DRL Homework 2

Topics

- DQN
- Rainbow
  - Double Q-learning
  - Prioritized Replay
  - Dueling Network
  - Multi-step Learning
  - Distributional RL
  - Noisy Nets

Structure

Follow the "hw.ipynb" notebook for instructions:

You can run the **optional** test script that covers replay buffers from the "test" directory using:

```
python -m unittest test_replaybuffer.py
```

Installation

To start your homework, you need to install requirements. We recommend that you use conda environment for this homework.

```
conda create -n rlhw2 python=3.7
conda activate rlhw2
```

If you are going to use GPU, install Pytorch using the link and remove it from requirements.

You can install the requirements with the following commands in the homework directory:

```
conda install -c conda-forge swig
conda install nodejs
pip install -r requirements.txt
python -m ipykernel install --user --name=rlhw2
```

Then you need to install the homework package. You can install the package with the following command: (Make sure that you are at the homework directory.)

```
pip install -e .
```

This command will install the homework package in development mode so that the installation location will be the current directory.

Docker

You can also use docker to work on your homework. Build a docker image from the homework directory using the following command:

```
docker build -t rlhw2 .
```

You may need to install docker first if you don't have it already.

After building a container we need to mount the homework directory at your local computer to the container we want to run. Note that the container will install necessary python packages in build.

You can run the container using the command below as long as your current directory is the homework directory:

```
sudo docker run -it --rm -p 8889:8889 -v $PWD:/hw2 rlhw2
```

This way you can connect the container at `localhost:8889` in your browser. The container will automatically run Jupyter-Notebook. Note that, although we are using docker, changes are made in your local directory since we mounted it.

You can also use it interactively by simply running:

```
sudo docker run -it --rm -p 8889:8889 -v $PWD:/hw2 rlhw2 /bin/bash
```

Docker with GPU + CUDA Support

The Docker image can be built by typing in the same command:

```
docker build -t rlhw2 .
```

If you would like to use the container with GPU and CUDA support, you first need to install the NVIDIA Container Toolkit on your host computer via (you may want to copy and paste this from Readme.md):

```
curl -fsSL https://nvidia.github.io/libnvidia-container/gpgkey | sudo gpg
--dearmor -o /usr/share/keyrings/nvidia-container-toolkit-keyring.gpg \
  && curl -s -L https://nvidia.github.io/libnvidia-
container/stable/deb/nvidia-container-toolkit.list | \
    sed 's#deb https://#deb [signed-by=/usr/share/keyrings/nvidia-
```

```
container-toolkit-keyring.gpg] https://#g' | \
    sudo tee /etc/apt/sources.list.d/nvidia-container-toolkit.list

sudo apt-get update

sudo apt-get install -y nvidia-container-toolkit
```

After installing the NVIDIA Container Toolkit, Docker service should be restarted by typing the following command:

```
sudo service docker restart
```

Running the container with the GPU support can be performed with adding the `--gpus all` flag to the Docker run command as in the following:

```
sudo docker run -it --rm --gpus all -p 8889:8889 -v $PWD:/hw2 rlhw2
```

```
sudo docker run -it --rm --gpus all -p 8889:8889 -v $PWD:/hw2 rlhw2
/bin/bash
```

Submission

You need to submit this repository after you filled it (and additional files that you used if there happens to be any). You also need to fill "logs" directory with the results of your experiments as instructed in the Jupyter notebook (progress.csv files that you used in your plots). Please only include the valid and necessary log files in your final submission. Submissions are done via Ninova until the submission deadline. For the atari model parameters, you should put a google drive link in the Jupyter notebook.

Evaluation

- Implementations 50%
  - DQN (25%)
  - RAINBOW (75%)
    - Prioritized Replay 20%
    - Distributional RL 20%
    - Noisy Nets 15%
    - Multi-step learning 10%
    - Dueling Networks 5%
    - Double Q-learning 5%
- Experiments 50%
  - LunarLander (75%)
  - Pong (25%)

Related Readings

- DQN
- Double Q-learning
- Prioritized Replay
- Dueling Network
- Multi-step Learning - Richard S. Sutton and Andrew G. Barto Chapter 7
- Distributional RL
- Noisy Nets
- Rainbow

Contact

TA: Kubilay Kağan Kömürcü kubilaykagankomurcu@gmail.com

TA: Caner Özer ozerc@itu.edu.tr

Author

TA: Tolga Ok