UDACITY

‹ Return to "Machine Learning Engineer Nanodegree" in the classroom

DISCUSS ON STUDENT HUB

Teach a Quadcopter How to Fly

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

You have done a really awesome job. I have to appreciate the effort you have put in completing this project.

Some useful resources:

- https://www.youtube.com/watch?v=lvoHnicueoE
- https://www.youtube.com/watch?v=UoPei5o4fps&t=2085s
- http://pemami4911.github.io/blog/2016/08/21/ddpg-rl.html

Define the Task, Define the Agent, and Train Your Agent!

The `agent.py` file contains a functional implementation of a reinforcement learning algorithm.

- I have to appreciate you have properly implemented the agent with DDPG algorithm.
- The actor and critic networks are properly implemented.
  Useful resource for understanding different algos in reinforcement learning.
- https://towardsdatascience.com/introduction-to-various-reinforcement-learning-algorithms-i-q-learning-sarsa-dqn-ddpg-72a5e0cb6287

The `Quadcopter_Project.ipynb` notebook includes code to train the agent.

Great! Including the code in the notebook to train the agent.

Plot the Rewards

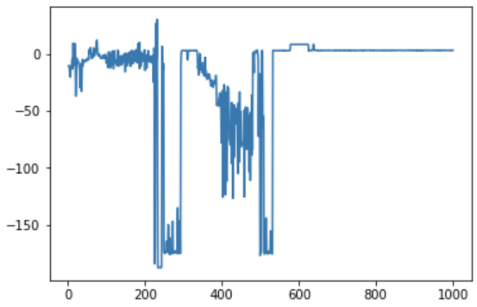
A plot of rewards per episode is used to illustrate how the agent learns over time.

Great job plotting reward obtained over period of time. The plot illustrates the learning of the agent over the time.

```
[32]: '''THESE REWARDS ARE TAKEOFF TASK'''

      import matplotlib.pyplot as plt
      %matplotlib inline

      plt.plot(results['episode2'], results['total_reward2'])
      #plt.legend()
      _ = plt.ylim()
```



Reflections

The submission describes the task and reward function, and the description lines up with the implementation in `task.py`. It is clear how the reward function can be used to guide the agent to accomplish the task.

Task and reward function is properly descibed and implemented. Good reward function is used it has the potential to learn the desired tasks.👏
Refernce for designing good reward function - https://bons.ai/blog/deep-reinforcement-learning-models-tips-tricks-for-writing-reward-functions

The submission provides a detailed description of the agent in `agent.py`.

The agent.py file includes the detailed description of the agent, implementation of the actor and critic network is there in the code.It is clear that you have tried various architectures and tuning hyper-parameters for the agent to learn the tasks.

The submission discusses the rewards plot. Ideally, the plot shows that the agent has learned (with episode rewards that are gradually increasing). If not, the submission describes in detail various attempted settings (hyperparameters and architectures, etc) that were tested to teach the agent.

Awesome! You have clearly mentioned out the implementation, algorithm, hyperparameters and model architecture. 👏

A brief overall summary of the experience working on the project is provided, with ideas for further improving the project.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review