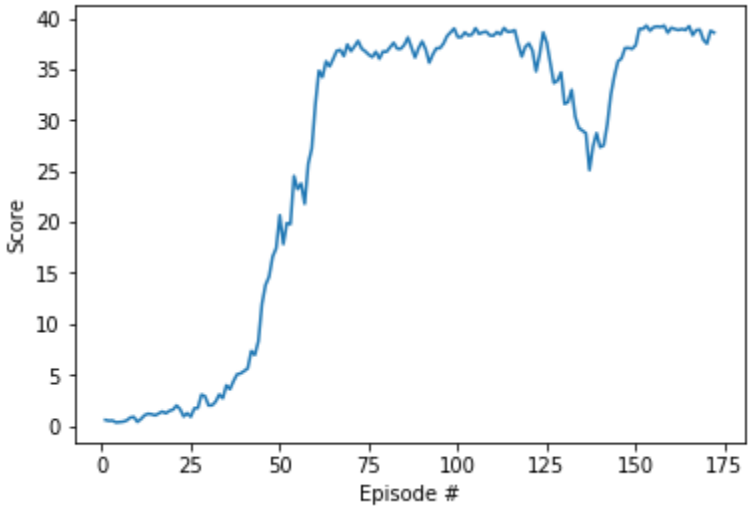

Figure 1: Training reward example of the average rewards for a trained agent that meets the assignment's criteria

Design

Model: Deep Q Networks:

An Actor Critic Method was used to solve this problem. Both the actor and the critic architecture are composed of neural networks with two hidden layers, each with 128 nodes. Both neural networks use the Leaky Rectified Linear Unit activation function, with 'leakiness' parameter that can be changed via a training agent argument to try different training configurations.

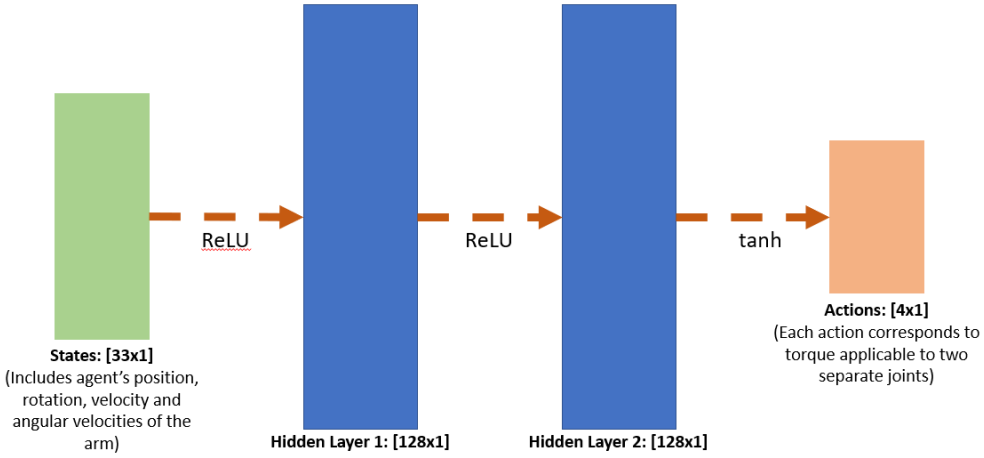

Figure 2: Actor Neural Network configuration includes an additional hyperbolic function to normalize inputs to correct range.

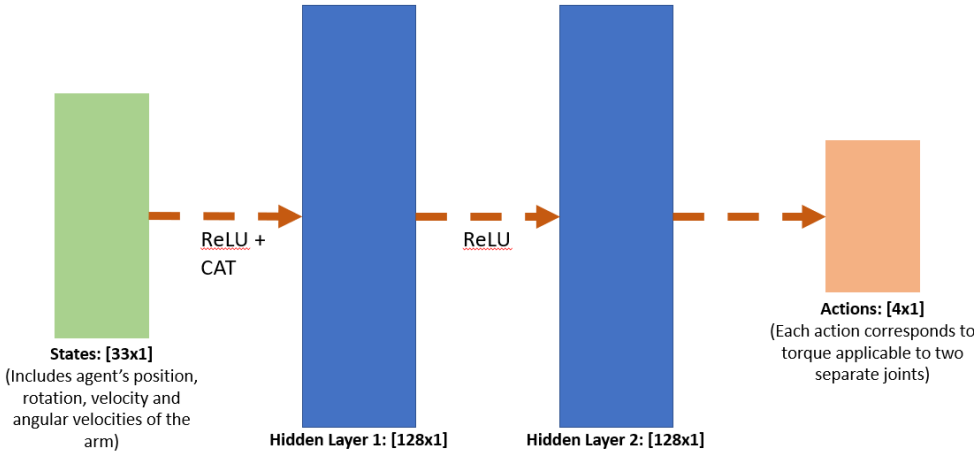

Figure 3: Critic Neural Network configuration.

Algorithm

The algorithm used mainly stems from the example provided in the Deep Deterministic Policy Gradient, Example in **Lesson 5: Actor-Critic Methods**. In the same algorithm, noise is added to each episode to improve the training. Also, a queue with results from previous episodes to sample experience replay.

Training Results

My recommendation to others solving this problem is to use the 20 Reacher Environment instead of the single agent environment. I attempted to train a DDPG brain using the single-agent environment and struggled to get acceptable results. Using different variables to find a signal on which parameters were impacting training. Below is a table that shows a few of my initial runs:

BATCH	LR_ ACTOR	LR_ CRITIC	DECAY	LEAK	Results
1024	1e-4	3e-4	0.0001	0.01	Training on cpu started... Episode: 10 Average Score: 0.27 Current Score: 0.33 Episode: 20 Average Score: 0.52 Current Score: 0.91 Episode: 30 Average Score: 0.69 Current Score: 1.45 Episode: 40 Average Score: 0.83 Current Score: 1.40
1024	1e-4	1e-4	0.0001	0.01	Training on cpu started... Episode: 10 Average Score: 0.28 Current Score: 0.29 Episode: 20 Average Score: 0.63 Current Score: 1.10 Episode: 30 Average Score: 0.67 Current Score: 0.56 Episode: 40 Average Score: 0.79 Current Score: 0.83 Episode: 50 Average Score: 0.86 Current Score: 1.18 Episode: 60 Average Score: 0.92 Current Score: 0.94 Episode: 70 Average Score: 1.08 Current Score: 3.50 Episode: 80 Average Score: 1.10 Current Score: 1.14 Episode: 84 Average Score: 1.09 Current Score: 1.96
1024	2e-3	3e-4	0.0001	0.01	Training on cpu started... Episode: 10 Average Score: 0.26 Current Score: 0.00 Episode: 20 Average Score: 0.42 Current Score: 0.06 Episode: 23 Average Score: 0.41 Current Score: 0.11
128	1e-3	1e-4	0.0001	0.01	Training on cpu started... Episode: 10 Average Score: 0.55 Current Score: 0.79 Episode: 20 Average Score: 0.73 Current Score: 0.91 Episode: 30 Average Score: 0.81 Current Score: 0.92 Episode: 40 Average Score: 0.84 Current Score: 1.90

512	5e-4	5e-4	0.005	0.01	Training on cpu started... Episode: 10 Average Score: 0.14 Current Score: 0.18 Episode: 20 Average Score: 0.30 Current Score: 0.55 Episode: 29 Average Score: 0.43 Current Score: 1.28

After having difficulties training my agent, I reached out to the Udacity mentor's to help me with my work. The best suggestion that helped was to change environments. As soon as I changed to the 20 Reacher environment, I was able to start tuning the training parameters, more specifically, the Learning Rates for the Actor and Critic neural networks.

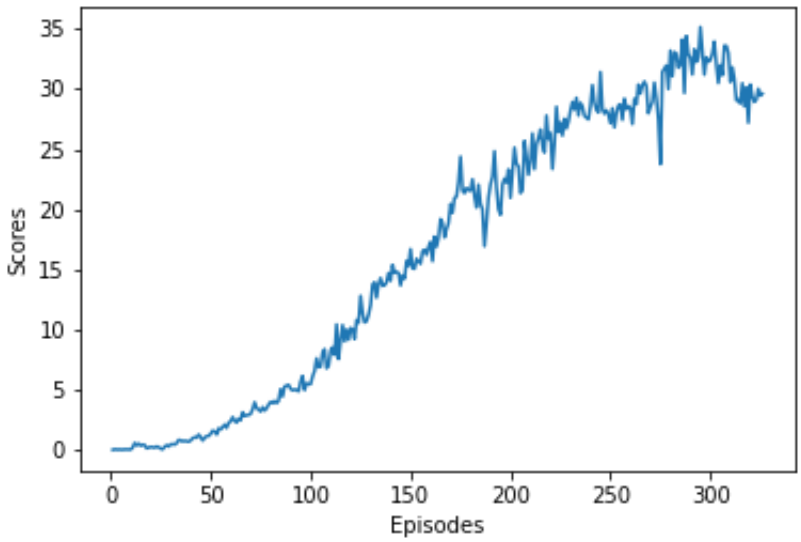

Figure 3: Earned rewards through training agent using the 20 Reacher environment, with an average of 30.00 in rewards after within 100 episodes. The solution was reached after 326 episodes.

Next Steps

First steps I want to take with this project is generalizing the code to work with multiple Udacity environments. Creating a list of all applicable environment, and using the correct state values, build an applicable Deep Q Network.

Next, I would like to build an optimization script that iterates through hyperparameters at the model and algorithm label. These parameters include number of nodes in each hidden layer, buffer size and learning rate,

Furthermore, I would like to use the DDPG agent in the Multi Agent assignment and continue improving the efficiency of the mode.

Appendix

1) I was helped by a Udacity mentor in the following post: https://knowledge.udacity.com/questions/855616