Programming Assignment 3

DA5007: Special Topics in ML (Reinforcement Learning)

1 Problem Statement

In this assignment, you will implement and evaluate the following reinforcement learning algorithms:

- **Dueling Deep Q-Network (Dueling-DQN)** with two Q-value aggregation variants.
- **Monte Carlo REINFORCE (Policy Gradient)** with and without a baseline.

You will train these algorithms on two standard continuous-control environments and analyse their comparative learning efficiency, stability, and convergence behaviour.

Environment

In this programming task, you'll utilize the following **Gymnasium environments** for training and evaluating your policies. The links associated with the environments contain descriptions of each environment. Please use the exact version of the environment as specified:

- **Acrobot-v1**: The system consists of two links connected linearly to form a chain, with one end of the chain fixed. The joint between the two links is actuated. The goal is to apply torques on the actuated joint to swing the free end of the linear chain above a given height while starting from the initial state of hanging downwards.
- **CartPole-v1**: A pole is attached by an un-actuated joint to a cart, which moves along a frictionless track. The pendulum is placed upright on the cart and the goal is to balance the pole by applying forces in the left and right direction on the cart.

Each algorithm and its variants must be trained and evaluated on both environments.
OpenAI's Gymnasium provides a wide range of environments for testing and benchmarking reinforcement learning algorithms. For more information on various environments, please refer to: `https://gymnasium.farama.org/index.html`

Algorithms

You are tasked with training two variants of each Dueling-DQN and Monte Carlo REINFORCE and assessing their comparative performance.

Dueling Deep Q-Network (Dueling-DQN)

Dueling DQN is an extension of the DQN algorithm, designed to improve learning efficiency by decomposing the Q-value function into two separate streams: one estimating the state value ($V(s;\theta)$) and the other estimating the advantage of each action ($A(s,a;\theta)$). The update equation for the dueling network is:

Type-1 (Mean-Normalized Advantage)

$$Q(s,a;\theta) = V(s;\theta) + \Big(A(s,a;\theta) - \frac{1}{|A|}\sum_{a' \in A} A(s,a';\theta)\Big)$$

Type-2 (Max-Normalized Advantage)

$$Q(s,a;\theta) = V(s;\theta) + \Big(A(s,a;\theta) - \max_{a' \in A} A(s,a';\theta)\Big)$$

Implement both versions (Type-1 and Type-2) and compare their training stability and convergence.

Monte Carlo REINFORCE (Policy Gradient)

The REINFORCE algorithm estimates gradients using complete episode returns.

Without Baseline

$$\theta \leftarrow \theta + \alpha G_t \frac{\nabla_\theta \pi(A_t \mid S_t, \theta)}{\pi(A_t \mid S_t, \theta)}$$

With Baseline Incorporating a learned baseline $V(S_t;\Phi)$ reduces gradient variance:

$$\theta \leftarrow \theta + \alpha(G_t - V(S_t;\Phi)) \frac{\nabla_\theta \pi(A_t \mid S_t, \theta)}{\pi(A_t \mid S_t, \theta)}$$

The baseline is updated using the TD(0) rule.
You must implement both variants and compare their training stability and convergence.

2 Tasks

You are required to compare each algorithm with its own variant ((Type-1) vs (Type-2)) and not with the other algorithm. Please adhere strictly to the following instructions.

- Use $\gamma = 0.99$ for all experiments.
- Tune the hyper-parameters to minimise the regret in all experiments.
- To account for stochasticity, use the average of 5 random seeds for each experiment/plot.
- Plot the episodic return versus episodic number for every experiment.
- The plots should consist of the mean and variance across the 5 runs/seeds.

3 Submission Instructions

- You are required to create a comprehensive PDF report containing key code snippets, observations, insights, and plots.
- Your submission may include either `.ipynb` (Jupyter Notebook) files or `.py` (Python script) files. Ensure that your code is **well-documented** and easy to understand. Each submitted file should contain sufficient explanations, comments, and instructions so that anyone can reproduce your experiments and results without additional clarification. **We will run your code, which must generate the same plots and charts as shown in the report.**

- In your report, include only the important code snippets rather than every line of code.
- Zip your report and code files together and submit the zip file through the portal.
- Submission of both the report and code files is mandatory. Incomplete submissions will not be evaluated and will receive 0 marks.
- Use clear and meaningful code comments for readability, and follow all problem specifications carefully.
- It is recommended to include a brief description at the beginning of your code or notebook explaining the overall structure, dependencies, and execution steps.