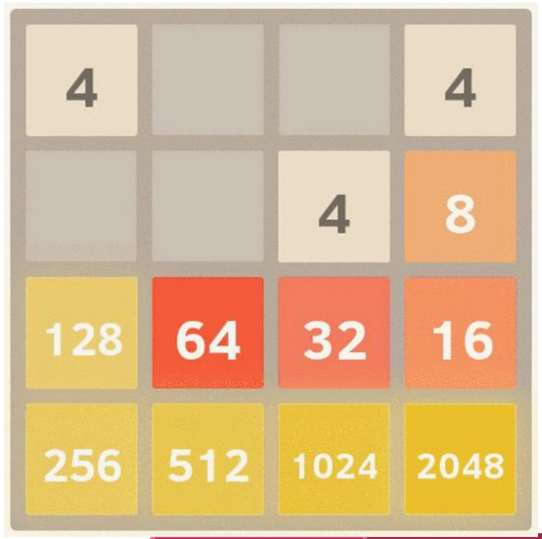
2048 Game Agent Using Reinforcement Learning

By

Stuti Agarwal (stuti.agarwal@sjsu.edu)
Bharath Gunasekaran (bharath.gunasekaran@sjsu.edu)
Riddhi Jain (riddhi.jain@sjsu.edu)
Tamanna Mehta (tamanna.mehta@sjsu.edu)

Introduction

- 2048 is a popular web and mobile game developed in 2014.
- A single player plays the game on a 4x4 grid.
- The player slides the board tiles in one of the four directions to merge the tiles with the same value, intending to create a tile with higher values each time.
- A player moves the tiles using the four arrow keys
    - up,
    - down,
    - right and
    - left keys.



Problem Statement & Motivation

2048 game is time-consuming and take hundreds of moves to reach 2048 or higher value.

Currently, there is no strategy to overcome the randomness involved in this game. The use of RL can help find the most efficient moves to achieve the highest score in an optimal manner.

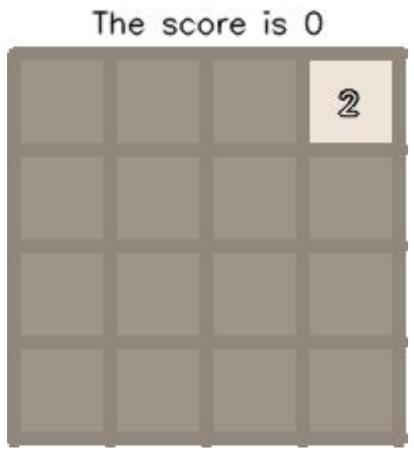
A Reinforcement Learning approach can be beneficial to solve the issue of inherent randomness in this game.

Methodologies

RL Technique	Team Member
TD0 (Temporal Difference)	Riddhi Jain
Sarsa (State Action Reward State Action)	Stuti Agarwal
Q-Learning	Bharath Gunasekaran
DQN	Tamanna Mehta

TD0 (Temporal Difference)

- It is a reinforcement learning technique which is a blend of Monte Carlo and Dynamic Programming.
- use random sampling to learn from the current environment of the value function.
- It is a better technique which adjusts the prediction for the future before the final result is known.
- The Gif here shows the TD0 agent playing the game the Score obtained is **2896** and the max value it converges on is **256** for 10 episodes.

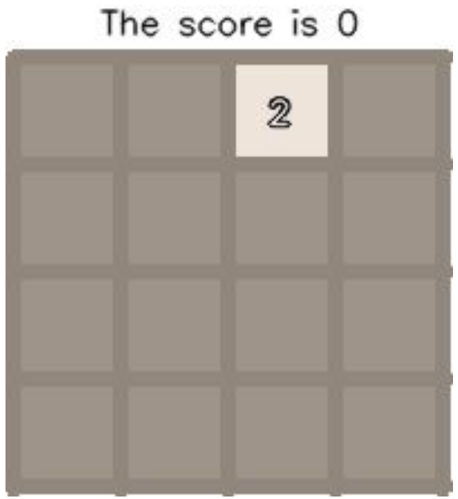


TD0 Hyperparameters Initialization

Hyperparameter	**Value**
Alpha (Learning Rate)	0.02
Gamma (Discount Factor)	0.9999
Epsilon (Exploration Rate)	0.0001
Episodes	10

Sarsa(State Action Reward State Action)

- SARSA is an ON-Policy technique in Reinforcement learning
- It uses the action performed by the current policy to learn the Q-Value.
- It Updates the Q-table with (S,A,R,S’) samples generated by the current policy. (S’,A’) is the next state and next action in transition samples.
- After reaching S’ it will take action A’ and Use Q(S’,A’) to update the Q-Value.
- The Gif here shows the SARSA agent playing the game the Score obtained is **5772** and the max value it converges on is **512** for 10 episodes**.**

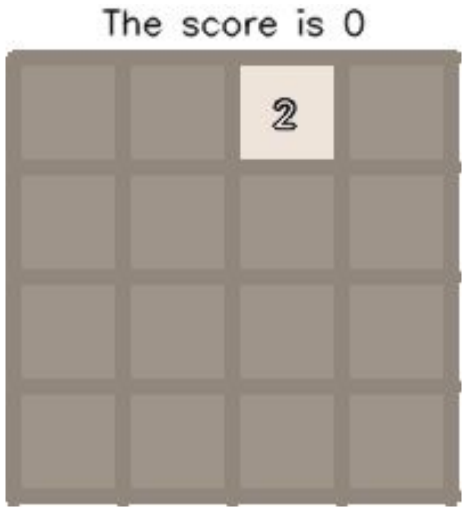


SARSA Hyperparameters Initialization

Hyperparameter	**Value**
Alpha (Learning Rate)	0.01
Gamma (Discount Factor)	0.75
Epsilon (Exploration Rate)	0.001
Episodes	10

Q Learning

- Q-Learning is an OFF-Policy technique in Reinforcement learning
- It uses the greedy approach to learn the Q-Values
- we store all the Q-values in a table that we will update at each time step using the Q-Learning iteration
- The Gif here shows the Q-learning agent playing the game the Score obtained is **5388** and the max value it converges on is **512** for 10 episodes.



Q-Learning Hyperparameters Initialization

Hyperparameter	**Value**
Alpha (Learning Rate)	0.025
Gamma (Discount Factor)	0.9999
Epsilon (Exploration Rate)	0.0001
Episodes	10

DQN

- Combination of Deep Neural Network and Q learning.
- Agent’s Brain-Deep Neural Network consist of Convolutional Layers and FC Layers.
- Replaces regular Q table with a neural network.
- Uses a loss function rather than Q-Learning equation where loss is MSE.
- Uses Predicted Q value, target Q value and observed reward to compute the loss to train the network
- Max value on board is 256 and Highest score is 2400 for number of episodes-10

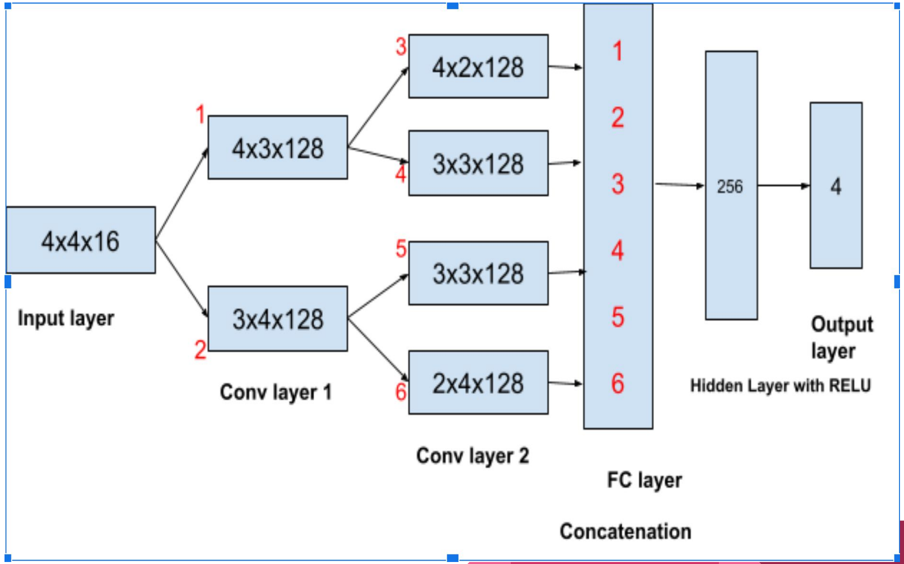


Figure1: Network Architecture

DQN Hyper Parameters Setting

Hyperparameter	**Value**
Epsilon (for Greedy approach)	0.9
Gamma for Q Learning	0.9
Memory capacity	5000
Learning Rate	.0005
Batch size	512
Optimizer	RMS prop
Loss Function	MSE **(mean(square(Q(st,at) - (r + gamma x max(Q(st+1,a)))))**

Code Walk Through

Conclusion

Due to lack of computing resources, currently models are able to converge with max value of 512 for number of episode=10. The model needs to be trained with more no of episodes to achieve a value of 2048 on a tile/matrix cell.

RL Technique	**Max value in board**	**Total Score of board**
Q-Learning	512	5388
SARSA	512	5772
TD0	256	2896
DQN	256	2400