BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

WORK INTEGRATED LEARNING PROGRAMMES DIVISION

Deep Reinforcement Learning
Lab Assignment 2

Total Marks = 12 Marks

Intended Learning Outcome:

Students should be able to

- Understand the basic functionality of Q-learning, DQN, and DDQN Methods.
- Implement the concepts of Q-learning, DQN, and DDQN.

Prerequisite:

(1) Students should go through the lectures CS7 - CS14;

(2) Webinar demonstrations

Please note that this assignment will involve some amount of self-learning (on the part of modelling solutions appropriately + programming skills)

Submission Deadline: 31st January, 2026

Instructions:

- Read the assignment proposal carefully.
- Solve all the assignment problems. Submit only one solution file. (Team # - Q_learning_DQN_DDQN)
- It is mandatory to **submit** the assignment in **PDF format only** consisting of all the outcomes with each and every iteration printed. Any other format will not be accepted.
- Add comments and descriptions to every function you are creating or operation you are performing. If not found, then 1 mark will be deducted. There are many assignments that need to be evaluated. By providing the comments and description it will help the evaluator to understand your code quickly and clearly.
- No email submissions will be accepted. Kindly make sure you will submit the assignment on/before the deadline.

How to reach out for any clarifications:

This assignment is administered by

(1) Pooja Harde - pooja.harde@wilp.bits-pilani.ac.in

(2) Divya K - divyak@wilp.bits-pilani.ac.in

(3) Dincy R Arikkat - dincyrarikkat@wilp.bits-pilani.ac.in

Any request for clarification must be addressed through email (official email only) to all the three instructors listed.

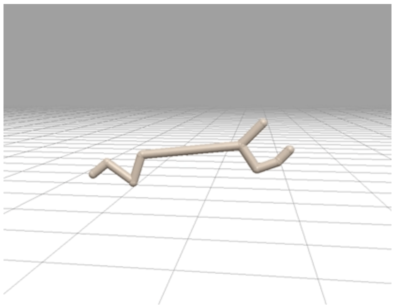
Messaging in TEAMS is discouraged. This is to ensure we maintain track of all the transactions. If we find any clarifications to be shared across all the students, we will share this using discussion forums.

-

Title: Analysing the Q-learning, DQN and DDQN Approaches for Continuous Locomotion

1. Background

The **HalfCheetah** environment models a planar robotic system composed of nine rigid body segments connected by eight joints, including two feet. The objective is to control the robot by applying torques at the joints so that it achieves fast forward locomotion in the positive (rightward) direction as shown in below figure.



To achieve this task, the environment provides:

a. High-dimensional continuous state space
b. Continuous action space
c. Dense but noisy reward signal

<u>Task:</u>

While Deep Q-Networks (DQN) and Double DQN (DDQN) were originally proposed for discrete-action environments such as Atari games, they are often adapted to continuous control tasks through approximation and discretization.

In this task, you will investigate whether value-based methods can be meaningfully applied to Half-Cheetah, and what limitations arise.

2. Environment Description

The task will use HalfCheetah-v5 environment only from MuJoCo.

2.1 Observation Space

The observation space consists of the following parts (in order):

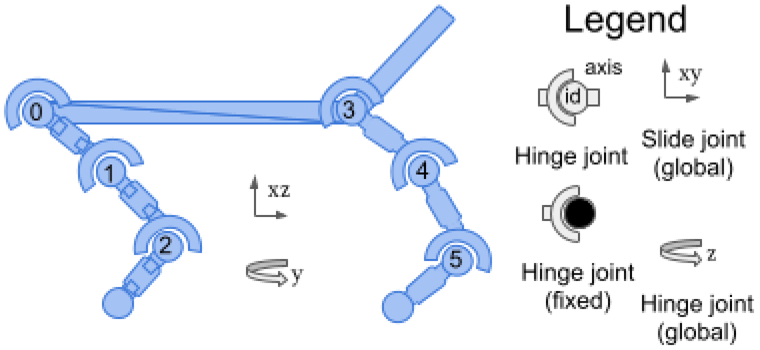
- *qpos (8 elements by default):* Position values of the robot's body parts.
- *qvel (9 elements):* The velocities of these individual body parts (their derivatives).

By default, the observation does not include the robot's x-coordinate (rootx). This can be included by passing exclude_current_positions_from_observation=False during construction. In this case, the observation space will be a Box(-Inf, Inf, (18,), float64), where the first observation element is the x-coordinate of the robot. Regardless of whether exclude_current_positions_from_observation is set to True or False, the x- and y-coordinates are returned in info with the keys "x_position" and "y_position", respectively.

More details can be found on link: [Half Cheetah - Gymnasium Documentation]()

2.2 Action Space

The action space is a Box(-1, 1, (6,), float32). An action corresponds to continuous torques applied at the hinge joints. For more details: [Half Cheetah - Gymnasium Documentation]()



2.3 Reward Function

The reward function provides positive feedback proportional to the forward distance traveled, while penalizing backward movement. The torso and head remain fixed, and control inputs are applied only to six actuated joints corresponding to the front and rear thighs (connected to the torso), the shins (connected to the thighs), and the feet (connected to the shins).

Hence, the reward includes;

a. Forward velocity incentive
b. Control cost penalty

The total reward is: ***reward = forward_reward - ctrl_cost.***

- ***forward_reward:*** A reward for moving forward, this reward would be positive if the Half Cheetah moves forward (in the positive direction / in the right direction. $w_{forward} \times \frac{dx}{dt}$, where dx is the displacement of the "tip" ($x_{afterAction} - x_{beforeAction}$), dt is the time between actions, which depends on the frame_skip parameter (default is 5), and frametime which is 0.01 - so the default is $dt = 5 \times 0.01 = 0.05$, $w_{forward}$ is the forward_reward_weight (default is 1).
- ***ctrl_cost:*** A negative reward to penalize the Half Cheetah for taking actions that are too large. $w_{control} \times ||action||_2^2$, where $w_{control}$ is ctrl_cost_weight (default is 0.01).

info contains the individual reward terms.

3. Starting State (This will be provided in the colab notebook)

The initial position state is $\mathcal{U}_{[-reset\_noise\_scale \times I_9, reset\_noise\_scale \times I_9]}$. The initial velocity state is $\mathcal{N}(0_9, reset\_noise\_scale^2 \times I_9)$.

where $\mathcal{N}$ is the multivariate normal distribution, $\mathcal{U}$ is the multivariate uniform continuous distribution, O_9 is a 9-dimensional zero mean vector, I_9 is a 9 x 9 identity matrix and reset_scale_noice is a hyperparameter to control the magnitude of reset noise ranging from 0.0 (no noise variation) to 0.1 (strong variation).

4. Episode End

Termination - The Half Cheetah never terminates.

Truncation - The default duration of the program is 10000 episodes.

5. Requirements and Deliverables

Common instructions:

- Run all the experiments for 10000 timesteps.
- Draw the plots wherever specified.

Solve the questions from Q1-Q3 using Q-leaning Approach [4 Marks].

Q1. Discretization of action space [1 Marks]

Since we know the Q-learning uses the Q-table for storing the Q(s,a) values for every action taken in the given state, how will you design this problem for the HalfCheetah problem where the action space is continuos. Remember that you cannot have the infinite size for the Q-table to store the values. Based on this understanding, implement the below questions:

a. Convert the continuous action space to discrete set of action space for HalfCheetah. (E.g. the torque values range from -1 to +1, so rather than having all the possible continuous values, create a set of torque values of 12, 24, 36, etc.) **(The formula for this conversion is provided in the colab. The students need to decide the number of values in which they want to distribute the action space into.)**.

b. Identify whether the selected discretization set leads to the control behaviour of the halfCheetah or it introduces instability and poor control flow.

c. Support your answer using the below plots:

   i. action usage statistics

   ii. reward distribution per discrete action

Q2. Q-learning update observation [2 Marks]

Given the observed behavior in Q1, determine **which part of the Q-learning update is most affected**:

$$Q(s,a) \leftarrow Q(s,a) + \alpha \Big[r + \gamma \max_{a'} Q(s',a') - Q(s,a) \Big]$$

For all the mentioned below points draw separate visualization plots and write an explanation for your observation:

1. learning rate interaction
2. max-operator over discretized actions
3. state-action visitation imbalance
4. delayed reward propagation

Q3. Algorithmic Change [1 Marks]

Modify **exactly one** component of Q-learning obtained in the above question Q2.

Allowed components:

1. learning rate schedule
2. discount factor handling
3. exploration policy
4. update frequency

You are not allowed to:

1. change discretization
2. change reward function
3. change environment

Deliverables:

1. Identify the modified component
2. Explain why and how this compensates for discretization
3. Provide before/after learning curves along with the observation summary.

Solve the questions from Q4-Q7 twice one for DQN [4 Marks] and another for DDQN [4 Marks].

Q4. Early Learning Reward Decomposition [0.5 Mark]

Run the HalfCheetah environment using the online setting and collect reward statistics during the **initial phase of interaction** (before the DQN has converged). Based on this data:

1. Identify one behavior that appears profitable early but degrades later
2. Identify one behavior that initially looks unpromising but improves with learning

Support your arguments using appropriate **time-segmented reward plots**, not verbal explanation alone.

Q5. Instability Identification in Value Estimates [1 Mark]

Train a standard DQN for at least **10000 episodes**.

Using plots of:

- predicted Q-values
- episode returns
- training loss

Answer the following questions:

- Does improvement in Q-values always correspond to improvement in performance?
- Identify **one specific divergence pattern** where this assumption fails.

Simply stating "underestimation/overestimation bias" will not receive credit — you must show **where and how it manifests**.

Q6. Targeted Algorithmic Modification [1 Mark]

Modify **any two components** of your DQN pipeline separately to address the instability identified in Q5**: (E.g. if you are choosing a) and d) then you will not perform both the changes at the same time in the network. Consider the network from Q5 as a baseline network to perform these changes).**

a. target network update frequency
b. experience reply buffer size (should be very less than the number of episodes)
c. modifying the epsilon decay
d. adjust the discount factor
e. any algorithm/approach (other than FIFO) to remove the entry from experience reply buffer to add new experiences.

Constraints:

- You may not change the environment
- You may not change reward formulation
- You may not change more than one algorithmic component at a time.

Report:

- What was changed?
- Why this specific change addresses the observed failure/progress mode?

Q7. Confidence-Driven Reduction in Exploration [0.5 Mark]

After reviewing the learning behavior, consider the following statement:

> *"Exploration should decrease as the agent becomes more confident in its action-value estimates."*

Based on your experiments:

- Identify one action or action category that was frequently selected during early exploration but became rarely selected in later training.
- Briefly explain what learning signal led the agent to reduce exploration of this action.

Support your conclusions using plots for action-selection frequency during training.

The below questions (Q8-Q9) are generalized and concludes the whole assignment.

Q8. Performance Visualization and Comparison [1 Mark]

Plot the **cumulative episode return curves and action selection plots** for all the three approaches (Q-learning, DQN and DDQN).

Q9. Answer the following questions.

a. What changed from DQN to DDQN implementation and how it worsen/imporved the half-Cheetah performance? Support your answer with the required plots, architectural change and the learning process. **[0.5 Marks]**

b. Summarize your learnings and observations while implementing both the techniques on the continuous action space. **[0.5 Marks]**

References:

https://www.kaggle.com/code/stpeteishii/gym-halfcheetah-v4-ddpg
https://rickstaa.dev/stable-gym/envs/mujoco/half_cheetah_cost.html
https://gymnasium.farama.org/environments/mujoco/half_cheetah/#rewards