UDACITY

< Return to Classroom

DISCUSS ON STUDENT HUB

# Continuous Control

REVIEW

CODE REVIEW

HISTORY

## Meets Specifications

Dear student, great job. Performance of the agent is very good and the project meets the specifications. 😄

The recent application of deep reinforcement learning to play the game Dota 2 (https://openai.com/blog/dota-2/) is a substantial step.

Also, to know more about reinforcement learning algorithms and applying these to real world problems, please do check
Deep Reinforcement Learning (https://www.youtube.com/watch?v=MQ6pP65o7OM) which is a part of the course MIT 6.S094: Deep Learning for Self-Driving Cars (2018 version).

Some of the resources to read

- for continuous control using deep reinforcement learning (https://arxiv.org/abs/1509.02971)
- Towards Generalization and Simplicity in Continuous Control (https://proceedings.neurips.cc/paper/2017/file/9ddb9dd5d8aee9a76bf217a2a3c54833-Paper.pdf)
- Robust Reinforcement Learning for Continuous Control with Model Misspecification (https://deepmind.com/research/publications/2019/Robust-Reinforcement-Learning-for-Continuous-Control-with-Model-Misspecification)

Happy learning and all the very best

## Training Code

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

Awesome work implementing a reinforcement learning agent to solve the "reacher" environment.

- TD3 is an off-policy algorithm.
- TD3 can only be used for environments with continuous action spaces.
- The Spinning Up implementation of TD3 does not support parallelization.
- TD3 concurrently learns two Q-functions, Q{\phi_1} and Q{\phi_2}, by mean square Bellman error minimization, in almost the same way that DDPG learns its single Q-function.

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

Good job completing the project using Pytorch and Python3.

You should definitely check this post comparing different deep learning frameworks: Deep Learning Frameworks Comparison – Tensorflow, PyTorch, Keras, MXNet, The Microsoft Cognitive Toolkit, Caffe, Deeplearning4j, Chainer (https://www.netguru.com/blog/deep-learning-frameworks-comparison)

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

Great work. The submission includes the saved model weights of the successful agent.

## README

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

Great work here. Very well done. A detailed README file has been provided and is present in the repository.

Take a look at the following links to improve how your README looks

- https://blog.bitsrc.io/how-to-write-beautiful-and-meaningful-readme-md-for-your-next-project-897045e3f991
- https://dev.to/scottydocs/how-to-write-a-kickass-readme-5af9

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

Awesome work providing the project environment details including the state and action spaces, the reward function and when the agent is considered solved. The description is provided in the Introduction and Solving the Environment sections and is very informative

Solving the Environment sections and is very informative.

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

Great work. The README talks about on how to install the dependencies and how to run the code. All the requirements are met. Very nicely done.

**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.**

Great work. You have explained how to run the code and train the agent. Very well done. Keep up the good work

## 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.**

Brilliant work. Report has been included in the root of the github repository.

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

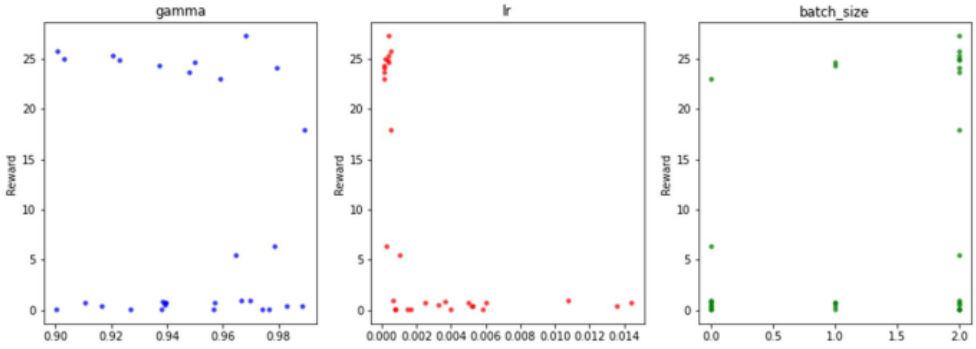
Description of the learning algorithm, hyperparameters, and network architecture have been provided in great detail in the project report.

**{'action_size': 4, 'batch_size': 128, 'brain_name': 'ReacherBrain', 'gamma': 0.9681993333005682, 'lr': 0.00036026520899097987, 'n_agents': 20, 'state_size': 33}**

**With a reward of 27.26 after 100 experiments**

```
parameters = ['gamma', 'lr', 'batch_size']
colors =  ['blue', 'red', 'green']
cols = len(parameters)
f, axes = plt.subplots(nrows=1, ncols=cols, figsize=(15,5))
cmap = plt.cm.jet

for i, val in enumerate(parameters):
    xs = np.array([t['misc']['vals'][val] for t in trials.trials]).ravel()
    ys = [-t['result']['loss'] for t in trials.trials]
    xs, ys = zip(*sorted(zip(xs, ys)))
    ys = np.array(ys)
    axes[i].scatter(xs, ys, s=20, linewidth=0.01, alpha=0.75, c=colors[i])
    axes[i].set_title(val)
    axes[i].set_ylabel('Reward')
```

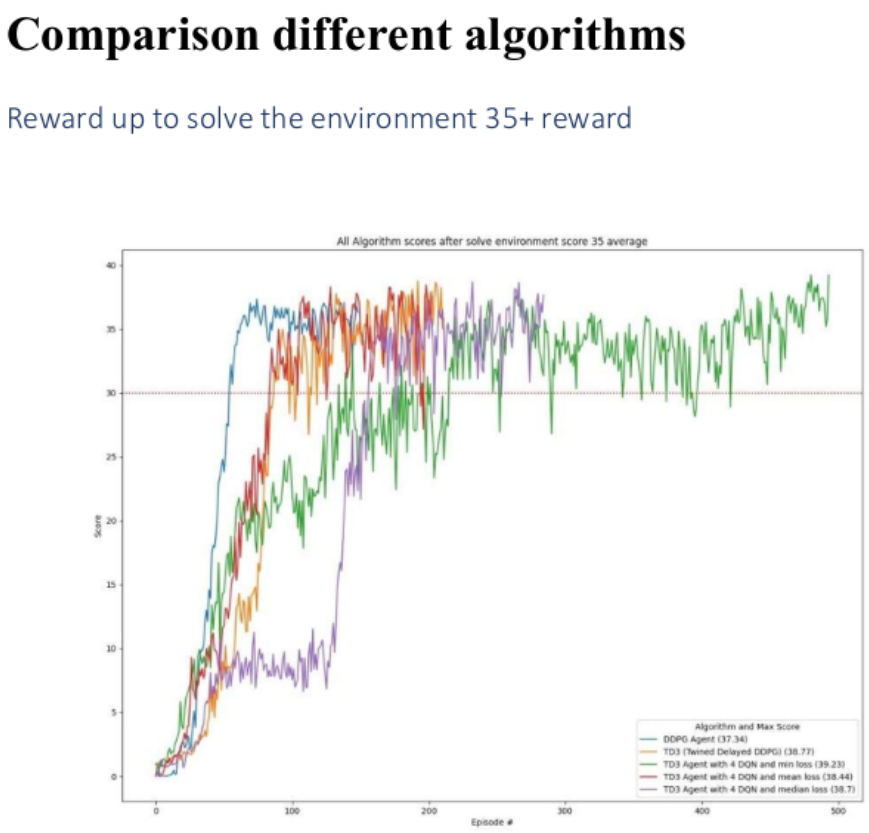



A plot of rewards per episode is included to illustrate that either:

- *[version 1]* the agent receives an average reward (over 100 episodes) of at least +30, or
- *[version 2]* the agent is able to receive an average reward (over 100 episodes, and over all 20 agents) of at least +30.

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

Great work. Rewards plot has been provided for the implemented agent and seems to be great.




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

Great intuition
Also The following posts give an insight into some other reinforcement learning algorithms that can be used to solve the environment.

- Proximal Policy Optimization by Open AI (https://openai.com/blog/openai-baselines-ppo/)
- Introduction to Various Reinforcement Learning Algorithms. Part II (TRPO, PPO) (https://towardsdatascience.com/introduction-to-various-reinforcement-learning-algorithms-part-ii-trpo-ppo-87f2c5919bb9)

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START