UDACITY

< Return to Classroom

Navigation

REVISÃO

REVISÃO DE CÓDIGO

HISTORY

Meets Specifications

Dear Udacian,

Great job getting acquainted with the Deep Q Network algorithm and implementing it to successfully solve the Navigation environment. The implementation is pretty good and the agent solves the environment in ~500 episodes. The agent further goes on to achieve the maximum average score of +16.00 in 1046 episodes. The architecture used for the Q network is good in size with two hidden layers of size 64 units each. Good work using `relu` activation in the network. The implementation is quite good. 😄

I would suggest you to go through Deep Reinforcement Learning for Self Driving Car by MIT. You'd get to know more about reinforcement learning algorithms in broader and real-world perspective and, more importantly, how to apply these techniques to real-world problems.

All the best for future endeavors. ✨

Training Code

The repository (or zip file) includes functional, well-documented, and organized code for training the agent.

Awesome

- Good implementation of the agent for Deep Q Network.
- Correct decoupling of the parameters being updated from the ones that are using a target network to produce target values.
- Good implementation of The Epsilon-greedy action selection to encourage exploration.

- Good use of tau parameter to perform soft-update to prevent variance in the process caused by individual batches.
- Good use of the replay memory to store and recall experience tuples.

The code is written in PyTorch and Python 3.

Awesome

The code is written in PyTorch and Python 3.

Lately, PyTorch and TensorFlow happen to be most extensively used frameworks in deep learning. It would be good to get some insight by comparing them, please see the following resources:

- Sebastian Thrun on TensorFlow
- PyTorch vs TensorFlow—spotting the difference
- Tensorflow or PyTorch : The Force is Strong with which One?

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

Awesome

- Saved model weights of the successful agent have been submitted.
- The `checkpoint.pth` file is present in the submission.

README

The GitHub (or zip file) submission includes a `README.md` file in the root of the repository.

Awesome

- Great work documenting the project details and submitting the README file.

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

Awesome

- README file describes the project environment details properly in the `Project Details` section.
- Information about the state and action spaces, and when the environment is considered solved has been provided.

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

Awesome

- Great work providing the all the necessary instructions in the `Getting Started` section to install the dependencies.

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.

Awesome

- Great work providing necessary instructions to run the code in the `How to Run` section.
- All the cells in `Navigation.ipynb` file should be executed to train the agent.

Report

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

Awesome

- Report for the project with all the details of the implementation has been provided 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

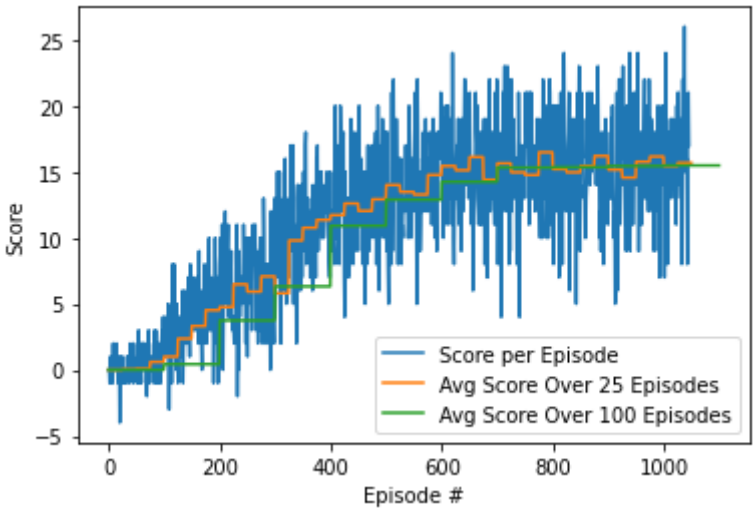
Great work providing the details of the implemented agent. Details of the learning algorithm used, hyper-parameters, and architectural information of the deep learning model have been provided.

- Good decision to choose DQN algorithm for the discrete action space problem.
- Good work including model architecture in the report.
- Good work using two hidden layers in the network with 64 units each.
- Good decision choosing to use `relu` activation.
- Hyperparameters you have used seem to be good.

A plot of rewards per episode is included to illustrate that the agent is able to receive an average reward (over 100 episodes) of at least +13. The submission reports the number of episodes needed to solve the environment.

Awesome

- Discussion for the rewards is provided in the report.
- The rewards plot seems to be good and average score of +13 is achieved in ~500 episodes.



Reinforcement learning algorithms are really hard to make work.
But it is substantial to put efforts in reinforcement learning as it is close to Artificial General Intelligence.
This article is a must read: Deep Reinforcement Learning Doesn't Work Yet.

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

Awesome

Impressive work listing the following concrete future ideas for improving the agent's performance:

- Modifying the architecture
- Solving the pixels based environment
- Double DQN
- Prioritized Experience Replay
- Hyperparameter optimization

Suggestions

As specified in the report, Prioritized Experience Replay should help to improve the performance and significantly reduce the training time also. Using Sum Tree, a special data structure, a fast implementation of PER is possible. It is requested to check this implementation.

It would be very useful to check Improvements in Deep Q Learning: Dueling Double DQN, Prioritized Experience Replay, and fixed Q-targets

It is requested to check the following resources to get familiar with the Rainbow algorithm:

- Rainbow: Combining Improvements in Deep Reinforcement Learning
- Conquering OpenAI Retro Contest 2: Demystifying Rainbow Baseline

Please also check this talk by David Silver, the author of the DQN paper

And, you should definitely try applying these algorithms by taking raw screen pixels as input also. In case you get stuck, definitely check this github repository.

BAIXAR PROJETO

RETORNAR

Avalie esta revisão

COMEÇAR