National University of Sciences and Technology (NUST)
School of Electrical Engineering and Computer Science

Faculty of Computing

CS 368: Reinforcement Learning

Lab 11: Deep Q Networks - I

(Stable Baseline3 Library)

Date: 06 December 2024

Time: 02:00 PM – 5:00 PM

Instructor: Dr. Zuhair Zafar

Lab 11: Deep Q Networks - I (Stable Baselines3 Library)

Objectives

Implementing and solving RL problems using Stable Baselines3 Library. Goal is to implement Deep Q Networks using MIP policy.

Tools/Software Requirement:

Google Colab, Python, Gymnasium Library, PyTorch, Stable Baselines3 Library

Introduction

Stable-Baselines3 (SB3) is a popular Python library for reinforcement learning (RL) that provides reliable implementations of state-of-the-art RL algorithms. It is widely used for research, education, and applications of RL in various domains. SB3 builds on top of PyTorch and focuses on simplicity, extensibility, and reproducibility, making it accessible to both beginners and experienced practitioners.

Stable Baselines3 Library

Key Features

1. **Wide Range of Algorithms:**
   - SB3 implements several popular RL algorithms, including:
     - **Model-Free Algorithms:**
       - PPO (Proximal Policy Optimization)
       - A2C (Advantage Actor-Critic)
       - DDPG (Deep Deterministic Policy Gradient)
       - TD3 (Twin Delayed Deep Deterministic Policy Gradient)
       - SAC (Soft Actor-Critic)
       - DQN (Deep Q-Network)
     - **Multi-Agent Algorithms** (via extensions such as SB3-Contrib).
2. **User-Friendly API:**
   - High-level interface to train agents in a few lines of code.
   - Predefined methods for logging, evaluation, and saving/loading models.
3. **Seamless Integration with Gymnasium:**
   - Supports environments from the Gymnasium library and custom environments that follow the OpenAI Gym API.
4. **Extensibility:**
   - Easy to customize or extend algorithms for advanced use cases.

5. **Evaluation Tools:**
    - Built-in support for evaluating policies, hyperparameter tuning, and monitoring progress using TensorBoard.

Installation

You can install Stable-Baselines3 using pip:

```
pip install stable-baselines3[extra]
```

The [extra] includes dependencies for additional tools like Gymnasium and TensorBoard.

Example Code

```
import gymnasium as gym

from stable_baselines3 import A2C

env = gym.make("CartPole-v1", render_mode="rgb_array")

model = A2C("MlpPolicy", env, verbose=1)
model.learn(total_timesteps=10_000)

vec_env = model.get_env()
obs = vec_env.reset()
for i in range(1000):
    action, _state = model.predict(obs, deterministic=True)
    obs, reward, done, info = vec_env.step(action)
    vec_env.render("human")
    # VecEnv resets automatically
    # if done:
    #   obs = vec_env.reset()
```

Lab Tasks

1. **Solve the Lunar Lander problem of gymnasium library using the Stable Baselines3 Library. You must use the Deep Q Network to find optimal Q values using MIP policy. Plot the returns and the MSE (network loss). Train the network for 200,000 timesteps.**

Hint: Look at the examples given in the Stable Baselines3 Documentation. (LINK)

Deliverable:
Please submit your notebook on LMS before the deadline **(9th December 2024, 11:59pm).**

Lab Rubrics

Assessment	**Does not meet expectation** **(1/2 marks)**	**Meets expectation** **(3/4 marks)**	**Exceeds expectation** **(5 marks)**
Software Problem Realization **(CLO1, PLO1)**	The student struggles to formulate the problem as RL and does not apply Deep Q Network algorithm using Stable Baselines3 library to solve it. There is a lack of understanding of the problem's requirements and no attempt to find the optimal policy effectively.	The student formulates the problem as RL with some guidance, applies Deep Q network algorithm using Stable Baselines3 library with hints, and shows it’s working. However, the approach might not be fully optimized or lacks a thorough justification.	The student independently formulates the given problem as RL, applies Deep Q network algorithm using Stable Baselines3 library without guidance, and effectively finds an optimal policy. The approach is fully optimized and can be applied to different similar problems.
Software Tool Usage **(CLO4, PLO5)**	Code has syntax errors, and the implementation of the Deep Q network algorithm using Stable Baselines3 library is incorrect or incomplete. The code is not modular and lacks comments for readability and reuse. The student shows limited ability to use gymnasium / pytorch / Stable Baselines3 library functions where required.	The student has implemented the Deep Q networks algorithm using Stable Baselines3 correctly for the given problem with minor mistakes. The code is mostly correct in terms of syntax and functionality but might not be optimized or well-structured for reuse. Some documentation is provided. The student also shows working knowledge of gymnasium and Stable Baselines3 library where required.	The student has implemented the Deep Q network algorithm using Stable Baselines3 efficiently and correctly. The code is clean, modular, well-documented, and follows best practices for reproducibility and reuse. The student demonstrates full command of the gymnasium and Stable Baselines3 library and its functions.