CSE 4/546: Reinforcement Learning
Spring 2021

Instructor: Alina Vereshchaka

Final Course Project

Multiple deadlines, please see below

Description

The goal of the Final Course Project is to explore advanced methods and/or applications in reinforcement learning. You will be expected to prepare a proposal, checkpoint, final submission, and presentation. All projects should evaluate novel ideas that pertain to deep RL or its applications and must involve reinforcement learning algorithms. You are encouraged to use your ongoing research work as a project in this course, provided that this work relates to deep reinforcement learning. You may discuss the topic of your final project with course staff by private message in Piazza, or during OH. If you are not sure about the topic, we encourage you to speak with us.
There are few directions suggested, please check the end of the description for more details.

1. Registering your team (March 21)

You may work individually or in a team of up to 3 people.
Register your team at UBLearns (UBlearns > Groups)

2. Project Management Tool Setup (March 21)

If you are working in a team of two or three people, set up a project management (PM) tool. Choose from Trello, JIRA, Basecamp

1. Create a board with at least these columns: To do, In Progress, Done
2. Divide the project to at least 20 steps/milestones (E.g. Explore the topics, Create a proposal, Implement a basic version, Implement advanced algo, Submit Checkpoint, Submit Final, Prepare for the presentation, etc).

3. Add course instructors as observers or as team members to your board [avereshc@buffalo.edu and nitinvis@buffalo.edu]
4. There should be some activities every week

3. Submitting the proposal (March 28)

The project proposal should be a one page single-spaced extended abstract motivating and outlining the project you plan to complete. Submit at **UBLearns > Assignments.** You proposal should have the following structure:

1. Topic
2. Objective. Explain the objective of the project and why that objective is relevant and important.
3. Related Work. Briefly review the most relevant prior work, and highlight where those works fall short of meeting the objectives described above.
4. Technical Outline. Explain your approach at a high-level, making clear the novel technical contribution. What environment and algorithm you are planning to use.
5. List any reference you are planning to use.

4. Submitting the checkpoint (April 11)

Submit the initial results of your model (e.g. built baseline model and prepared a pipeline for training). You need to submit the code and a draft report with prior results. Submit at **UBLearns > Assignments.**

5. Submitting the final project (April 30)

- 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.
- All project files should be packed in a ZIP file named: **TEAMMATE#1_UBIT_TEAMMATE#2_UBIT_final_project.zip** (e.g. **avereshc_nitinvis_final_project.zip**).
- Your Jupyter notebook should be saved with the results. If you are submitting python scripts, after extracting the ZIP file and executing command python main.py in the first level directory, all the generated results and plots you used in

your report should appear printed out in a clear manner.
- A report can be submitted in combination with the presentation. Thus if you discuss all the technical implementation of your project and provide the results, a separate report will not be necessary.
- Include all the references that have been used to complete the project.

6. Presenting your work (May 3)

Present your work during the presentation day. Registration slots will be available prior to dates. The whole team should equally present the work. Your presentation should represent the work you will submit. Submit the final presentation by **May 3, 11:59pm**.

Presentation details

Length: 15 mins + follow-up questions

Suggested Templates: UB branded templates or UB CSE template

Suggested presentation structure:

- Project Title / Team's Name / Course / Date [1 slide]
- Project Description [1 slide]
- Background [max 2 slides]
- Implementation [max 2 pages]
- Demo (if available)
- Results (Graphs & Any Visuals) [max 4 slides]
- Key Observations / Summary [1 slide]
- Thank you Page [1 slide]

Extra Points [max +10 points]

Participate in Weekly Scrum Meetings [5 points]

Weekly Scrum (Friday 12pm-12:30pm) is a regular group meeting led by Nitin Kulkarni where at least one team member gives updates about the project progress. This is not evaluated and you can also consider these meetings as an opportunity to get feedback on your current results.

Your team has to take part at least 4 times to be eligible for the bonus. We

encourage all team members to join scrum meetings.

CSE Demo Days [5 points]

If you get interesting results, we would encourage you to share your project with the public in terms of participating in the CSE Demo Days. CSE Demo Days is a semester event, where you can highlight your project results.

Send us your prior results around April 18. Selected teams will have to prepare a poster.

Important Information

This project can be done in a team of up to three people. The standing policy of the Department is that all students involved in an academic integrity violation (e.g. plagiarism in any way, shape, or form) will receive an F grade for the course. Refer to the Academic Integrity website for more information.

Late Submission Policy

For the final project everyone will be provided with 3 late days per team, no matter whether you are working individually or with teammates. These late days can be applied only to final project-related due dates. Be aware that some of the project components have hard deadlines. These cannot be combined with your individual late days. You do not have to inform the instructor, as the late submission will be tracked in UBlearns.

Important Dates

March 21, Sun, 11:59pm - Register your team (UBLearns > Group) and setup PM tool
March 28, Sun, 11:59pm - Abstract is Due
April 11, Sun, 11:59pm - Checkpoint is Due
April 30, Fri, 11:59pm - Final Submission Deadline
May 3, Mon - Presentation **[hard deadline, no late days can be applied]**

Final Project Directions

Below is a list of possible directions for your final project, you need to choose one.

MULTIAGENT REINFORCEMENT LEARNING

For this option, you are expected to build and solve multiagent tasks (including but not limited to agents communications, transportation problems, multi-agent cooperation, etc) - all might potentially lead to a research project.

Steps include:

1. Building a small multiagent environment with at least two agents (this can be an extension of your work at Assignment 1).
2. Solving the environment using any tabular methods.

   [Steps 1 & 2 can be used for checkpoint submission]

3. Solving the environment using any deep RL methods (DQN, DDQN, AC, A2C, DDPG, TRPO, PPO, etc) and comparing the results.
4. Applying the algorithm from Part 3 to solve any of the existing MARL problems. E.g. Open AI Particle Environment, Hide&Seek

References:

- A collection of papers/books/etc [github]
- Multi-Agent Particle Environment (often used for baseline) [github]
- A collection of well-defined MA Gym-like env (good reference) [github]
- Hide and Seek [blog post, github]

Suggestions:

1. Go with collaborative multi-agent systems, it is easier to scale and has more real-world reference. Possibly you can think about a mixed cooperative-competitive, e.g. Hide and Seek example.
2. While building your env, think about some real world problem, that you can solve using MARL settings. It is good if the problem you formulate is a socially important one (e.g. mitigating natural disasters, transportation, rescuing people, etc), so it can be later converted to the research project.

RC CARS

For this option, you are expected to set up the simulator and train the cars in the simulator. These steps may require prior knowledge in robotics or autonomous vehicles. Since the simulator is based on Amazon AWS, you need to setup a local host to train the models.

Steps include:

1. Install & explore the DeepRacer RC cars simulator
2. Complete the Udacity DeepRacer course
3. Check existing solutions & current rewards functions

   [Steps 1 - 3 can be used for checkpoint submission]

4. Apply other RL methods to teach the car to drive in the simulator

References:

- Udacity AWS DeepRacer Course [free]
- How to run DeepRacer locally [medium]

PARALLEL REINFORCEMENT LEARNING

Parallel processing is a good skill that can be applied not just to RL problems, but also to many other applications. During the course we will cover a few algorithms that introduce parallel training (e.g. A3C). For the final project you are expected to implement an already existing algorithm and apply the parallel training to some deep RL algorithms and compare the efficiency.

EXPLORING DEEP RL ALGORITHMS

For this option, you are expected to research and apply recent advances in RL. This may include solving any of the following environments using deep RL algorithms.

Possible environments include:

- MALMO (platform built on top of Minecraft) [github]

- Robotics or MuJoCu by OpenAI [robotics, mujoco, blog post]
- Atari by OpenAI [details]

Steps include:

1. Set up the environment
2. Explore the existing baseline methods applied to solve it & Run it

   [Steps 1 & 2 can be used for checkpoint submission]

3. Apply a few other deep RL to improve the results. You can use your Assignments 2 & 3 implementations as a baseline code for comparison.

Some advanced directions to consider:

1. Random Network Distillation (RND) [blog post, paper]
2. Offline reinforcement learning [blog post, tutorial]
3. Inverse Reinforcement Learning

Propose you own topic

You may also come to us with your topic proposal! We understand that it might end up to be pretty challenging, if you find out you are completely stacked, you are welcome to discuss your possible switching to any other directions. Although you will follow your own direction, the general requirements for the first checkpoint should include the following:

1. Build the environment or use the existing environment
2. Explore the already existing solutions and replicate it, you will later use it as a baseline to compare your solution.

Please talk to the course instructors to ensure the project you have in mind is feasible.