Prof. Frank Hutter, Prof. Joschka Bödecker
Dr. Marius Lindauer
Gabriel Kalweit

Reinforcement Learning

Project

Submit until **Wednesday, February 21 at 11:00am**

The project is based on the *Pendulum Swing up* problem. You have to decide on an algorithm and evaluate its performance. This time, we do not provide any code base or hyperparameter setting. **In the first part of the exam, we are going to discuss your results based on a short presentation you submit beforehand. The second part of the exam will be about the rest of the lecture and exercises. Aim to optimize your results considering the return, learning stability and speed (in terms of episodes/samples).** Please push your slides as a PDF and your code to subdirectory `project` in your assigned git-repository.

0 Problem

The task to solve is the *Pendulum Swing up* problem from OpenAI Gym. This classic control task has a three-dimensional continuous observation space in $[(-1,-1,-8),(1,1,8)]$ – sine and cosine of the angle and the angular velocity – as well as a one-dimensional continuous action space in $[-2,2]$ representing the force we apply. However, the internal state of the Pendulum environment is represented by the angle and angular velocity. The goal is to push and hold the pole in an upright position. We modified the environment, such that it yields binary rewards (1 if the pole is upright and 0 otherwise). It can be found in `lib.envs.pendulum` and a visualization can be seen in Figure 1.

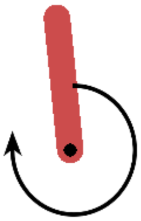

Figure 1: Visualization of the Pendulum environment.

You can pass your own reward function as a parameter to overwrite the binary reward. It has to have the form `def reward(pendulum)`, where `pendulum` is a `PendulumEnv`-instance. You get the current state of the Pendulum via `pendulum.state` and its last action via `pendulum.last_u`. In the control loop, `lib.envs.pendulum` can be used as any Gym environment. You can set the episode length to 200.

1 Implementation

You can choose to implement and apply any reinforcement learning algorithm (from the lecture or beyond) to solve this problem. You have to be able to explain all your choices and steps. Aim to optimize your results considering the return, learning stability and speed (in terms of episodes/samples).

Since the Pendulum is a continuous control problem, you have to discretize the actions in order to apply your *DQN* implementation, whereas policy gradient methods, such as *REINFORCE*, naturally fit continuous action spaces. You can think about subtracting a baseline or even implement an actor-critic approach.

You can also develop a shaped reward function, your own exploration strategy or make use of a heuristic (such as hint-to-goal). We also discussed employing a model of the environment, off-policy learning and experience replay.

2 Evaluation

The evaluation should at least include some learning curves (i.e. the return over time) of your chosen approach and settings. You can additionally think of your own metric and evaluate that as well.

The initialization of function approximators and exploration lead to some randomness, so take into account the performance of multiple runs (e.g. mean and standard deviation or median and interquartile ranges). It is important that your evaluation builds the basis for discussion and scientifically analyzes which are the important aspects and characteristics of your approach – your presentation has to present your findings in a convincing manner. Please also make sure to carefully keep track of intermediate results.

3 Presentation

In the first part of the exam, we are going to discuss your results, so pepare a short presentation of five slides as a PDF. The presentation should include

- 1 slide of motivation (Why did you choose your algorithm? How did you implement your method?),
- 2-3 slides of results (How well did your approach perform and how fast? What was important?) and
- 1-2 slides of discussion (What do your results mean? What are your experiences? What could be improved in your method and how?).

Please push your slides as a PDF and your code to subdirectory `project` in your assigned git-repository **until Wednesday, February 21 at 11:00am**. Submissions after that will not be accepted.