UDACITY

< Back to Deep Reinforcement Learning Nanodegree

# Collaboration and Competition

REVIEW

CODE REVIEW

HISTORY

## Meets Specifications

Awesome ⭐⭐⭐⭐⭐
You have done a great work and acquired all the concepts needed in this project. Congratulations 😃

Some good resources to refer:

- The Coopetition Dilemma: Building Reinforcement Learning Agents that Learn to Collaborate and Compete at the Same Time
- Competition and collaboration among fuzzy reinforcement learning agents
- Learning to Cooperate, compete and communicate

### Training Code

**The repository includes functional, well-documented, and organized code for training the agent.**

Well documented code. 👍 Some good resources to make it further better:

- Documenting Python Code: A Complete Guide
- Google Python Style Guide

**The code is written in PyTorch and Python 3.**

Now that you have mastered PyTorch you can learn about tensorFlow:

- TensorFlow Vs PyTorch: Which Framework Is Better For Implementing Deep Learning Models?
- PyTorch vs Google TensorFlow – Which AI will take over the world?

A list of books to help you in your further study

**The submission includes the saved model weights of the successful agent.**

Saved model weights are included.

## README

**The GitHub submission includes a `README.md` file in the root of the repository.**

`Readme.md` is included.

**The README describes the the project environment details (i.e., the state and action spaces, and when the environment is considered solved).**

The Readme file describes the project environment details.

**The README has instructions for installing dependencies or downloading needed files.**

Well documented `Readme.md` you have given clear instructions for downloading the relevant files, details about the project environment etc. Great work. 😀

However, it is a convention to include dependencies too in `Readme.md` you can add in the readme that you need Python 3.5, PyTorch, Numpy... etc so that anyone accessing your repo, knows what are the modules required.

**The README describes how to run the code in the repository, to train the agent. For additional resources on creating READMEs or using Markdown, see here and here.**

The instructions to train the network are included. Good Job.
You can make the instructions more clear by including the cell numbers needed to:

- Train the Agent.
- Run the trained Agent after loading the stored weights

## Report

**The submission includes a file in the root of the GitHub repository (one of `Report.md`, `Report.ipynb`, or `Report.pdf`) that provides a description of the implementation.**

`Report.pdf` is included in the submission.

**The report clearly describes the learning algorithm, along with the chosen hyperparameters. It also describes the model architectures for any neural networks.**

Awesome! 😃 A very well written report. It includes the algorithm, model architecture and major hyperparameters. The report clearly demonstrates that you have understood the concepts of competition and collaboration among agents. 🏆

**A plot of rewards per episode is included to illustrate that the agents get an average score of +0.5 (over 100 consecutive episodes, after taking the maximum over both agents).**

**The submission reports the number of episodes needed to solve the environment.**

Great work in achieving an average score of +0.5 (over 100 consecutive episodes, after taking the maximum over both agents).
It is nice you experimented with batch size.

**The submission has concrete future ideas for improving the agent's performance.**

Nice work suggesting the ideas for future improvement.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review