Project Report

The goal of this project was to train a simulated robot arm to reach for a certain position nearby. The training environment allows for multiple parallel agents, which benefits the generation of training data and breaks correlations that may hinder training.

Environment

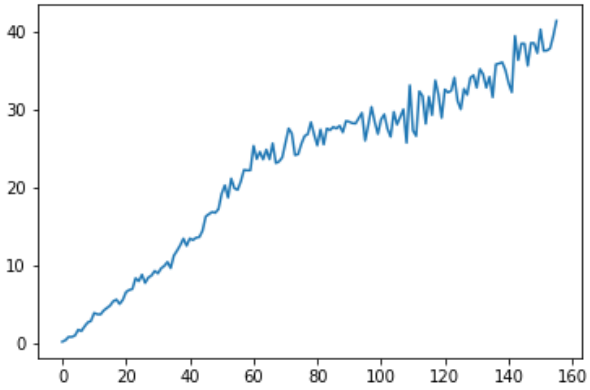
The environment is a Unity Machine Learning Agents environment with a pre-compiled executable. This project uses Version 2, which comes with 20 independent identical agents acting in the environment. Every agent receives a positive reward whenever its arm reaches inside a spherical object that hovers around it. The observation space has 33 dimensions and contains positions, rotations and angular velocities. The action space per agent contains 4 continuous actions with values between -1.0 and +1.0, which correspond to torque on the robot arm's joint motors. The environment is considered solved when the average reward over all agents over the last 100 episodes is larger or equal to 30.

Learning Algorithm

The agent is trained using Proximal Policy Optimization (PPO) with clipped surrogate functions (see Readme file for references) with a basic actor-critic setup. Both actor and critic are realized as neural networks with two hidden layers with 64 units each and ReLU activation. Due to the multi-agent nature of the environment, PPO is especially useful since its original version was already intended to run on several parallel agents. During training, the algorithm first collects trajectory data up to a horizon tmax=300. It also calculates advantages from the rewards with a discount factor of 0.98. Then the algorithm performs 5 epochs of PPO with a mini-batch size of 500. The r-value used is 0.2 and the calculated gradients are clipped at a value of 0.5. The code used here is partially based on the PPO exercise in the DRLND github repository and the PPO agent in Shangtong Zhang's DeepRL github repository. However, most parts were written from scratch with information from the OpenAI paper in order to accommodate the UnityML environment.

Rewards

The agent was able to reach a 100-episode-average reward of 30.114 in 156 episodes. The reward plot shows a steady learning throughout all episodes with some increased variance after 60 episodes.



Future Work

The hyperparameters used are close to the ones provided in the continuous control examples from the DeepRL repository. Thus, it is possible that further tuning might speed up training. It would also be interesting to see the effects of a GAE implementation. On the other hand, it also remains to be seen what happens when the agent does not use a critic function at all and instead directly uses the environment rewards, as in the DRLND exercise example.