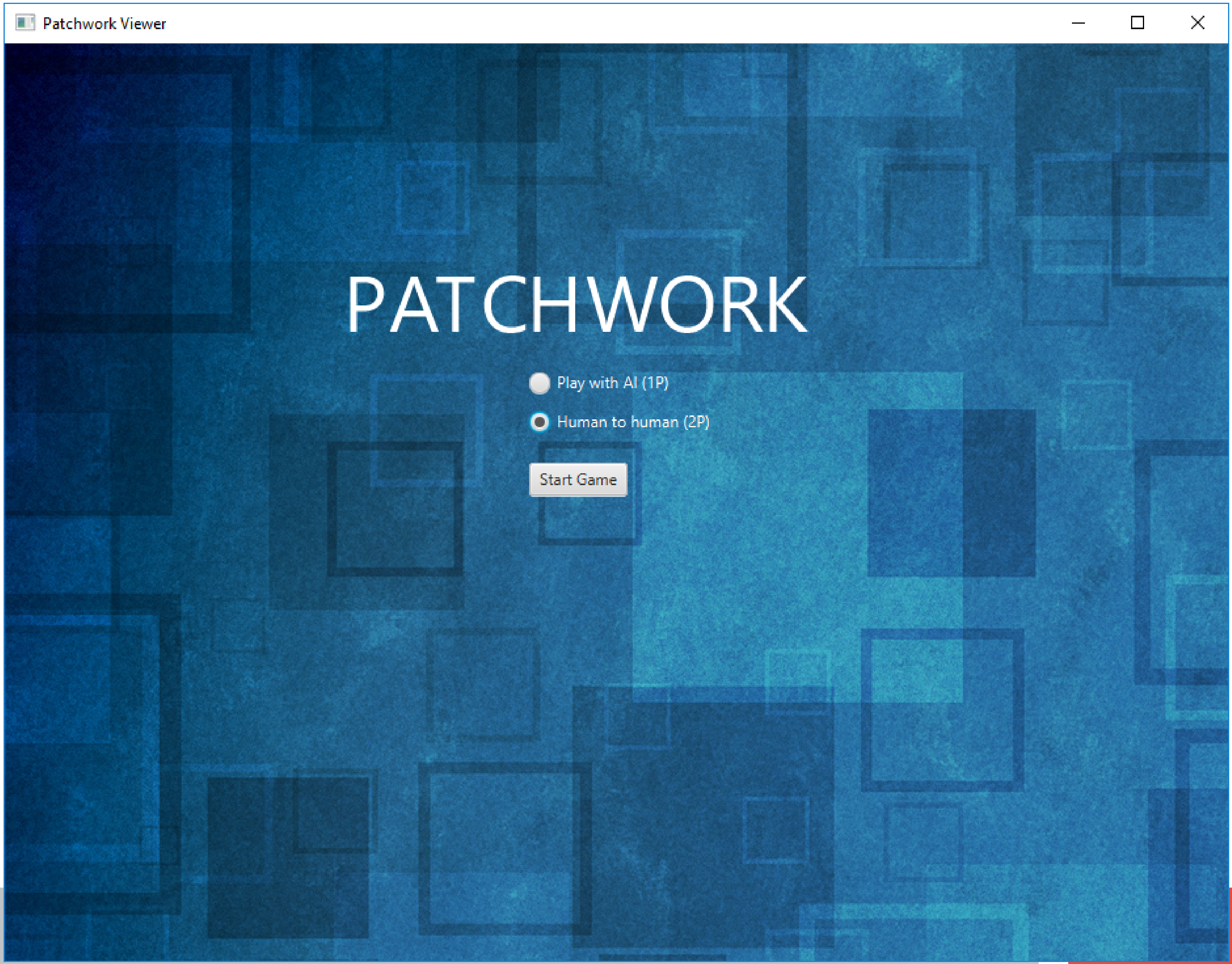
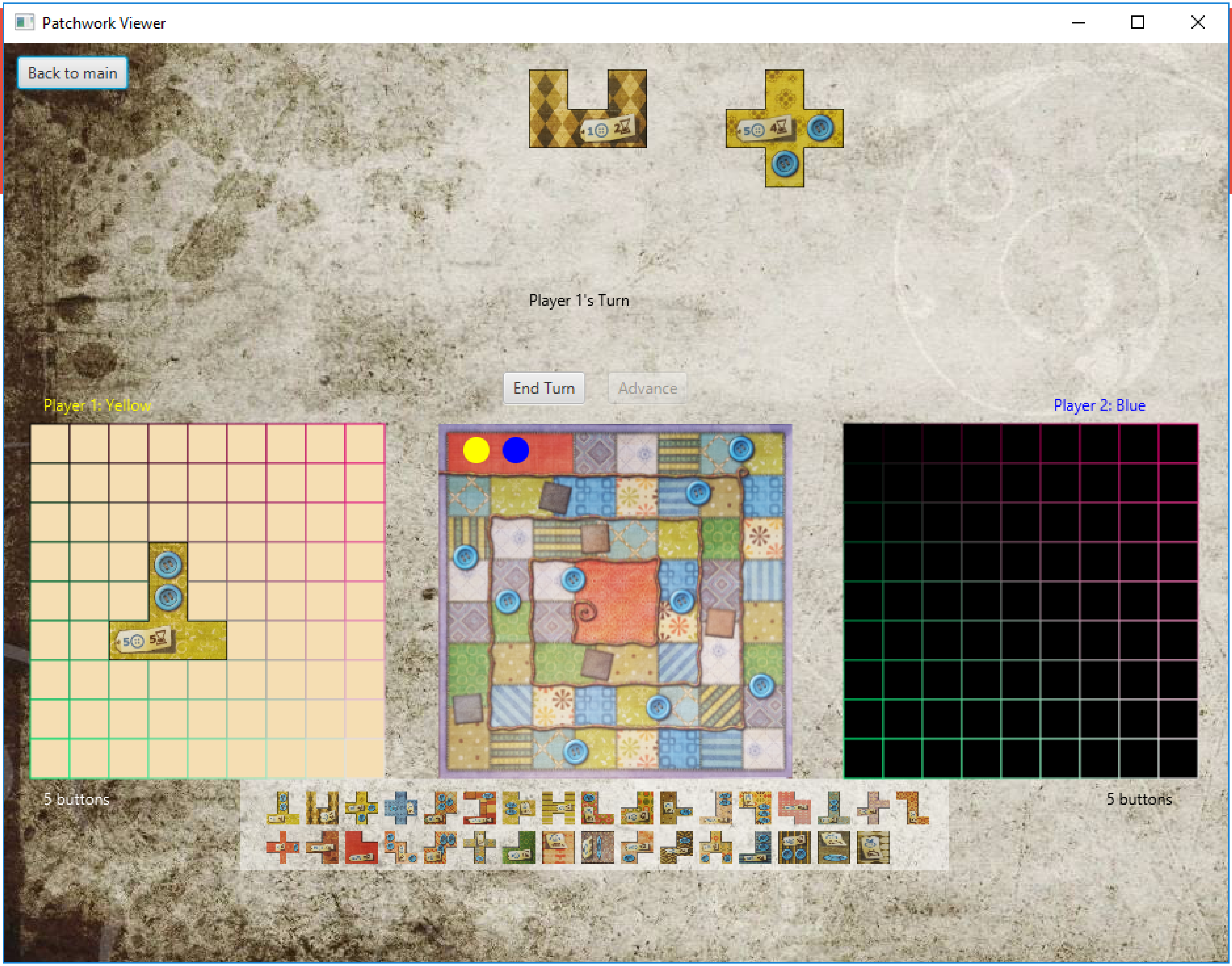
COMP1140 Assignment 2

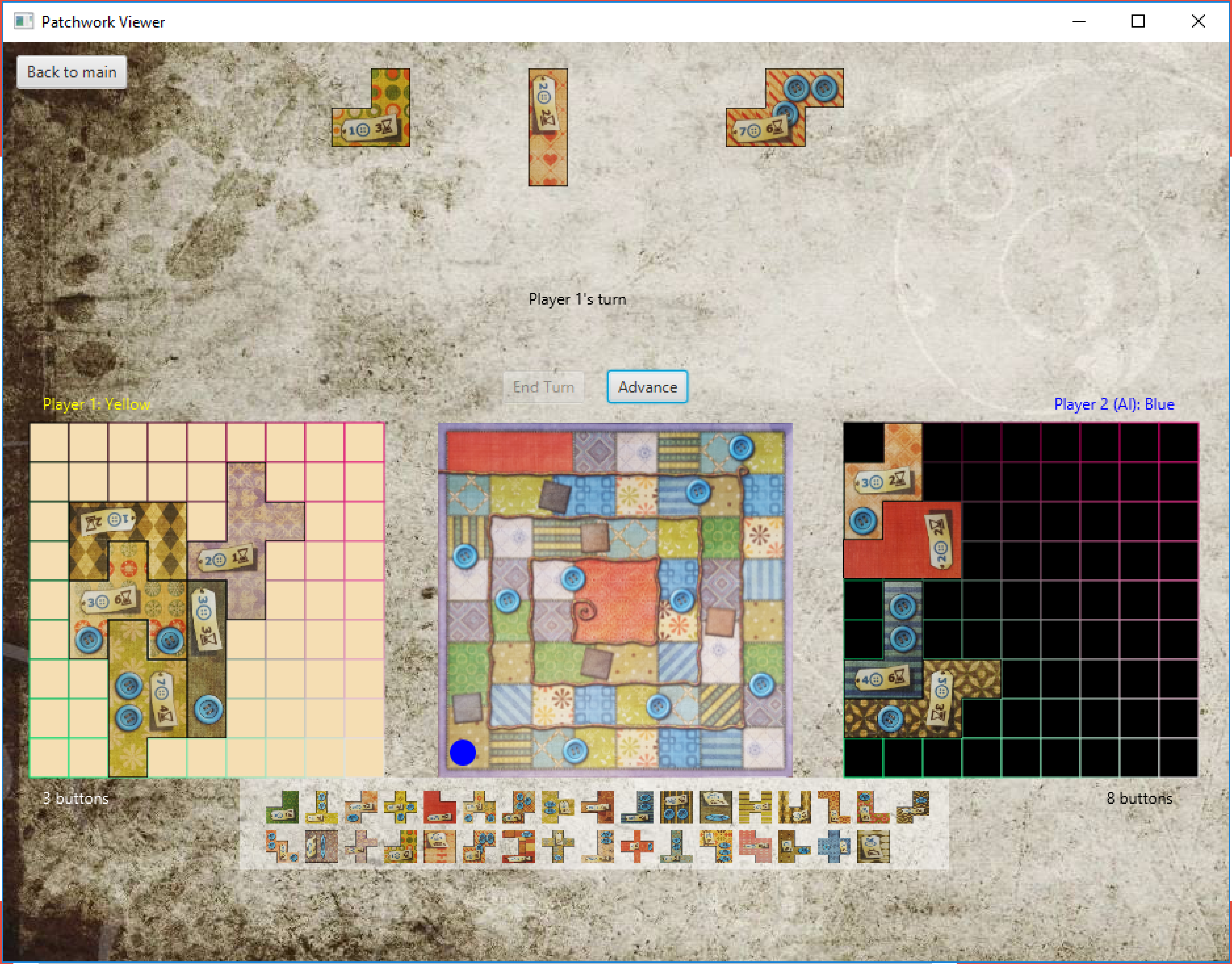
Ziyang Liu (u6210090)
Adonis Mouti (u6385898)
Jay Chen (u6309924)

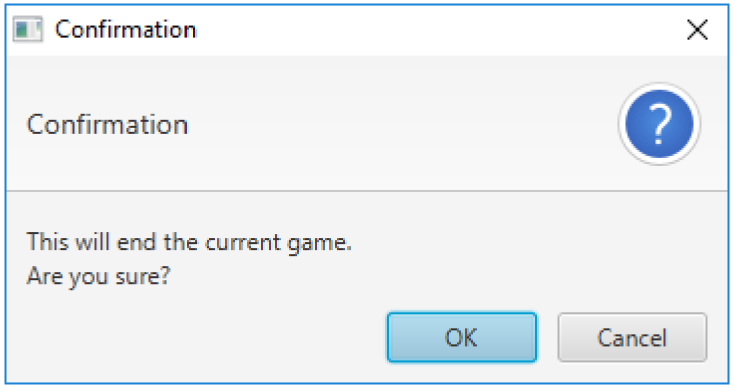
Demo of the Patchwork



Play with AI (1P) [✓ Human to human (2P)]


Patchwork Viewer
Back to main
1 2
5 4
Player 1's Turn
End Turn
Advance
Player 1: Yellow
Player 2: Blue
5 5
5 buttons
5 buttons


Patchwork Viewer



Confirmation

Confirmation

This will end the current game.
Are you sure?

Design & Structure

1. PatchworkGame

2. State

3. PatchworkAI

4. Viewer

5. AITraining

***6. MCTS**

***7. Matrix**

***8. AITrainingMultiThreading**

Methods

1. Data & logic process: Java8

2. GUI: JavaFx

*3. Nerual Network: Keras, Tensorflow(python framework)

*4. Monte-Carlo Tree Search: Java8

Inspirations & Reflection

1. Static field or Class field

Static field: esaier to access and maintain the data

e.g. three candidate tiles are stored as static

Class field: safer to access, data bounded together, but difficult to share the data to every method

e.g. Multithreads: objects are safer to deal with, but when trying to share the data, need to pass the reference of the object in the parameters...

Inspiration & Reflection

2. Less code: methods and variables makes less coding

1. If some code are duplicate in doing configuration(e.g. JavaFx), extracts it as a method

2. Variables + proper data structure

Inspiration & Reflection

3. Well documentation saves teamates' time

1. Comments about the method's functionality and the role of the variable

2. All of methods should be summerised in one document, easier for other developer to check

Inspiration & Reflection

4. Gitlab makes communication easier

1. Well use of the Wiki
2. Importance of issues
3. Attention to the merge request unit

Inspiration & Reflection

5. Right use of Git

1. Branch should be function-base, not developer-base

2. Better to do the code review before merging

AI

Methods:

Inspired by AlphaGO, attempt to use Nerual Network and Reinforcement Learning

AI

Available methods:

0. **Human simple algorithms**
1. **Nerual Network(Supervised Learning)**
2. **CNN**
3. **Q-Learning (Model-based RL)**
4. **Monte-Carlo Tree Search**
5. **Mone-Carlo Tree Search + Nerual Network**

AI

0. Human simple algorithm

OriginGenerator(): Gives the first available placement string by brute force searching

SmarterGenerator(): Picks the best tile among three and get the first suitable position and rotation.

RandomGenerator(): Gets all available placement strings and randomly choose one of it.

Data

Smart:Origin = 53%:47% (10765:9235, 20000 in total)

Smart:Random = 53%:47% (2657:2343, 5000 in total)

Random: Random = 52%:48%(10280:9720, 5000 in total)

Smart: Smart = 50%: 50%(10002:9998, 20000 in total)

*: All data stored in the project/data folder

Nerual Network Design

Two network:

1. Learn to choose the tile
2. Learn to place the tile

Supervised Learning on NN

By Learning on the winner’s choice of the tile in datasets and predict the best tile according to the generated feartures

Model config:

1. Layers: 6(1 input, 1 output, 4 hidden layers)
2. Activation Functions: ReLu, Softmax
3. Features : 8 (Including player’s state and playboard state)

Supervised Learning on NN

Model result:

1. Training sets: 300 samples
2. Testing sets: 440297 samples
3. Accuracy: 99.04%
4. Accuracy on All player’s choice (not only winners’): 98.67%

*DetailDataset06.csv

5. Conclusion: Performance is the same as the simple human algorithm, abandonded.

Supervised Learning on NN

Learning to predict the best position

Model config:

1. Layers: 6(1 input, 1 output, 4 hidden layers)
2. Activation Functions: ReLu, Softmax
3. Features : 170 (Including player’s state and playboard state, 8 previous feature and 81*2)

Model result:

1. Training sets: 1000 samples
2. Testing sets: 436032 samples
3. Accuracy: 43.60%
4. Competition with *smartGenerator:49%:51%(248:252,*500 in total)

Reinforcement Learning

Patchwork is a modeless game due to its large search space.

Don’t have a fixed reward matrix.

Use Monte-Carlo Tree Search instead of other methods(e.g. Q-learning)

The combination of MCTS and NN makes the AI possible to have a value function evaluating the action instead of searching the tree during the competition.

REFERENCE

[1] C.Szepesvari, 2009, *Algorithm for Reinforcement Learning*

[2] D.Silver, A.Huang et.al, 2016, *Mastering the Game of Go with Deep Nerual Networks and Tree Search*