MONASH University
Engineering

Department of Electrical and
Computer Systems Engineering

ECE4095 Final Year Project 2020

Phadon Phipat

Solving the Rubik's Cube using Reinforcement Learning

Supervisor: Dr Mehrtash Harandi

Objective:

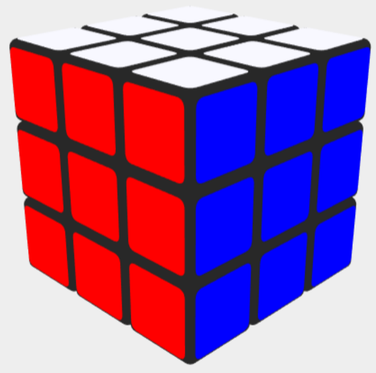
Many previous attempts at a deep learning solver for the Rubik's Cube use supervised learning where the AI learns to copy another solver, or results in an solver only able to solve partially scrambled cubes. This project demonstrates an algorithm that is able to solve most cubes without any specific domain knowledge given during the training process.

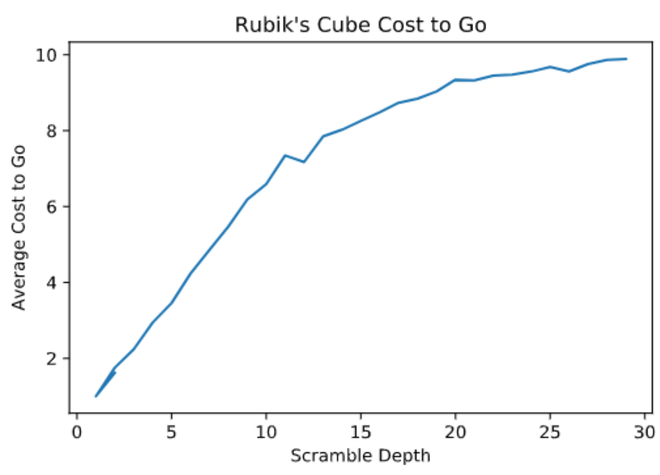
Artificial General Intelligence:

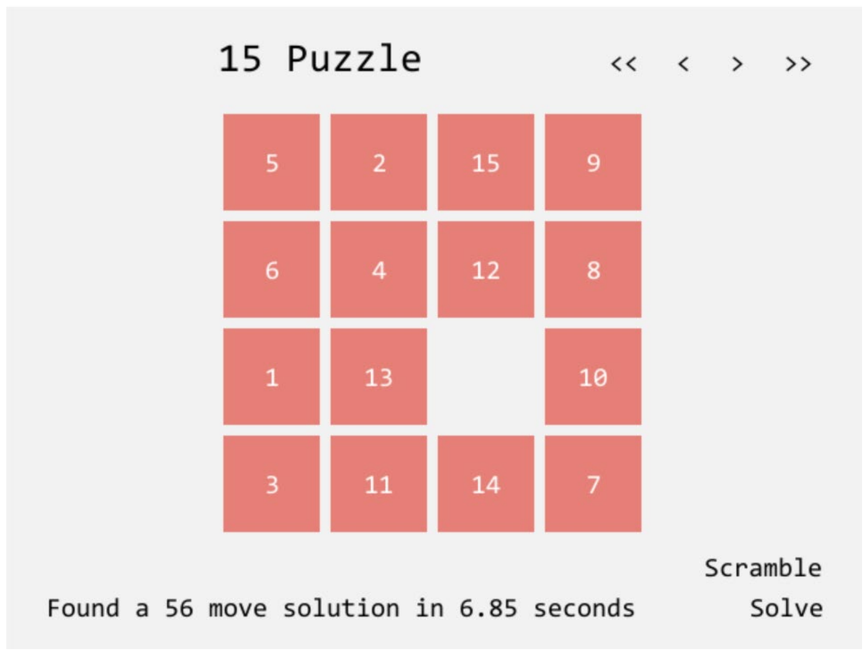
Researchers believe that reinforcement learning is a step towards achieving artificial general intelligence. Reinforcement learning can build more generalised solutions that can be deployed to many different scenarios.

"As a technologist, I see how AI and the fourth industrial revolution will impact every aspect of people's lives."

Fei-Fei Li, Professor of Computer Science at Stanford University.







Deep Reinforcement Learning:

A network is trained to predict the amount of moves needed to solve a cube from a given state via a dynamic programming approach called value iteration. If the state is not a goal state, there is one move that will bring it one step closer to being solved. Therefore the network is trained to predict its value as one more than the best value of the next states that can be reached within one move.

$$V(s) = \begin{cases} 0, if\ goal\ state \\ 1 + \min_a V(s')\ otherwise \end{cases}$$

To solve a cube, A* search is performed on the scramble using the network as a heuristic function. This search uses the network to search in promising directions that bring the cube closer to being solved.

Results:

The algorithm is able to solve all scrambles from the 2x2 Cube and the 15 and 24 puzzle. The algorithm is able to solve most scrambles on the 3x3, however unfortunately takes a large amount of moves to solve more difficult states. Training was cut short while the network was still learning due to time constraints.

Check out the code at

python PyTorch Pygame three.js JS