Investigating the Impact of Exploration and Reward Function on the Gameplay of Snake using Monte Carlo Q-Learning

Introduction:

We performed an experiment on the Snake game using the Monte Carlo Q-learning algorithm with an epsilon-greedy policy for action selection. We primarily focused on how the balance between exploration and exploitation, as well as the reward function assigned to different game events, affect the gameplay and learning efficiency.
Methods:

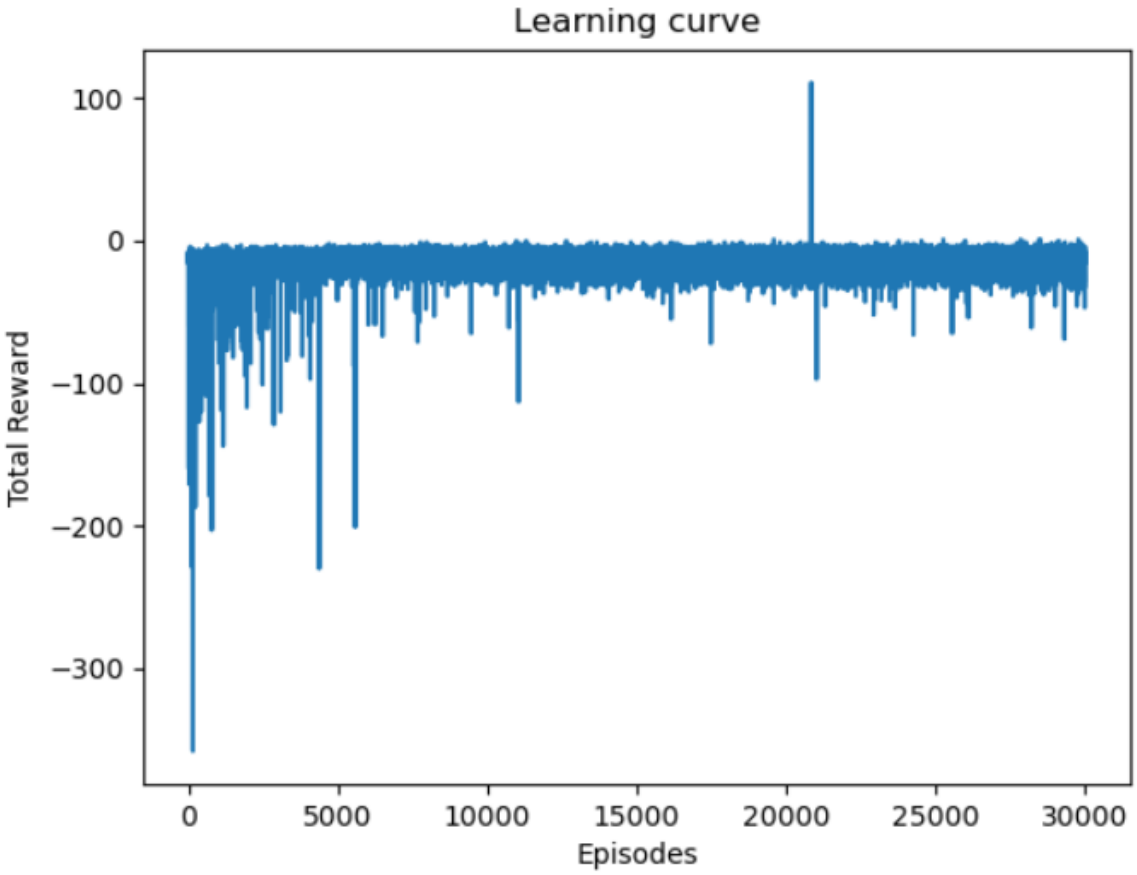
Methods:

The parameters are shown below

```
grid_length = 4
n_episodes = 50000
epsilon = 0.01
gamma = 0.87
rewards = [-10, -1, 2, 100]
```

We set the grid length of the snake environment to 4x4. The learning was conducted over 50,000 episodes, with an epsilon value of 0.01, indicating that 1% of the actions were chosen randomly to facilitate exploration. The discount factor gamma was set at 0.87, reflecting a balance between immediate and future rewards.

The reward function was set as follows: a reward of -10 for a losing move (e.g., crashing into the wall or into the snake's own body), -1 for an inefficient move (making a move that does not result in eating an apple), 2 for an efficient move (making a move towards the apple), and 100 for a winning move (eating an apple). Noting that when snake is full in the game board, we will assign the reward with 100 and stop the game.



Learning curve is shown above. Noticing that at the beginning of training, the reward function is extremely low, and gradually increases, indicating that the snake is learning to get the apple. After around 40000 episodes, we can tell that the snake can have positive gain for each game, indicating that the snake has explored the method to

Results and Discussion:

The learning curve, as evidenced by the plot of cumulative rewards over episodes, showed that the agent's performance improved significantly over time. Initially, the agent made several losing moves, indicated by the negative rewards at the start. However, as the agent learned from its mistakes, the performance improved, reflected by the increasing cumulative rewards.

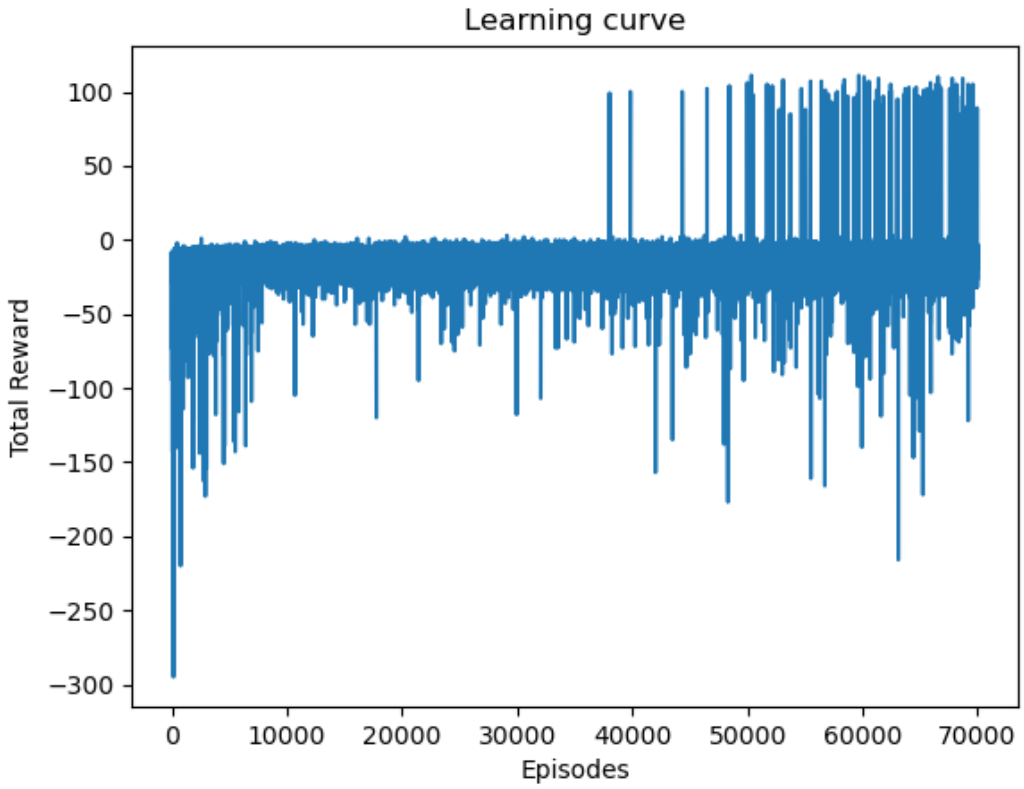
We observed a critical dependency on the gamma value for the learning process. When gamma was very low (e.g., 0, 0.1), the agent fell into an infinite loop at the very first stages of learning epochs. Similarly, slightly higher gamma values such as 0.2 and 0.3 led to an infinite loop around the 20th game, and with gamma = 0.4, the loop occurred

around the 50th game. The agent's performance significantly improved with a gamma value of 0.87, providing the best outcomes among the tested values.

We found that the chosen epsilon value of 0.05 provided a good balance between exploration and exploitation. The agent took random actions 5% of the time to explore the environment and exploited its learned knowledge in 95% of its moves, which proved to be a beneficial strategy for this particular task.

The chosen reward function also played a crucial role in shaping the agent's behavior. The significant penalty for losing moves motivated the agent to avoid crashing, while the reward for efficient moves encouraged it to move towards the apple. The relatively high reward for winning moves drove the agent to prioritize eating apples.

Furthermore, we observed that by increasing the number of training episodes to 70,000, the agent's performance improved notably. The learning curve became steeper in the later episodes, indicating a positive correlation between the number of training episodes and the agent's performance. This suggests that more training allows the agent to better learn and adapt its strategies, leading to improved gameplay over time.



In addition, below is the screenshots of snaking winning the game

```
Reward on game 0 was 8
Reward on game 1 was 5
Reward on game 2 was 24
Won
```

Conclusion:
In conclusion, the balance between exploration and exploitation, the reward function, and the discount factor gamma significantly impact the gameplay in the Snake game. Future work should explore different epsilon values, reward functions, and gamma values, including dynamic ones that change over time, to further improve the efficiency and performance of the agent.

It's worth noting that the penalty for inefficient moves was relatively low (-1), which led to some inefficiencies in the agent's paths to the apples. A higher penalty might push the agent towards finding more efficient paths to the apples. In practical, due to the insufficient of computational power