UDACITY

‹ Return to Classroom

Collaboration and Competition

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).

Also take a look at the following links which are good starting points and also provide an understanding in laymen terms

- https://towardsdatascience.com/multi-agent-deep-reinforcement-learning-in-15-lines-of-code-using-pettingzoo-e0b963c0820b
- https://towardsdatascience.com/ive-been-thinking-about-multi-agent-reinforcement-learning-marl-and-you-probably-should-be-too-8f1e241606ac
- https://arxiv.org/pdf/1911.10635.pdf
- https://medium.com/@RemiStudios/multi-agent-reinforcement-learning-3f00b561f5f0
- https://www.youtube.com/watch?v=Yd6HNZnqjis

Happy learning ✌️

Training Code

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

Great work in doing what the previous reviewer had asked you to look into. Very well done. Shows your great commitment and confidence in problem solving to the highest extent.

Also All the files are present and are in order. Nicely done. Take a look at the following links for better understanding about the algorithms in depth

- http://www.dcsc.tudelft.nl/~bdeschutter/pub/rep/10_003.pdf
- https://d-nb.info/1005943761/34

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).

Good work providing the details of the project environment. State space, action space, reward function and when the environment is considered solved is specified

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 thoroughly 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.

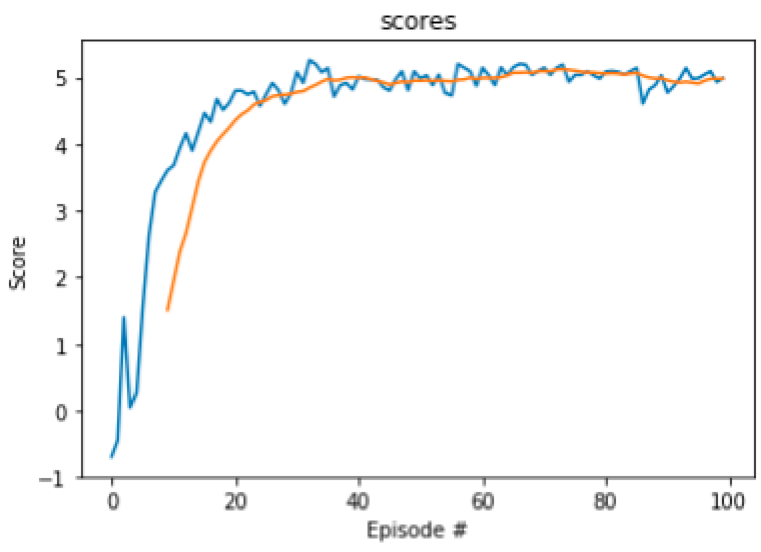
Great work. Description of the learning algorithm, hyperparameters, and network architecture have been provided in great detail in the project report.

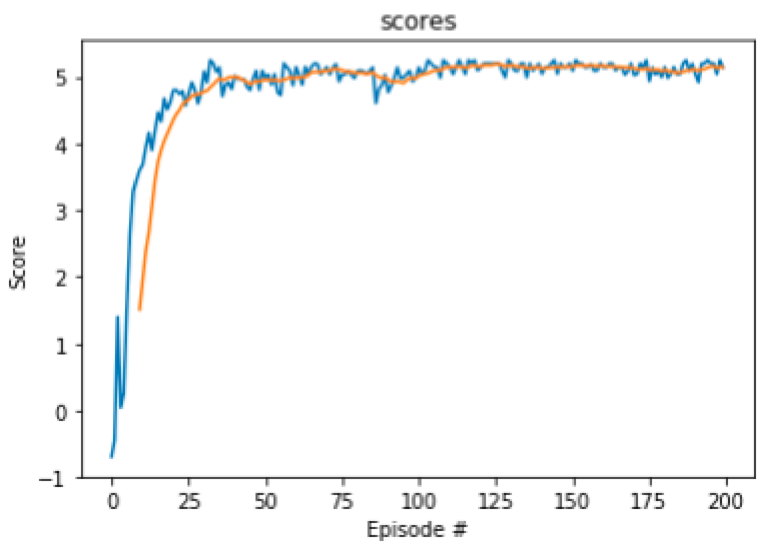
A plot of rewards per episode is included to illustrate that the agents get an average score of +0.5 (over 100 consecutive episodes, after taking the maximum over both agents).

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

Great work in competing the requirements of the previous reviewer. Very well done. The code seems fine now.

```
Episode 100	Average Score: 4.55
Episode 200	Average Score: 5.14
Episode 240	Average Score: 5.15
```





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

Great intuition.. Also take a look at the following link to understand more algorithms in play https://mayankm96.github.io/tutorials/

Yes, as you correctly mentioned about the more advanced versions, Hindsight Experience Replay and Distributed Prioritized Experience Replay, this will definitely help.

DOWNLOAD PROJECT

RETURN TO PATH

Rate this review

START