Reinforcement Learning for Game 2048

Team Members

- Rishitha Rani Pakam - 00917146
- Narasimha Reddy Padire - 00915003

Optimizing Gameplay Strategies for 2048
Using Reinforcement Learning Techniques

Statement of project:

- We hypothesize that an AI agent, utilizing reinforcement learning methods like Q-learning and Temporal Difference (TD) Learning, can effectively play the game 2048 by selecting optimal moves to maximize the score, achieve tile values of 2048 or higher, and minimize decision time compared to other strategies like Expectimax and baseline approaches.
- The agent will work within constraints like limited board space and random tile generation to maximize its score and outperform baseline strategies.
- Different agent designs will be tested to find the best approach for this goal.

Methodology

An environment is established for an AI agent that employs reinforcement learning techniques, such as Q-learning and TD Learning, to make optimal moves in the game 2048, maximizing scores and high tile values while navigating challenges and comparing its performance against Expectimax and baseline strategies.

Techniques: TD Learning, Q-Learning, Expectimax, Baseline Model

Programming Language: Python

Platform: Jupyter notebook

Python Libraries: NumPy, Scipy and Pandas for data manipulation. Matplotlib, Seaborn for visualization of results

Deliverables

- Agents will be created using selected algorithms (TD Learning, Q-learning, Expectimax and Baseline models) to decide the optimal move during gameplay.
- A class for environment called game.py will be created for the game, which will contain several methods to handle different tasks, like setting up the game board, calculating scores, adding new tiles, moving and merging tiles, and checking if the game is over.
- State Evaluation: Function to evaluate current state of game board.
- Move Validation: Function to determine if a move is valid based on game rules.
- Score Calculation: Function to calculate the score based on the moves made.
- Test Cases: A set of test cases will be created to verify that the game and the expert agent function correctly.

Evaluation

Performance Metrics:

Average Score Achieved: Determines how well each algorithm performs over multiple games.

Highest Tile Achieved: Indicates the highest tile reached by each agent during simulations.

Win Rate: Percentage of times the agent reaches or surpasses the 2048 tile.

Decision Time per Move: Average time taken by each agent to make a move.

Learning Rate and Convergence: Measures how quickly the agent adapts and stabilizes its strategy.

Comparative Analysis:

- Analyze the performance differences for implemented algorithms.
- Assess how well each agent meets the defined objectives.

Thank you