Solving Reacher environment using Deep Deterministic Policy Gradient (DDPG)

Author: Jakub Bielan

Learning Algorithm

DDPG is a model-free policy based learning algorithm in which the agent will learn directly from the un-processed observation spaces without knowing the domain dynamic information. That means the same algorithm can be applied across domains which is a huge step forward comparing with the traditional planning algorithm. In contrast with DQN that learn indirectly through Q-values tables, DDPG learns directly from the observation spaces through policy gradient method which estimates the weights of an optimal policy through gradient ascent which is similar to gradient descent used in neural network. Also, policy based method is better suited in solving continuous action space environment. DDPG also employs Actor-Critic model in which the Critic model learns the value function like DQN and uses it to determine how the Actor's policy based model should change. The Actor brings the advantage of learning in continuous actions space without the need for extra layer of optimization procedures required in a value based function while the Critic supplies the Actor with knowledge of the performance.

To mitigate the challenge of unstable learning, a number of techniques are applied like Soft Target Update through twin local / target network and Replay Buffer. The most important one is Replay Buffer where it allows the DDPG agent to learn offline by gathering experiences collected from environment agents and sampling experiences from large Replay Memory Buffer across a set of unrelated experiences. This enables a very effective and quicker training process. Also, Batch Normalization plays an important role to ensure training can happen in mini batch and is GPU hardware optimization friendly.

I have decided to not use Gradient Clipping as I don't believe the concept fully and very likely it's a misconception repeated in many projects.

Model Architecture

The main idea behind this project is to use experience of 20 active agents and train just one brain (2 neural networks) helping them maximize total reward.

The Actor model is a neural network with 2 hidden layers with size of 256 and 128, Tanh is used in the final layer that maps states to actions. Batch normalization is used for mini batch training.

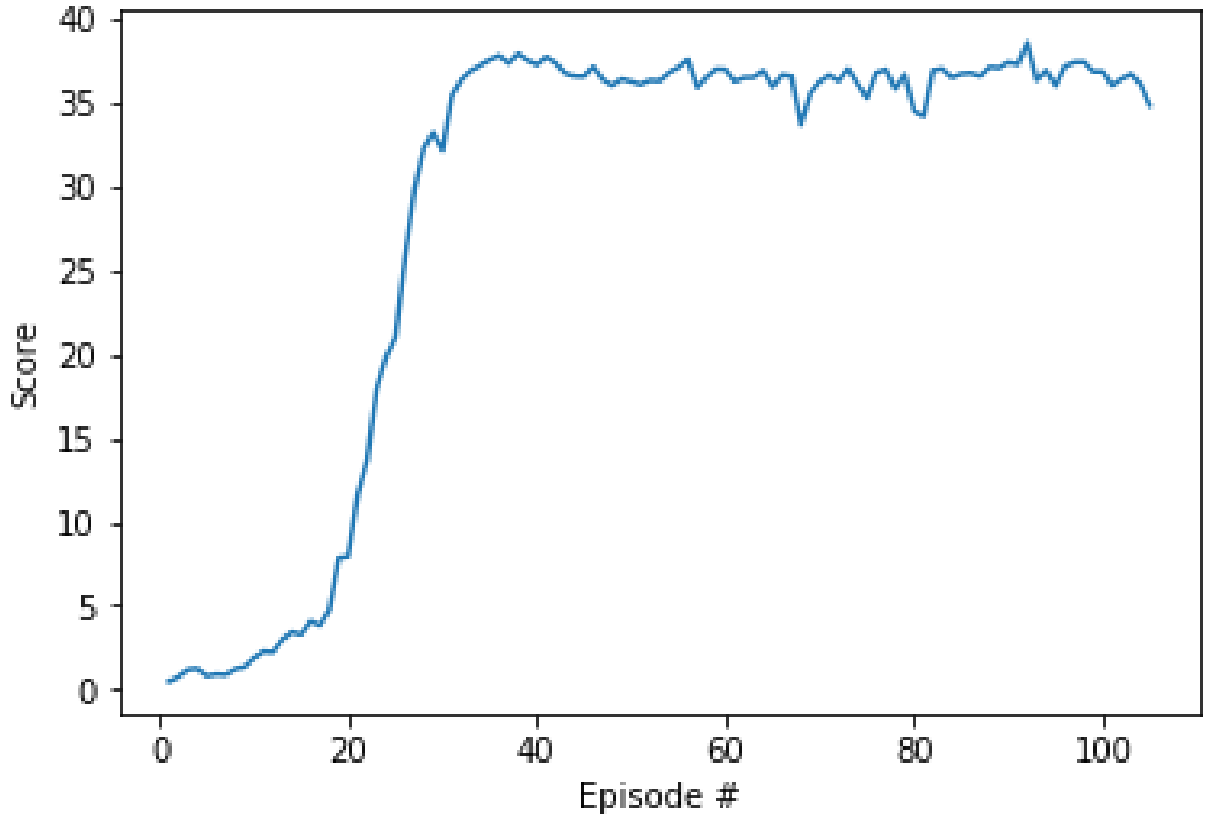
The Critic model is similar to Actor model except the final layer is a fully connected layer that maps states and actions to Q-values.

Hyperparameters

BATCH_SIZE = 128
BUFFER_SIZE = int(7e4)
GAMMA = 0.99
LR_ACTOR = 1e-4
LR_CRITIC = 1e-4
TAU = 1e-3
WEIGHT_DECAY = 0

Training Result

It took 105 episodes to achieve an average reward of 30 scores in about an hour without even using GPU. Generally GPU optimization is not as important in reinforcement learning as for example in computer vision. To optimize performance most of the time it’s better to run episodes simultaneously on many CPUs. Model achieved peak performance at about 32 episode and it keeps it till end. You can see it on the graph:



Future improvements

We could improve the novel DDPG algorithm by applying Priority Experienced Replay in which important experience will be sampled more often. Based on the paper “A novel DDPG method with prioritized experience replay”, it can reduce the training time, improve the stability of the training process and is less sensitive to the change in hyperparameters