**A screenshot of a game

AI-generated content may be incorrect.**

**Project Name: Frozen lake V1**

**Environment: Frozen lake V1**

**Project applied by:**

**Abdelrahman Walid 20213859**

**Ahmed Mohamed 20212924**

**Mohamed Ashraf 20213875**

**Yahia Ahmed 20213774**

**Supervisor: Eng. Salma Shalaby**

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **Tasks** |
| **20212924** | **Ahmed Mohamed** | **Model search and explanation** |
| **20213875** | **Mohamed Ashraf** | **Model creation and training** |
| **20213859** | **Abdelrahman Walid** | **Plot creation** |
| **20213774** | **Yahia Ahmed** | **Documentation** |

**Frozen lake v1:**FrozenLake-v1 is an environment provided by OpenAI Gym, widely used in reinforcement learning (RL) for benchmarking algorithms in a simple, discrete grid world.

🔸 Environment Description

Frozen Lake is a 2D grid-based environment where the agent (a player) must navigate from a start tile (S) to a goal tile (G), avoiding holes (H). The rest of the tiles are frozen (F) and safe to walk on.

The environment is stochastic by default, meaning that when the agent takes an action, it may slip and move in a different direction.

🔸 Key Features

* Grid Size: Default is 4x4 or 8x8.
* Observation Space: Discrete values representing each grid cell.
* Action Space: 4 discrete actions → 0: LEFT, 1: DOWN, 2: RIGHT, 3: UP.
* Reward:
  + +1 for reaching the goal.
  + 0 otherwise (no penalty for falling into holes).

🔸 Environment Modes

You can change the environment’s behavior with these flags:

* is\_slippery=True: Makes the environment stochastic (default).
* is\_slippery=False: Makes it deterministic.

🔸 State Representation

Each cell in the grid is encoded as an integer in the observation space. For example, in a 4x4 grid:

[ 0, 1, 2, 3,

4, 5, 6, 7,

8, 9, 10, 11,

12, 13, 14, 15 ]

**🔸 Use Cases in RL:**

* Testing value-based methods like Q-learning and SARSA.
* Demonstrating exploration vs. exploitation.
* Understanding policy evaluation and policy improvement in tabular methods.

**Model Used:**

Deep Q Network (DQN) is a reinforcement learning model that combines the classic Q-learning with a deep neural network. DQN is ideal for problems or agents that have a discrete action space, because instead of using the complicated and large Q-table it uses a neural network to make smart fast decisions while learning from past experiences.

Overall, the DQN model is a great fit for our project due to the discrete action space making use of its simplicity and speed.

A graph showing a performance

AI-generated content may be incorrect.

Explanation of Reward vs Timestep Analysis:

The graph illustrates the relationship between training duration (total timesteps) and the agent’s performance (mean reward) in the **FrozenLake-v1** environment using a **Deep Q-Network (DQN)**.

At the early stages of training (around 10,000 timesteps), the agent performs poorly, achieving a low mean reward. Notably, performance dips further around 25,000 timesteps, likely due to insufficient training and high exploration. However, as training continues and reaches approximately **75,000 timesteps**, the mean reward sharply increases, peaking at **0.90**, indicating this is the optimal training duration among the tested values.

Beyond this point, the performance plateaus or slightly declines, suggesting that simply increasing timesteps does not guarantee better performance. This demonstrates the importance of balancing **training duration** and **model stability** and emphasizes the need for **multiple iterations and tuning** to identify the optimal timestep for maximum reward.