

# Лабораториска вежба 3.1 - Длабоко учење со поттикнување

## Gym - Python библиотека




Gym е Python библиотека која поддржува развој и споредба на алгоритми за учење со поттикнување. Документацијата на оваа библиотека е достапна на следната [страница](#). Инсталација на библиотеката:

```
pip install gym
```





Околина може да се креира на следниот начин:

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

каде `env_name` е името на околината. Библиотеката `gym` обезбедува голем број на [околин](#) кои може да се користат. Пример за околин:

-  FrozenLake-v0
-  Taxi-v3
-  MountainCar-v0

Некои атрибути на околината кои може да се користат:

-  `Action_space` - валидни акции
-  `Observation_space` - валидни состојби
-  `Reward_range` - ранг на наградата која агентот ја добива
-  `Env` - дополнителни информации за околината

Околината се ресетира со повик на функцијата `reset()`:

```
env.reset()
```

Оваа функција ја враќа почетната состојба.

Со повик на функцијата `render()` може да се визуелизира тековната состојба:

```
env.render()
```

За движење низ околината се користи функцијата `step`:

```
state, reward, done, info = env.step(action)
```

каде `action` е акцијата која се презема. Оваа функција како резултат враќа 4 вредности:

- State - објект кој ја претставува следната состојба на околината
- Reward - награда која се добива по преземање на акцијата action
- Done - информација дали епизодата е завршена (ако е завршена, околината треба да се ресетира)
- Info - дополнителни информации специфични за конкретната околина

## Задачи

### Задача 1 (15 поени)

За околината „MountainCar-v0“ изградете и тренирајте DQN агент со користење  $\epsilon$ -greedy политика со  $\epsilon$ -decay. Тестирајте со различни вредности за број на епизоди. Може да ја користите имплементацијата на DQN достапна во скриптата **deep\_q\_learning.py**. Потоа визуелизирајте го движењето на агентот низ оваа околина.

Тестирајте го тренираниот модел во 50 и 100 итерации. Колкава е просечната награда? Дали се добиваат подобри резултати од резултатите во втората лабораториска вежба?

### Задача 2 (15 поени)

За околината „MountainCar-v0“ изградете и тренирајте Double-DQN агент со користење  $\epsilon$ -greedy политика со  $\epsilon$ -decay. Тестирајте со различни вредности за број на епизоди. Може да ја користите имплементацијата на DDQN достапна во скриптата **deep\_q\_learning.py**. Потоа визуелизирајте го движењето на агентот низ оваа околина.

Тестирајте го тренираниот модел во 50 и 100 итерации. Колкава е просечната награда? Дали се добиваат подобри резултати од резултатите во втората лабораториска вежба?