TASK 4

Implementing a policy gradient algorithm.

You are in the group Randomguess consisting of pingli (pingli@student.ethz.ch (mailto://pingli@student.ethz.ch)), umbacht (umbacht@student.ethz.ch (mailto://umbacht@student.ethz.ch)) and yucsun (yucsun@student.ethz.ch (mailto://yucsun@student.ethz.ch)).

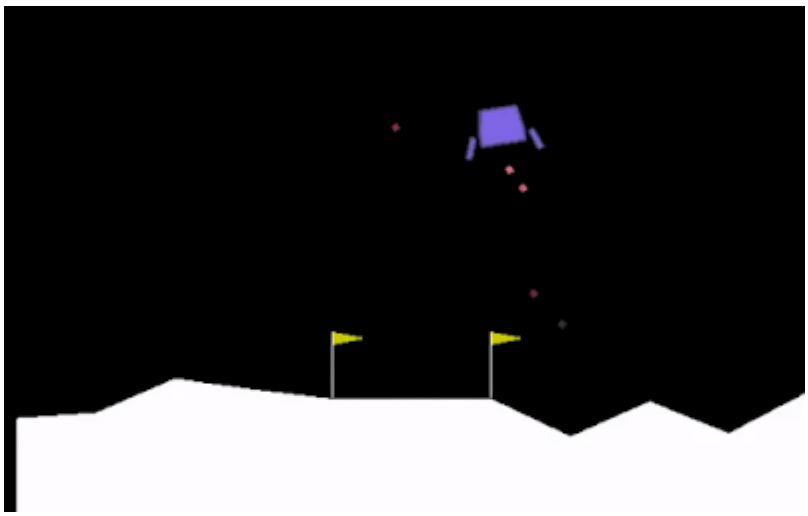
1. READ THE TASK DESCRIPTION

2. SUBMIT SOLUTIONS

3. HAND IN FINAL SOLUTION

1. TASK DESCRIPTION

In this project you will implement a reinforcement learning (RL) algorithm for playing a variant of a game called “lunar lander”.



In the lunar lander environment, a *lander* must be controlled to land between two flags on the ground. When controlling the lander, the goal is to successfully land upright on its two legs, between the flags and in a limited timeframe, whilst expending minimal fuel. Your task is to implement an algorithm that, by practicing on a simulator, learns a control policy for the lander.

The `.pdf` of the handout contains the following additional sections. Environment and Scoring Details will give a more thorough description of the Lunar Lander environment, along with the requirements for passing the baseline. Solution Details will guide you through a set of minimal modifications to `solution.py` for passing the task baseline. We also provide an appendix on policy gradients which will briefly review the lecture material and give you the additional information needed for completing the TODOs in the code for this task.

SUBMISSION WORKFLOW

1. Install and start Docker (https://www.docker.com/get-started). Understanding how Docker works and how to use it is beyond the scope of the project. Nevertheless, if you are interested, you could read about Docker's use cases (https://www.docker.com/use-cases).
2. Download handout (/static/task4_handout.zip). Alternatively you can use the handout for Macbook M1 devices (/static/task4_handout_M1.zip). See below for more details.
3. The handout contains the solution template `solution.py`. You should write your code `# TODO: enter your code here` marker in the solution template. You can make further changes provided that the interface for the `agent` class remains the same.
4. You should use Python 3.8.5. You are free to use any other libraries that are not already imported in the solution template. Important: please make sure that you list these additional libraries together with their versions in the `requirements.txt` or `env.yaml` file provided in the handout.
5. Once you have implemented your solution, run the checker in Docker:
    - if you are using Linux or MacOS (without Apple silicon), run `bash runner.sh`. On some operating systems, you might need to run `sudo bash runner.sh` if you see a Docker permission denied error. Docker might by default restrict how much memory your solution may use. Running over the memory limit will result in docker writing "Killed" to the terminal. If you encounter out-of-memory issues you can increase the limits as described in the Docker Desktop for Mac user manual (https://docs.docker.com/desktop/mac/). Running over the memory limit will result in docker writing "Killed" to the terminal.
    - Note that some required Python packages do not support ARM-based MacBooks (M1 or M2) yet. Above you can see we have a handout specifically for Macbook M1 users. We expect the handout to also work on M2 machines but we have not tested it on an M2. M2 users might want to stick with using a virtual machine or the euler cluster with the regular handout. The M1 handout also requires Docker. The workflow with an M1 and the M1 handout is identical to the workflow for an Intel Mac (paragraph above) using the regular handout.
    - if you are using Windows, open a PowerShell, change the directory to the handout folder and run `docker build --tag task4 .; docker run --rm -v "$(pwd):/results" task4`
    - If you are having trouble running your solution using docker locally, consider using the ETH Euler cluster to run your solution. Please follow the guide specified by *euler-guide.md* in the handout. The setup time of using the cluster means that this option is only worth doing if you really can't run your solution locally. Note that on euler you will not be able to generate a video of your agent solving lunarlander, unlike on other systems. This video is purely aesthetic so this will not be an issue.
6. There is a known issue where the Docker container gets killed during training or evaluation of the model due to running out of memory. If this happens, you need to

increase the RAM available to your Docker instance. We successfully ran everything with 8 GB of RAM.

7. If the checker fails, it will display an appropriate error message. If the check was successful, then a file called `results_check.byte` will be generated. You should upload this file together with your source code to the project server. The checker will also generate a `.mp4` file giving a video of one of your evaluation episodes, which does not need to be submitted.
8. We limit submissions to the server to 40 per team, with at most 20 in a 24 hour period.

GRADING

Some tasks have a public and private score. This task has **only a public score**. Your algorithm will make predictions on a held out test set, or (for tasks 3 and 4) obtain a single score for its interactions with the environment. When handing in the task, you need to select which of your submissions will get graded and provide a short description of your approach. This has to be done **individually by each member** of the team. We will then compare your selected submission to our baseline. This project task is graded with either **pass (6.0) or fail (2.0)**. To pass the project, you need to achieve a better score than the baseline. In addition, for the pass/fail decision, we consider the code and the description of your solution that you submitted. We emphasize that the public score leaderboard is just for fun: the scores of other teams will not effect the baseline or your own grade. The following **non-binding** guidance provides you with an idea on what is expected to pass the project: If you hand in a properly-written description, your source code is runnable and reproduces your predictions, and your submission performs better than the baseline, you can expect to have passed the assignment.

⚠ Make sure that you properly hand in the task, otherwise you may obtain zero points for this task.

FREQUENTLY ASKED QUESTIONS

⊙ WHICH PROGRAMMING LANGUAGE AM I SUPPOSED TO USE? WHAT TOOLS AM I ALLOWED TO USE?

You are free to choose any programming language and use any software library. However, **we strongly encourage you to use Python**. You can use publicly available code, but you should specify the source as a comment in your code.

⊙ AM I ALLOWED TO USE MODELS THAT WERE NOT TAUGHT IN THE CLASS?

Yes. Nevertheless, the baselines were designed to be solvable based on the material taught in the class up to the second week of each task.

⊙ IN WHAT FORMAT SHOULD I SUBMIT THE CODE?

You can submit it as a single file (main.py, etc.; you can compress multiple files into a .zip) having max. size of 1 MB. If you submit a zip, please make sure to name your main file as *main.py* (possibly with other extension corresponding to your chosen programming language).

⊙ WILL YOU CHECK / RUN MY CODE?

We will check your code and compare it with other submissions. We also reserve the right to run your code. Please make sure that your code is runnable and your predictions are reproducible (fix the random seeds, etc.). Provide a readme if necessary (e.g., for installing additional libraries).

⊙ SHOULD I INCLUDE THE DATA IN THE SUBMISSION?

No. You can assume the data will be available under the path that you specify in your code.

⊙ CAN YOU HELP ME SOLVE THE TASK? CAN YOU GIVE ME A HINT?

As the tasks are a graded part of the class, **we cannot help you solve them**. However, feel free to ask general questions about the course material during or after the exercise sessions.

⊙ CAN YOU GIVE ME A DEADLINE EXTENSION?

> ⚠ We do not grant any deadline extensions!

⊙ CAN I POST ON MOODLE AS SOON AS HAVE A QUESTION?

This is highly discouraged. Remember that collaboration with other teams is prohibited. Instead,

- Read the details of the task thoroughly.
- Review the frequently asked questions.
- If there is another team that solved the task, spend more time thinking.
- Discuss it with your team-mates.

⊙ WHEN WILL I RECEIVE THE PRIVATE SCORES? AND THE PROJECT GRADES?

We will publish the private scores, and corresponding grades before the at exam the latest.