CSE 4/546: Reinforcement Learning
Spring 2022

Instructor: Alina Vereshchaka

Assignment 3 - Actor-Critic

Team Registration: April 15, Fri, 11:59pm

Due Date: May 1, Sun, 11:59pm

1 Assignment Overview

The goal of the assignment is to help you understand the concept of policy gradient algorithms, to implement the actor-critic algorithm and apply it to solve OpenAI gym environments. We will train our networks on various reinforcement learning (RL) environments.

Part I [Total: 60 points] - Implementing Actor-Critic

In this part we will implement an actor-critic algorithm and test it on any simple environment.

1. Implement an actor-critic algorithm. It can be completed in any of the following: Q Actor-Critic, TD Actor-Critic, Advantage Actor-Critic (A2C), Proximal Policy Optimisation, etc. You may use any framework (Keras/Tensorflow/Pytorch).
2. Train your implemented algorithm on any environment. Check "Suggested environments" section.
3. Show and discuss your results after applying your actor-critic implementation on the environment. Plots should include the total reward per episode.
4. Provide the evaluation results. Run your environment for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.

Part II [Total: 40 points] - Solving Complex Environments

In this part, test your actor-critic algorithm implemented in Part I on any other two complex environments.

1. Choose an environment. At least one of the environments has to be among "Suggested environments - Part II". The environment with multiple versions is considered as one environment.
2. Apply your actor-critic algorithm to solve it. You can adjust the neural network structure or hyperparameters from your base implementation.
3. Show and discuss your results after applying the actor-critic implementation on the environment. Plots should include the total reward per episode.
4. Provide the evaluation results. Run your environment for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.
5. Go to Step 1. Provide the results for TWO environments.

Suggested environments

Part I

- Your grid world from A1 or A2,
- OpenAI CartPole
- OpenAI Acrobot
- OpenAI Mountain Car
- OpenAI Pendulum

Part II

- Any multi-agent environment
- OpenAI LunarLander
- OpenAI BipedalWalker
- OpenAI HandManipulateBlock
- OpenAI Ant
- RL Bench (link)
- Any other complex environment that you will use for your Final Project

In your report:

1. Discuss the algorithm you implemented.
2. What is the main difference between the actor-critic and value based approximation algorithms?
3. Briefly describe THREE environments that you used (e.g. possible actions, states, agent, goal, rewards, etc). You can reuse related parts from your Assignment 2 report.
4. Show and discuss your results after training your Actor-Critic agent on each environment. Plots should include the reward per episode for THREE environments. Compare how the same algorithm behaves on different environments while training.
5. Provide the evaluation results for each environments that you used. Run your environments for at least 10 episodes, where the agent chooses only greedy actions from the learnt policy. Plot should include the total reward per episode.
6. If you are working in a team of two people, we expect equal contribution for the assignment. Provide contribution summary by each team member.

Extra Points [max +12 points]

- **Implement more complex actor-critic [7 points]**
  Extend your actor-critic algorithm to a more advanced version, e.g. TRPO/DDPG/TD3/SAC. Compare the results after applying it to the same THREE environments used in the assignment. Provide three rewards dynamic plots for each environment with the results of two algorithms: actor-critic and improved version. Disscuss the results.

- **Solve Image-based Environment [5 points]**
  Use one of the environments with image representation of the state that requires a utilization of CNN (Convolution Neural Network) for the state preprocessing (e.g. OpenAI Breakout).

2 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:

2.1 Report

Combine the reports for two parts into one pdf. It is recommended for you to use the NeurIPS 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. Follow the name for your report as

$UBIT\_TEAMMATE1\_TEAMMATE2\_assignment3\_report.pdf$
(e.g. $avereshc\_nitinvis\_assignment3\_report.pdf$)

2.2 Code

Python is the only code accepted for this project. Submit the code in Jupyter Notebook with the saved results. You can submit multiple files, but they all need to have a clear name. After executing commands Jupyter Notebook, it should generate all the results and plots that were used in your report and should be able to be printed out in a clear manner. Additionally you can submit the trained parameters, so your results can be fully replicated.

For the final submission, submit your code as Jupyter Notebooks named as

$UBIT\_TEAMMATE1\_TEAMMATE2\_assignment3.ipynb$
(e.g. $avereshc\_nitinvis\_assignment3.ipynb$)

or

$UBIT\_TEAMMATE1\_TEAMMATE2\_assignment3\_partI.ipynb$

$UBIT\_TEAMMATE1\_TEAMMATE2\_assignment3\_partII.ipynb$

3 References

- NeurIPS Styles (docx, tex)
- Overleaf (LaTex based online document generator) - a free tool for creating professional reports
- GYM environments
- Lecture slides

4 ASSIGNMENT STEPS

1. Register your team (Due date: April 15, Fri)

- You may work individually or in a team of up to 2 people. The evaluation will be the same for a team of any size. **(Team members should be different for Assignment 2 and 3)**
- Register your team at UBLearns (UBlearns > Tools > Groups). In case you joined the wrong group, make a private post on piazza.

2. Submit final results (Due date: May 1, Sun)

- Fully complete all parts of the assignment
- In order to certify that you followed Academic Integrity policy while completing the assignment, include the following statement as a comment block at the beginning of your code. In case you submit multiple files, add this statement at the beginning of each code file submitted:

  "I/we certify that the code and data in this assignment were generated independently, using only the tools and resources defined in the course and that I/we did not receive any external help, coaching or contributions during the production of this work."

  Submissions without the academic integrity statement will not be evaluated and will receive a 0.
- Add your combined pdf and ipynb for Part 1 and Part 2 to a zip file $TEAMMATE1\_TEAMMATE2\_assignment3\_final.zip$ (e.g. $avereshc\_nitinvis\_assignment3\_final.zip$)
- Submit at UBLearns > Assignments
- The code of your implementations should be written in Python. You can submit multiple files, but they all need to be labeled clearly.
- Your Jupyter notebook should be saved with the results
- Include all the references that have been used to complete the assignment
- If you are working in a team of two, we expect equal contribution for the assignment. Each team member is expected to make a code-related contribution. Provide a contribution summary by each team member in a form of a table below. If the contribution is highly skewed, then the scores of the team members may be scaled w.r.t the contribution.

Team Member	Assignment Part	Contribution (%)

5 Academic Integrity

This assignment can be completed individually or in a team of two students. 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.

IMPORTANT NOTE

In order to certify that you followed Academic Integrity policy while completing the assignment, please include the following statement as a comment block at the beginning of your code. In case you submit multiple files, add this statement at the beginning of each code file submitted:

"I/we certify that the code and data in this assignment were generated independently, using only the tools and resources defined in the course and that I/we did not receive any external help, coaching or contributions during the production of this work."

Submissions without the academic integrity statement will not be evaluated and will receive a 0.

6 Important Information

This assignment can be completed in groups of two or individually. Teams can not be the same for A2 & A3 assignments.

7 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.

8 Important Dates

April 15, Fri, 11:59pm - Register your team (UBlearns > Tools > Groups)

May 1, Sun, 11:59pm - Assignment 3 is Due