**Assignment 5**

**Introduction:**

This assignment consists of 2 parts focused on reinforcement learning. Part 1 is Q-learning implementation on frozen lake environment and part 2 is Atari game environment.

**Part 1:**

In this section, we have successfully implemented the Q learning algorithm by using RL gym library and tutorial. The base code is referenced from the tutorial. The output is shown below:

A graph of a graph

Description automatically generated with medium confidenceA screenshot of a game

Description automatically generated

A graph of different colored lines

Description automatically generated

This is how the code was implemented in the tutorial. Based on this guide, we have chosen three hyperparameter configurations. Each configuration is shown below.

**Hyperparameter configuration 1:**

**alpha = 0.5, gamma = 0.99, epsilon = 0.1**

A graph of different colored lines

Description automatically generated

This hyperparameter configuration appears effective across all map sizes. Both graphs indicate that the agent's performance improves with training.  
The epsilon has a high value which can help explore better. The alpha helps in stability. Lowering the alpha migh help even more. The gamma value is high, lowering this value will increase the learning process.

**Hyperparameter configuration 2:**

**alpha = 0.8, gamma = 0.9, epsilon = 0.5**

A graph of different colored lines

Description automatically generated with medium confidence

This configuration is effective in faster learning through both aggressive info update and substantial exploration. The agent is able to learn more quickly. Both graphs indicate good performance.  
The decrease in epsilon helped to stabilize the learning. Higher gamma value has improved the performance.

**Hyperparameter configuration 3:**

**alpha = 0.1, gamma = 0.8, epsilon = 0.9**

A close-up of graphs

Description automatically generated

This configuration gave a poor performance compared to other configurations. The alpha needs to increased and Epsilon needs to decreased for better performance.

**Conclusion:**

We have successfully implemented the Q learning on frozen lake environment, we have taken 3 configurations for hyperparameter tuning. Out of these, 1 and 2 gave good performance and 3 gave a bad performance. Configuration 1 seems to be the most efficient, so policy iteration algorithm is applied to this set of parameters.

A comparison of a graph

Description automatically generated with medium confidenceA close-up of a graph

Description automatically generated

Based on the graphs, The policy iteration seems to be more efficient in terms of stability and convergence speed. Although both methods are effective, policy iteration algo might provide few advantages over q learning in certain environments. Therefore, we can say that policy iteration is preferred for small and predictable environments whereas q learning is preferred for large environments where evaluations are performed frequently.

**Part 2:**

In this section we have adapted the Q learning code from part 1 on an Atari game environment. We have chosen “BreakoutNoFrameskip-v4” as our Atari game environment. Below is the output generated for this part 2.

A screen shot of a computer

Description automatically generated

A graph showing a line

Description automatically generated

A blue graph with white text

Description automatically generatedThe rewards graph shows considerable variability, with rewards peaking and then dropping over subsequent episodes. This suggests that the agent experiences some episodes with high success followed by episodes of lower performance. These fluctuations indicate that the agent is trying different strategies to find an optimal one. The step logs indicate that the agent’s decision-making process is efficient. The consistency in step times shows that the computational demand per step is stable, which is beneficial for performance analysis and real-time application considerations.