CE888 Assignment 1 Report

Submitted as part of the requirements for:

CE888 Data Science and Decision Making

**Name**: Ai Xingyu

**Tutor**: Ana Matran-Fernandez

**Date**: 8 February 2019

Table of Contents

[1. Abstract 1](#_Toc621960)

[2. Introduction 1](#_Toc621961)

[3. Background 2](#_Toc621962)

[4. Methodology 2](#_Toc621963)

[5. Experiments 4](#_Toc621964)

[6. Discussion 5](#_Toc621965)

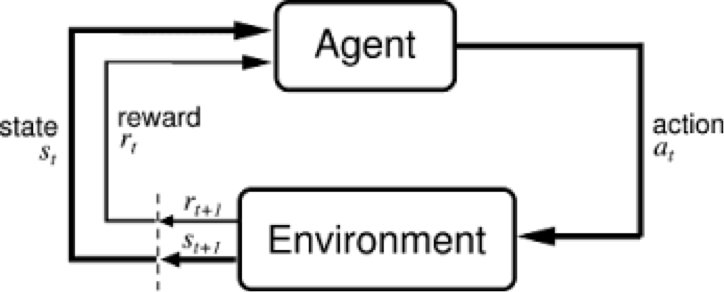
[7. Conclusion 5](#_Toc621966)

[8. Reference 5](#_Toc621967)

[9. Plan 6](#_Toc621968)

1. **Abstract**

Reinforcement learning is a highly effective machine learning technology in the game artificial intelligence. This is a kind of unsupervised learning which can let computer learn game playing skills by compete itself. In that case, the performance of AI player should be improved from almost making the random chooses at first to making perfect movement for every step. Hence by apply this type of learning methodology, computer will evaluate their strategy and try to get better reward.



1. **Introduction**

This project is launched as the requirement of module ce888 assessment to build a decision trees for expert iteration. [1] [2] In this programme, the basic idea is combine both neural network and Monte Carlo tree search(MCTS) to build a strong AI game player.

The player’s action of each round of game will be record and be used to train the neural network system. After training the neural network, it will be used to help and enhance the MCTS algorithm to achieve a better performance.

With the promotion of electronic equipment, video games are also constantly developing. Game AI will have a bright future as well as a vast market so it is necessary for developers to find out a new solution for create AI. Besides, reinforcement learning can be also applied to many aspects of industry such as navigation, robot design and so on. Because of all these reasons, as an experiment in applying reinforcement learning, this project is worth to pay researcher’s strength on it.

1. **Background**

There are already some examples which use reinforcement learning and MCTS in the real life. In 2016, a AI go player Alpha Go challenged and defeated one of the most skilful human players. [3] It is trained based on the human player’s record. However, one year later, AlphaGo Zero came out. This time it applied reinforcement learning and do not use any human player’s record. By fight against itself, the AI improved very fast and finally the AlphaGo Zero defeated all the previous version after trained for 40 days .

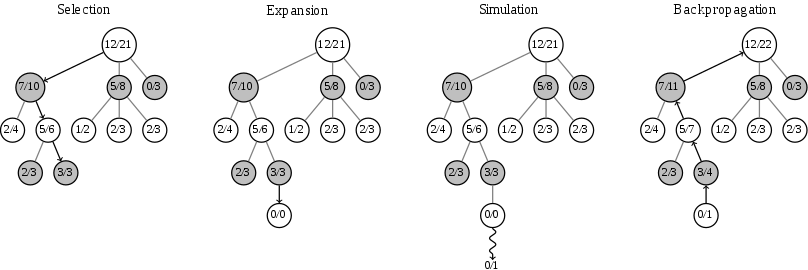
According to the paper provide by DeepMind, there are a lot of benefits to use reinforcement learning to train the AI.

*“Much progress towards artificial intelligence has been made using supervised learning systems that are trained to replicate the decisions of human experts. However, expert data is often expensive, unreliable, or simply unavailable. Even when reliable data is available it may impose a ceiling on the performance of systems trained in this manner” (Mastering the Game of Go without Human Knowledge, DeepMind,2017)* [4]

Collecting human player’s record will face many problems and what is more, these data may not useful enough after the AI has already go beyond human. The most obvious benefit of reinforcement learning is the developer do not need to collect training dataset. In addition, this algorism can let AI outperformed than humans so that their skill will not be bound by the upper limit of human player.

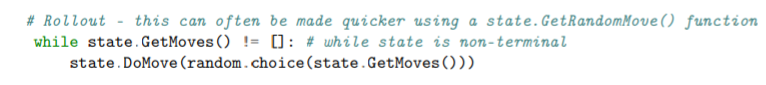
1. **Methodology**

In this project, the finial goal is to train and create a strong AI OXO game player that can defeat the original AI player. The original AI player is hard coded which let it can make random chooses. After testing in 100 round of games, the AI which trained by reinforcement learning will have a better performance than the original one.

The original AI will be used to generate training dataset. It can automatically play OXO games and give output on the console. It is using Monte Carlo tree search, it is a kind of searching method which will make random search at first time, and update the weight of each node of the tree. At the beginning the whole tree only have a root node which represent the begin of the game. When AI is playing the game, it will start searching the whole tree structure to fine a node which is suitable for current situation. Every time when this tree is dealing with a new situation, it will record it and make a new leaf node until the end of the game. This node is generated base on the random chooses. After the tree have already record all the possible situations, it will start making decision and start a backpropagation. 

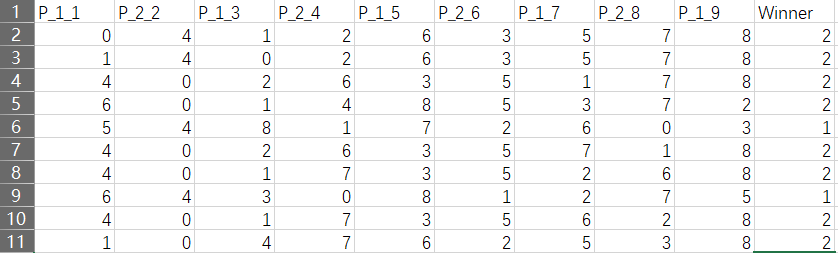
During the backpropagation, if a leaf node can lead to a better outcome, all the parent nodes will have a higher weight so that they have more chance to be picked when AI facing this kind of situation. Meanwhile the tree will also memorise the total visit time of a node so if this node most of time will not have a good ending, it will have a lower probability to be chosen.

Besides, the play record will be collected to train a neural network with reinforcement learning. Before the Monte Carlo tree complete searching, this network will be use to predict a possible next step for player and guide the search algorithm. Base on this it can take a better move than the original random chooses.

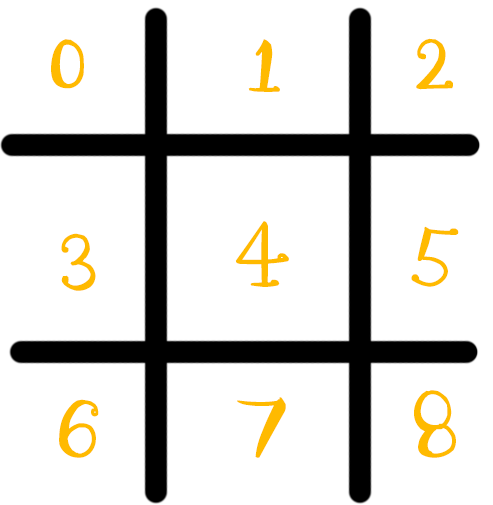
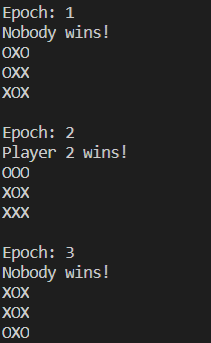


After each 10 iterations, the whole algorithm will be upgrade and trained by the last 10 round of games. Then the performance will be evaluated and check if the performance have been improved or not.

The dataset should look like this:



In the P\_X\_Y, X means the id of player, it is an integer between 1 to 2. Y means the index of movement; it is between 1 to 9. And the final attribute is the id of winner, it marked which player win this round of game. In this game, the game board is split to 9 parts, each part has a specific id. The dataset will record which block the player choose for each step.



1. **Experiments**

The experiment will be divided into 3 parts. The first is building the experiment environment. In this environment, the program should define all the possible moving chooses and game state. Different game has different game state; it will contain all kinds of operation. It also defined the rule of this game, and how to judge the result. Secondly, we also need to build the tree structure as well as the game states. The tree structure will package functions which can be used to add new child node or remove it from the tree. At last we need a training system which can get the data from game and use it to update the neural network.

1. **Discussion**

Evaluation of the OXO game is simple, it can be achieving by list all possible result of the game and compare the game bored with it. After decided which player win the game, it will be added to the winning time parameter of this player, the player which have a higher winning time will be recognized as having a better performance.

1. **Conclusion**

Although there is uncertainty in this project such as the performance of the final result and the training procedure, this project can be implemented.

1. **Reference**

|  |  |
| --- | --- |
| [1] | CSEE, “CE888 Assignment 1,” University of Essex, 2019. |
| [2] | “Project 4: Decision Trees for Expert Iteration (Reinforcement learning),” University of Essex, 2019. |
| [3] | Yahoo!, “Google's New AlphaGo Breakthrough Could Take Algorithms Where No Humans Have Gone,” 19 10 2017. [Online]. Available: https://finance.yahoo.com/news/google-apos-alphago-breakthrough-could-095332226.html. [Accessed 8 2 2019]. |
| [4] | DeepMind, “Mastering the Game of Go without Human Knowledge,” *Nature,* 2017. |

1. **Plan**

|  |  |
| --- | --- |
| **Procedure** | **Time(working day)** |
| Make project plan | 1 |
| Build the game environment | 2 |
| Collect data from the game | 1 |
| Build the training procedure | 4 |
| Test the performance | 1 |
| Problem modification | 2 |
| Generate the report | 2 |