CSE 4/546: Reinforcement Learning
Spring 2021

Instructor: Alina Vereshchaka

Assignment 2 - Deep Q-Networks

Checkpoint: March 21, Sun, 11:59pm

Due Date: April 4, Sun, 11:59pm

1 Assignment Overview

The goal of the assignment is to work with value function approximation algorithms, to explore OpenAI Gym environments and getting experience working with a project-management tool. In the first part of the project we will implement Deep Q-learning (DQN) following DeepMind's paper that explains how reinforcement learning algorithms can learn to play Atari from raw pixels. In the second part of the project we will implement an improvement to the DQN algorithm. The purpose of this project is to understand the benefits of approximation methods, the role of deep neural networks as well as some of the techniques used in practice to stabilize training and to achieve better performance. We will train our networks on the grid-world and OpenAI gym environments.

Part 1 [40 points] - Implementing DQN & Solving grid-world environment

1.1 Implementing DQN

Implement DQN from scratch following DeepMind's paper ([mnih2015human] and [mnih-atari-2013]). You may use any framework (Keras/Tensorflow/Pytorch).

1.2 Solving grid-world environment

Apply DQN to solve the environment you or your teammate have defined in Assignment 1. You can make slight modifications or improvements to the original environment, if required.

In your report for Part 1:

1. Discuss the benefits of:
    - Using experience replay in DQN and how its size can influence the results
    - Introducing the target network
    - Representing the Q function as $\hat{q}(s, \mathbf{w})$
2. Briefly describe the grid-world environment that you used (e.g. possible actions, states, agent, goal, rewards, etc). You can reuse related parts from your Assignment 1 report.
3. Show and discuss your results after applying your DQN implementation on the environment. Plots should include epsilon decay and the reward per episode.

Part 2 [60 points] - Improving DQN & Solving OpenAI Gym Environments

2.1 Improving vanilla version of DQN

There have been many improvements developed for the vanilla algorithm. In this part we will implement one of improved algorithms that is based on DQN. Modify your DQN code from Part 1.1 to one of the improved versions and apply it to one grid-world environment from Part 1.2 and compare the results.

Possible algorithms to implement include:

- Double DQN
- Dueling DQN
- Prioritized Experience Replay (PER)

2.2 Applying DQN-based algorithms to solve TWO complex environments

Test your DQN algorithm implemented in Part 1 and the improved version on 'CartPole-v1' and a complex environment. You may use your custom made multi-agent environment or any other complex environment that you will use for your Final Project (this has to be approved by the course staff). Compare the results. Describe the environments that you used (e.g. possible actions, states, agent, goal, rewards, etc). Provide reward dynamics (average reward in t-steps). The environment with multiple versions will be considered one environment.

'CartPole-v1' CartPole is one of the classical OpenAI gym environments that often used as a benchmark problem to check the algorithm performance. 'CartPole-v1' environment is considered to be solved if it is getting an average reward of more than 470 points over 100 consecutive episodes during evaluation.

Suggested a second environment to work with:

- OpenAI LunarLander
- OpenAI Atari Breakout
- OpenAI MountainCar
- OpenAI Space Invadors

In your report for Part 2:

1. Discuss the algorithm you implemented.
2. What is the main improvement over the vanilla DQN?
3. Describe 'CartPole-v1' and the complex environment that you used (e.g. possible actions/states, goal, rewards, etc).
4. Compare the performance of both algorithms (DQN & Improved version of DQN) on the same environments (e.g. show one graph with two reward dynamics) and provide your interpretation of the results. Overall three rewards dynamics plots with results from two algorithms applied on:
    - Grid-world environment
    - 'CartPole-v1'
    - OpenAI Gym or Multiagent environment
5. Provide your interpretation of the results. E.g. how the same algorithm behaves on different environments, or how various algorithms behave on the same environment.

Extra Points [max +10 points]

- **Using PyTorch [2 points]**
  PyTorch is becoming a standard framework to use for research related to AI (e.g OpenAI), as well becoming more popular in industry. It also mostly provides a better performance and faster convergence on RL-related tasks. We want to motivate you to develop skills working with this framework. If you choose to complete your assignment using PyTorch, you will receive additional points.
- **Solving Image-based Enviroment [5 points]**
  Use one of the environments with image representation of the state that requires to utilize CCN (Convolution Neural Network) for the state preprocessing (e.g. OpenAI Breakout).
- **Project Management [3 points]**
  Project management includes the application of specialized skills, knowledge, processes and procedures to successfully take an assignment from start to finish. These tools can be used for collaboration, status reporting, time management and many other cases. There is an extensive range of tools available with free tiers available for small team usage. You can choose any of the following:
  - Trello
  - JIRA
  - Asana
  - Basecamp

  Requirements:
  - Create a board with at least these columns: To do, In Progress, Done
  - Divide the project to at least 20 steps/milestones (E.g. Implement DQN, Prepare grid-world env, Generate rewards dynamic graph, Report for Part 1: Q1/Q2/Q2, Submit Checkpoint, Submit Final, etc).
  - Add course instructors as observers or as a team member to your board [avereshcbuffalo.edu and nitinvisbuffalo.edu]
  - The board should be created at least one week before the checkpoint submission and every week there should be some activities

3 Deliverables

Submit your work using UBLearns group in both cases if you work individually or in a team of two. There are two parts in your submission:

3.1 Report

The report should be delivered as a separate pdf file, and it is recommended for you to use the NIPS template to structure your report. You may include comments in the Jupyter Notebook, however you will need to duplicate the results in the separate pdf file. For the final submission, combine the reports for both Part 1 and Part 2 into one file.

3.2 Code

Python is the only code accepted for this project. You can submit the code in Jupyter Notebook or Python script. You can submit multiple files, but they all need to have a clear name. After executing command python main.py in the first level directory or Jupyter Notebook, it should generate all the results and plots you used in your report and should be able to be printed out in a clear manner. Additionally you can submit the trained parameters, so that the grader can fully replicate your results. For the final submission you can combine the code from both parts into one.

4 References

- NIPS Styles (docx, tex)
- Overleaf (LaTex based online document generator) - a free tool for creating professional reports
- GYM environments
- Lecture slides
- Human-level control through deep reinforcement learning
- Prioritized Experience Replay
- Deep Reinforcement Learning with Double Q-learning

5 Checkpoint Submission [Due date: Mar 21]

Complete Part 1 and submit the code and draft report. To submit your work, add your pdf, ipynb/python script to zip file with UBIT *TEAMMATE1_TEAMMATE2_assignment2_checkpoint.zip* (e.g. *avereshc_nitinvis_assignment2_checkpoint.zip* and upload it to UBlearns (Assignments section). Checkpoint will be evaluated after the final submission.

6 Final Submission [Due date: Apr 4]

Add your combined pdf and ipynb/python script for Part 1 and Part 2 to a zip file *TEAMMATE1_TEAMMATE2_assignment2_final.zip* (e.g. *avereshc_nitinvis_assignment2_final.zip*) and upload it to UBlearns using group submission (Assignments section). After the assignment is graded, you may be asked to demonstrate it to the instructor if your results or reasoning in your report are not clear enough.

7 Important Information

This assignment can be completed in groups of two or individually. Teams can not be the same for A2 & A3 assignments. The standing policy of the Department is that all students involved in any academic integrity violation (e.g. plagiarism in any way, shape, or form) will receive an F grade for the course. The catalog describes plagiarism as "Copying or receiving material from any source and submitting that material as one's own, without acknowledging and citing the particular debts to the source, or in any other manner representing the work of another as one's own.". Updating the hyperparameters or modifying the existing code is not part of the assignment's requirements and will result in a zero. Please refer to the UB Academic Integrity Policy.

8 Late Days Policy

You can use up to 5 late days throughout the course toward any assignments' checkpoint or final submission. You don't have to inform the instructor, as the late submission will be tracked in UBlearns. If you work in teams the late days used will be subtracted from both partners. E.g. you have 4 late days and your partner has 3 days left. If you submit one day after the due date, you will have 3 days and your partner will have 2 days left.

9 Important Dates

March 21, Sun 11:59pm - Checkpoint is Due

April 4, Sun, 11:59pm - Assignment 2 is Due