Self-Driving Cars

Exercise 0 - Introduction

Katrin Renz
Autonomous Vision Group
University of Tübingen

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

ellis
European Laboratory for Learning and Intelligent Systems

Outline

- Setup for Exercises
- PyTorch
- Cluster
- OpenAI Gym
- Exercise 0

Setup for Exercises

Exercise organization

- Exercises are offered **every 2 weeks, 6 assignments in total**
- Deepen understanding with **coding challenges** and **pen & paper tasks**
- Handed out via the website and introduced during the live sessions
- Can be conducted in groups **up to 2 students** (find partner via ILIAS forum)
- All lectures and exercises are **relevant** for the exam

.

Exercise organization

- We upload one folder per exercise with the tasks and code (for the challenge)
- Excercise folder can be accessed over the link on the course website
- Pen & Paper: we upload the solutions around one week before the final Q&A session of the excercise.
- Challenge: you hand in your final agent and your code and we provide the leaderboard with the final scores. Please be aware of the **submission deadlines**.
- If you have any questions, please ask at the **forum** on ILIAS.
- TA Email IDs: `katrin.renz@uni-tuebingen.de`, `joo-ho.lee@uni-tuebingen.de`, `apratim.bhattacharyya@uni-tuebingen.de`.

.

Important Rule

- You are not allowed to share results and code of the coding challenge or make it public (e.g., GitHub).

.

Cluster

TCML cluster

- Most parts of the challenge can be done without a GPU. You can use your local machine/ laptop.
- For the reinforcement learning part using a GPU makes sense.
- You are eligible to use the Training Center for Machine Learning (TCML) cluster for exercises in this lecture.
- We will create accounts for you on the cluster. Each group shares an account. **You are not supposed to apply for the account by yourself.**
- Cluster has a master node to launch jobs, and 40 compute nodes to execute jobs.
- Compute nodes are allocated based on a queuing system, so please **start the assignments early** if you want to use this resource.

TCML cluster

- Login to master:
  `ssh username@tcml-master01.uni-tuebingen.de`
- To access compute nodes: create an `.sbatch` file
- Please find more information/instructions about the cluster below:
  `https://docs.google.com/document/d/1AgtLy28VVZaPe79TwOb9jjC4F1KVzffb8y1vZoURZE8/edit?usp=sharing`.

Some useful tools

- Consider using tmux/screen for running your jobs.
  (Start a tmux session: ”tmux new -s mysession”)
- slurm_gpustat: pip install –user slurm-gpustat

PyTorch

PyTorch

- What is PyTorch?
  A Python-based scientific computing package for Deep Learning.
- Why PyTorch?
  Beginner friendly, well documented, good for fast development.
- How to install?
  `https://pytorch.org/get-started/locally/`

References

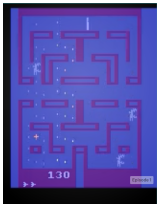
- PyTorch official tutorials: `https://pytorch.org/tutorials/`
- Stanford Course on Deep Learning for Computer Vision: `http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture08.pdf`
- NTU Machine Learning Course: `https://www.slideshare.net/lymanblueLin/pytorch-tutorial-for-ntu-machine-learing-course-2017`
- PyTorch tutorial with code examples: `https://github.com/MorvanZhou/PyTorch-Tutorial`

OpenAI Gym

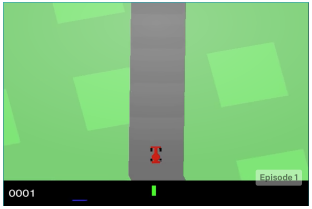
OpenAI Gym

- What is OpenAI Gym?
  A python based toolkit for developing and comparing RL algorithms.
- Why OpenAI Gym?
  Standardization of environments/benchmarks for RL algorithms.
- How to install?
  We provide a custom package with instructions.

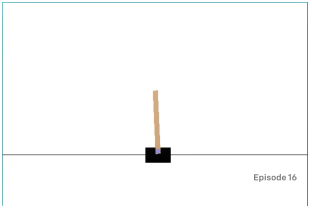
Examples



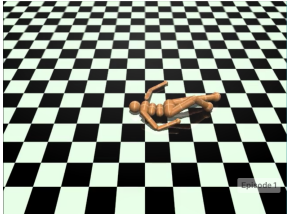
(a) Atari



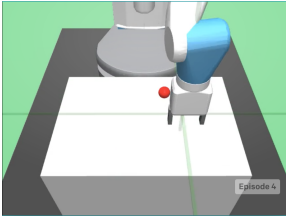
(b) Box2D



(c) Control



(d) MuJoCo



(e) Robotics

(f) Grid Worlds

Work flow

Create an Environment

- Each gym environment has a unique name
- To create an environment from the name use the

```
env = gym.make(env_name)
```

- For example, to create a CarRacing environment:

```
env = gym.make('CarRacing-v0')
```

Work flow

Initialize State

- Used to reinitialize a new episode
- Returns the initial state

```
init_state = env.reset()
```

Work flow

Take an action

- Performs the specified action and returns the resulting state
- The main method your agent interacts with

```
step(action) -> (next_state,
        reward,
        is_terminal,
        debug_info)
```

Work flow

Take an observation: Render

- Optional method
- Used to display the state of your environment
- Useful for debugging and qualitatively comparing different agent policies

```
env.render()
```

Car Racing

- Randomly generated tracks

Car Racing

- `action_space`: three continous values, including steer, gas, brake

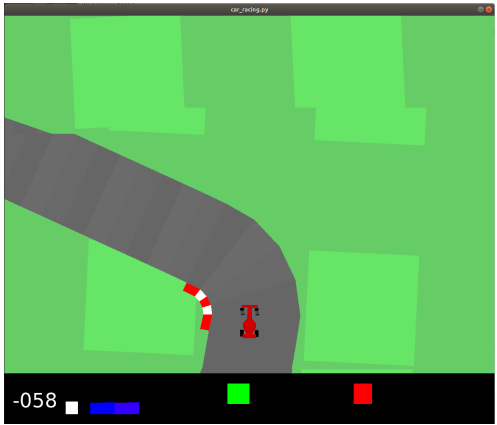
```
self.action_space = spaces.Box( low=np.array([-1,0,0]),
                                high=np.array([+1,+1,+1]),
                                dtype=np.float32)
```

- `observation_space`: color image

```
self.observation_space = spaces.Box(low=0, high=255,
                                    shape=(STATE_H, STATE_W, 3),
                                    dtype=np.uint8)
```

- `reward`: $R = N_{visited\_tile} * \frac{1000}{N_{all\_tile}} - N_{frame} * 0.1$

Car Racing



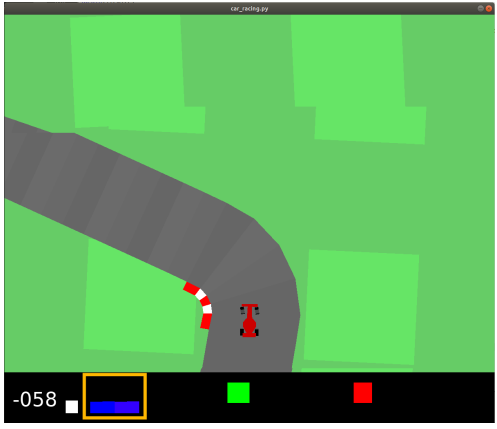
Car Racing

- Reward

Car Racing

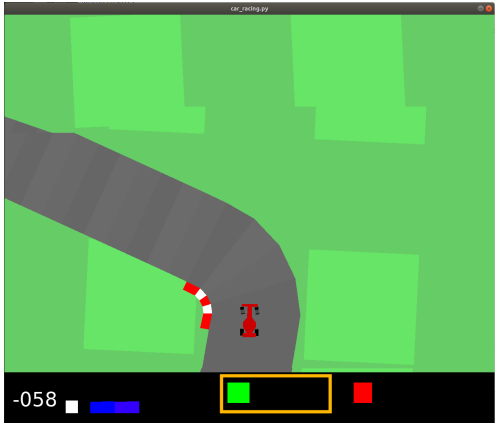
- Reward
- Car speed

Car Racing



- Reward
- Car speed
- Wheel speed

Car Racing



- Reward
- Car speed
- Wheel speed
- Joint angle

Car Racing

- Reward
- Car speed
- Wheel speed
- Joint angle
- Angular Velocity

Car Racing

- We will create a leaderboard for the exercises
- Evaluation metrics:
  - $R = N_{visited\_tile} * \frac{1000}{N_{all\_tile}}$ given a fixed N_{frame}
  - $R = R - 100$ if the car get too far away from the track
- Submit your code and we will evaluate it on our server

References

- Official document: `https://gym.openai.com/docs`
- Source code: `https://github.com/openai/gym`
- `https://katefvision.github.io/10703_openai_gym_recitation.pdf`

Exercise 0

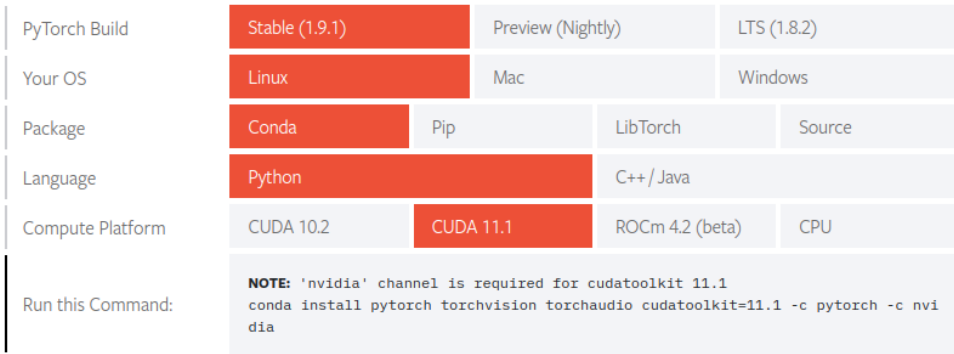
Exercise 0

- Goal: setting up the environment so that you can start with the challenge next week
- Creating a conda environment on your local machine.
- Getting the TCML cluster account if needed and try to access it.
- Familiarize with the car racing environment.
- Create teams (2 people) for the challenge.

Exercise 0

Install PyTorch locally

- Recommended to install with Anaconda



Exercise 0

Install OpenAI Gym locally

- Python 3.8
- Unzip `sdc_gym.zip` and enter the folder
- Install the box2d package
  pip3 install -e '.[box2d]'
- Please use the code we provided as there are some modifications compared to the official version.

Exercise 0

Work on Cluster

- Download the Singularity image and copy it to your home directories on the cluster: `https://owncloud.tuebingen.mpg.de/index.php/s/CbGdQrCfcpP4EFA/download`
  It is an environment that contains everything you need for the exercises of this lecture such as PyTorch, OpenAI Gym.
- Create .sbatch file to submit a job
- Run the code under the Singularity environment:

```
singularity exec --nv ~/sdc_gym.simg python your_python_file.py
```

Questions?