Noushin Islam Student No. 201508438

Assignment 2 – Report

Introduction

With Reinforcement Learning (RL), robots can be trained to perform tasks autonomously. One of the key algorithms in RL is the Soft Actor-Critic (SAC) algorithm. This report provides an overview of the SAC algorithm, its suitability for training robots, and the method used to apply it to the PandaPickAndPlace-v3 environment.

Method

The SAC algorithm is a model-free Reinforcement Learning algorithm that uses a policy and value function to maximize the expected cumulative reward. The algorithm optimizes a stochastic policy while maximizing the entropy of the policy, which can be achieved with the use of entropy regularization. This approach prevents the policy from prematurely converging to a bad local optimum, making it suitable for complex tasks. The SAC algorithm is also well-suited for continuous action spaces, which are common in robotic control tasks.

To apply the SAC algorithm, a Jupyter Notebook file is used to train the Panda robot arm in the PandaPickAndPlace-v3 environment. The environment, which has a continuous action space, simulates the robotic arm to pick up an object and place it in a designated location.
After creating an instance of the environment, the SAC algorithm is initialized with a multi-input policy and the model is then trained for the chosen amount of timesteps. After training, the model is saved and loaded for testing. Then a new instance of the environment is created, which runs the loaded model for 200 episodes. During each episode, the model predicts an action based on the current observation, and the environment returns a reward with the next observation. The code also tracks the success rate, episode length, and cumulative reward for each episode to then plot it.

During the training process, one can manually set the hyperparameters such as the learning rate, which is crucial to achieving good performance. The default learning rate of 0.0003 was found to be the most effective, as a higher learning rate can lead to model instability, and a lower learning rate can cause slow convergence. It is to note that, achieving good performance in RL often requires trial and error to find the optimal combination of hyperparameters and network architecture.

Result

The Soft Actor-Critic algorithm has been deemed effective in training robots in continuous spaces. To achieve a high success rate, a model needs to be trained for a set amount of timesteps. Due to my hardware limitations, I was only able to train it for nearly one million timesteps, which took me 11.5 hours. This is not sufficient for great results. The success rate is calculated as the percentage of episodes in which the robot successfully completes the task. The results of running the trained model on the PandaPickAndPlace-v3 environment show how well the model has learned to perform the task. By plotting the success rate, episode length, and cumulative reward over time, we can see how these metrics change as the model interacts with its environment.

Conclusion

In conclusion, the SAC algorithm is an effective approach for training robots to perform tasks autonomously in real-world scenarios. By balancing exploration and exploitation through maximum entropy reinforcement learning, SAC can learn effective policies for complex tasks. The example code demonstrates how SAC can be used with Stable Baselines3 to train a model on the PandaPickAndPlace-v3 environment.

References:

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018).
Soft actor-critic: *Off-policy maximum entropy deep reinforcement learning with a stochastic actor*.

Stable Baselines3. Soft Actor-Critic (SAC).
Retrieved from *https://stable-baselines3.readthedocs.io/en/master/modules/sac.html*