ARTIFICIAL INTELLIGENCE

2048

By Miguel Chavez and Edward Garcia

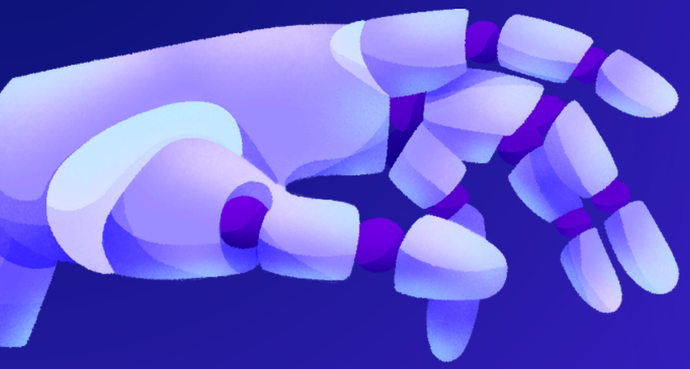
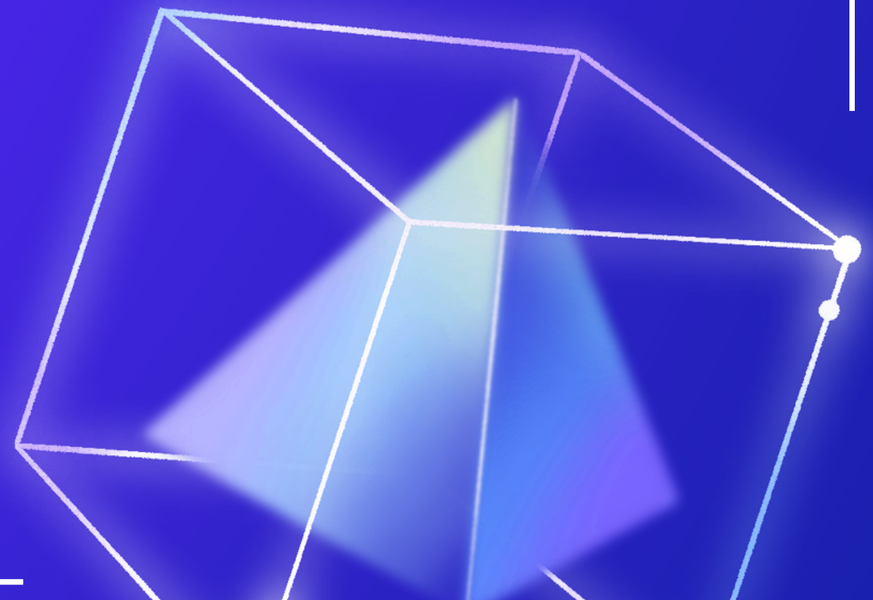


TABLE OF CONTENTS



INTRODUCTION

To develop a Reinforcement Learning (RL) agent capable of not just winning the 2048 game, but achieving high scores by learning adaptive strategies.

Why 2048

- Large and dynamic state space.
- Combines randomness and strategic planning.
- An ideal benchmark for RL in non-deterministic environments.

SYSTEM ARCHITECTURE OVERVIEW

- Environment: Custom 2048 simulator using Gymnasium.
- Agent: Learns and decides moves (up, down, left, right).
- Observation Space: 4x4 grid with tile values.
- Actions: Discrete action space (4 directions).
- Reward Function:
- Positive reward: based on merged tile value.
- Penalty: for invalid or ineffective moves.
- Bonus: reaching 2048 tile or high final score.

DYNAMIC SYSTEMS PERSPECTIVE

01

The system behaves like a non-linear discrete-time dynamical system.

02

Game progression:

$S_{t+1} = f(S_t, A_t, R_t)$

$S_{t+1}=f(S_t,A_t,R_t)$

Where R_t = randomness (new tile), and

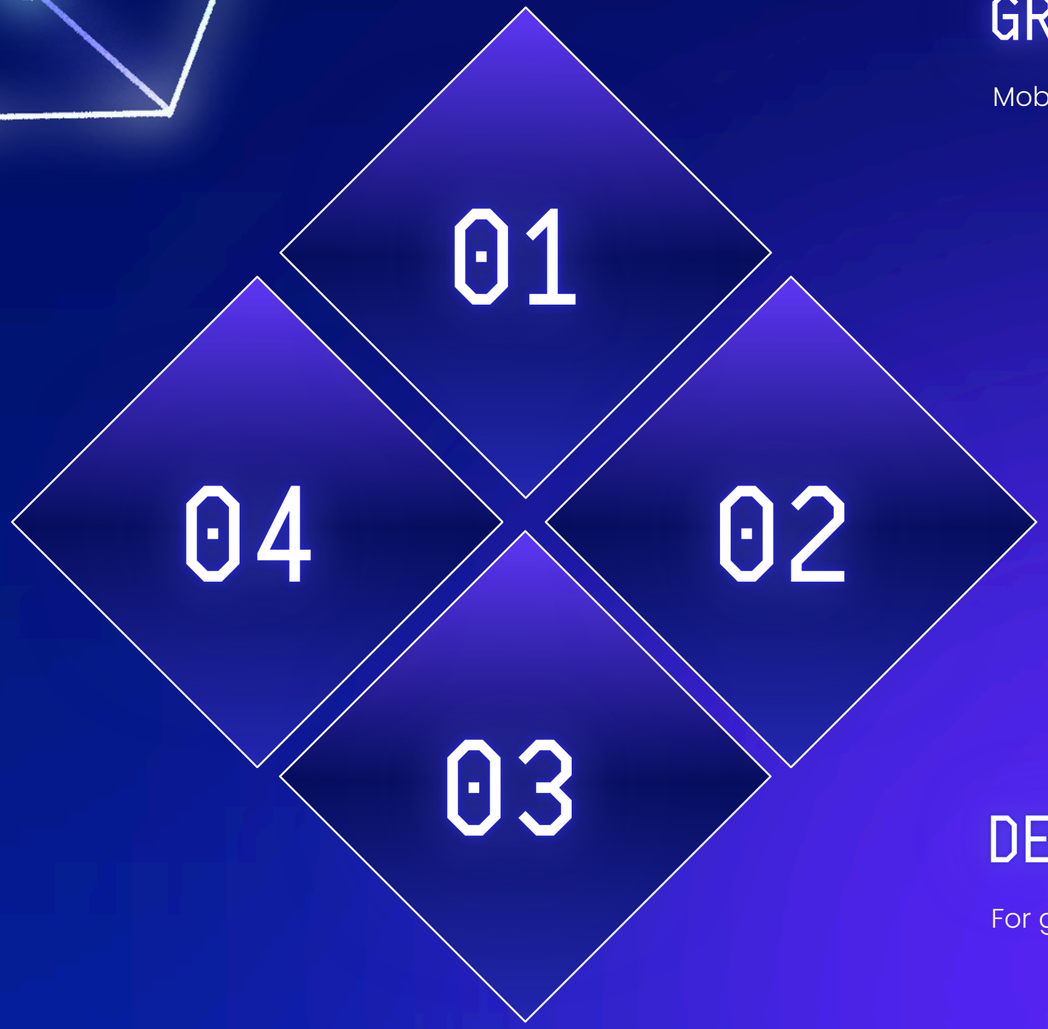
A_t = agent action.

03

- Small differences in initial states → large outcome variance.
- Highlights importance of robustness and generalization.

04

“Good path” vs “bad path” dynamics depending on agent effectiveness.



GRANULAR REWARD SIGNALS:

Mobility, clustering potential, future merges.

DYNAMIC EVALUATION

Stability over time, adaptiveness in late game

Q-LEARNING (TABULAR):

Initial phase for simplicity and fast testing.

DEEP Q-NETWORK (DQN):

For generalization across unseen board states.

NEXT STEPS

- Finalize and train DQN with target networks & replay buffer.
- Add performance logging and visualization.
- Run multiple experiments for robustness.
- Prepare final report with results and insights.

Q&A / FEEDBACK

THANK YOU!