Connect Four

Reinforcement Learning – 2 Player Game

Group Number: 06

PRESENTED TO:

Dr W. Wilfred Godfrey

PRESENTED BY:

Budati Chethana	2022-BCS-015
Gandham Pranitha	2022-BCS-025
Kappera Mahima	2022-BCS-036
Mirthipati Megha Vardhan	2022-BCS-045

This report explores a reinforcement learning method applied to the game Connect Four, using Q-Learning to train an agent to play optimally. Reinforcement learning entails the gradual improvement of decision-making through interaction with the environment. The agent navigates through different states within the game environment, selecting actions to maximize rewards or minimize penalties. Q-Learning employs Q-values, also known as action values, to iteratively refine the agent's behaviour. By continuously learning from its experiences, the agent strives to achieve optimal performance in any given state. This approach aims to develop a proficient Connect Four player capable of making strategic decisions based on past interactions and rewards received during gameplay. Through this process, the agent learns to anticipate opponent moves and devise winning strategies, enhancing its overall gameplay performance over time.

I. INTRODUCTION

Connect Four is a game where players select a color and strategically drop tokens into a six-row, seven-column grid, aiming to create a horizontal, vertical, or diagonal line of four of their own tokens. The pieces descend vertically and occupy the lowest available space within the column. Classified as an M,n,k-game (7, 6, 4) with limited piece placement, Connect Four has been solved, meaning the first player can consistently secure victory by executing optimal moves.

The game was created by Howard Wexler, and first sold under the Connect Four trademark by Milton Bradley in February 1974. [1].

Connect Four, despite its apparent simplicity, offers ample opportunity for strategic depth and increased chances of winning. For instance, prioritizing certain slots over others is crucial, with those in the middle holding greater value due to their increased potential for forming four in a row. The game's board layout and rules afford a multitude of potential final board configurations, with players having a range of actions at their disposal to achieve them. Our objective was to employ reinforcement learning to uncover the most effective strategies within the Connect Four Markov Decision Process.

II. CONNECT FOUR RULES

The objective of Connect Four is to align four of your checkers in a row while simultaneously blocking your opponent from doing so. However, players must remain vigilant as their opponent can swiftly secure victory. In a typical game scenario, the first player initiates by placing a yellow disc into the central column of an empty board. Subsequently, players take turns dropping their discs into unfilled columns until one player achieves a diagonal alignment of four red discs, clinching victory. Should the board reach full capacity without either player achieving four in a row, the game ends in a draw. The game progresses with each player strategically dropping discs into unfilled columns, aiming to outmanoeuvre their opponent and secure victory.

Figure (i): Connect Four Board

III. MATHEMATICAL SOLUTION

Connect Four is a two-player game with perfect information, ensuring transparency of the game state for both sides. It falls into the category of adversarial, zero-sum games, where one player's gain directly offsets the other's loss, emphasizing the competitive dynamics at play.

The complexity of Connect Four can be gauged by the vast number of potential game board positions it encompasses. In the traditional 7-column-wide, 6-row-high grid, there are approximately 4.5 trillion possible positions across all board configurations, ranging from empty boards to those filled with up to 42 pieces.

James Dow Allen and Victor Allis independently solved Connect Four in October 1988. Allis proposed a knowledge-based approach, outlining nine strategies as a solution, while Allen also identified winning strategies in his analysis. Due to the game's complexity and technological limitations at the time, brute-force analysis was considered impractical during the initial solution attempts.

Brute-force methods, initiated by John Tromp's compilation of an 8-ply database, have successfully solved Connect Four. Q-Learning stands out as the artificial intelligence algorithm proficient in achieving strong solutions for the game.

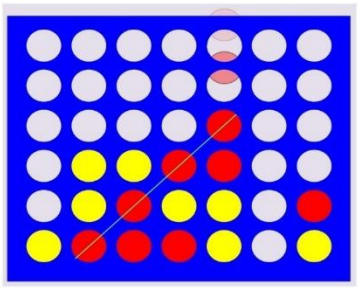

Figure (ii): Winning Connect four board

IV. PROBLEM DESCRIPTION

Our project endeavours to uncover the most effective strategy for playing Connect Four based on a given game board. We define an optimal playing strategy as the sequence of moves that maximizes the likelihood of winning a single game of Connect Four.

A. Action Space

In Connect Four, the action space is relatively limited, with each player having a maximum of seven potential actions available at any given state. An action is defined as placing a piece into one of the seven columns on the board. The number of actions a player can take is contingent upon the occupancy of the columns; thus, the calculation of possible actions at a specific state account for the current fullness of the columns on the board.

$$Actions\ Possible = 7 - C \quad (1)$$

$$C = Number\ of\ full\ columns\ at\ current\ state \quad (2)$$

B. State Space

In Connect Four, the state space significantly surpasses the action space in complexity. A state represents the configuration of the board with placed pieces visible to a player. Depending on whether a player starts the game or joins as the second player, the board will have an even or odd number of pieces, respectively. Estimating the upper bound of possible states involves considering that each position on the 6x7 board can be vacant, occupied by player one's piece, or player two's piece, resulting in a rough calculation of 3^42. However, this simplistic calculation overlooks the fact that certain board configurations are invalid due to gravitational constraints and game rules. For a more accurate estimate, a computer program has determined a lower bound of approximately 1.6 x 10^13 possible positions.

V. Q- LEARNING AND REINFORCEMENT

Q-learning is a reinforcement learning algorithm that doesn't rely on a model of the environment, making it "model-free". It efficiently learns the value of actions in specific states, adeptly managing problems with stochastic transitions and rewards without needing adjustments.

In the realm of finite Markov decision processes, Q-learning excels in discovering an optimal policy aimed at maximizing the expected total reward across successive steps from the present state. Given ample exploration time and a somewhat random policy, Q-learning can establish an optimal action-selection strategy for any finite Markov decision process. The term "Q" denotes the function computed by the algorithm, representing the anticipated rewards associated with taking a specific action in a particular state.

Reinforcement learning comprises an agent, a collection of states S, and a set A of actions for each state. When the agent executes an action $a \in A$, it moves from one state to another, receiving a numerical reward. The objective is to maximize the total reward by considering the potential future rewards. This involves adding the maximum achievable reward from subsequent states to the current state's reward. By doing so, the agent influences its current action based on the anticipated future rewards, which are calculated as a weighted sum of expected rewards from all future steps originating from the current state.

Instead of relying on transition and reward models, we utilize observed next states and rewards (s' and r). Hence, for every observation (s, a, r, s') at time (t), we apply the incremental update rule.

VI. ALGORITHM

1.Define the State Space: In Connect Four, the state space represents the current configuration of the game board. We can represent the board as a 2D array, where each cell can be empty, occupied by player 1, or occupied by player 2.

2.Define the Action Space: Actions represent the columns where a player can drop their piece. In Connect Four, players can drop their piece in any column that is not already full.

3.Define Rewards: In Connect Four, the game is won by getting four of our pieces in a row, either horizontally, vertically, or diagonally. We can assign positive rewards for winning, negative rewards for losing, and zero rewards for intermediate states or draws.

4.Intialize Q-table: Creating a Q-table to store Q-values for each state-action pair. Initially, these values can be random or set to some default value.

5. Q-Learning Algorithm:

Choosing an action based on exploration strategy.

Taking the chosen action and observing the next state and reward.

Updating Q-values using the Q-learning update rule:

$$Q(s, a) \leftarrow Q(s, a) + \alpha(r + \gamma \max_a Q(s', a) - Q(s, a))$$

where alpha is the learning rate, gamma , the discount factor, s' is the next state and a' is the next action.

6. Repeating the above steps for a large number of episodes or until convergence.

7. After training, we use the learned Q-values to play Connect Four by choosing actions with the highest Q-values for each state.

8.Optionally, we can fine-tune hyperparameters like learning rate (alpha), discount factor (gamma), and exploration rate for better performance.

VII . PSEUDO CODE

```
initialize Q-table with random values

for episode in range(num_episodes):
    reset the game to initial state
    while game not over:
        choose action using epsilon-greedy policy
        take action, observe next state and reward
        update Q-value using Q-learning update rule
        update current state

    reduce epsilon (if using epsilon-greedy strategy)

# After training, play using learned Q-values
play game using learned Q-values
```

VIII. REFERENCES

[1] Connect Four - Wikipedia

[2] Q-Learning in Python - GeeksforGeeks

[3] MIT. “Connect 4.” Connect 4, MIT, web.mit.edu/sp.268/www/2010/connectFourSlides.pdf

[4] Q-learning - Wikipedia