CE888 Assignment 1: Project Proposal on Decision Trees for Expert Iteration

Rohit Venugopal
School of Computer Science and Electronic Engineering
University of Essex
Colchester, UK
rv19514@essex.ac.uk

***Abstract*—Recent success in artificial intelligence agents for games can be credited to the approach which combines supervised and reinforcement learning. This work proposes to combine the popular Monte Carlo Tree Search with other classifiers such as Decision Trees or Neural Networks to play the game of Othello. The main idea behind this approach is to try and mimic the dual-process theory, which states that human reasoning has two distinct types of thinking. In the project, the MCTS portion of the agent shall be responsible for planning the moves by considering the future, while the classifier shall guide the search and help generalise the task.**

I. Introduction

Artificial Intelligence in games has always been a hot topic for research, and the concept of biasing MCTS to improve its performance has seen a lot of attention since AlphaGo [1] defeated the world Go Champion.

Creating Agents which use purely Supervised Learning can be quite difficult since expert knowledge is sometimes required in creating datasets and rules for the games. However, while Reinforcement Learning has shown to achieve good performance in various games, it can still take time for the agent to learn the game on its own, before it achieves a level of proficiency.

Recently there has been a lot of progress in combining both Supervised Learning with Reinforcement Learning to create different agents for various games. These methods and agents have shown the ability to defeat the best existing agents as well as human champions.

This project aims to develop agents which use classifiers like Decision Trees or Neural Networks to bias the Monte Carlo Tree Search (MCTS) Algorithm, to play the game of Othello. The Monte Carlo Tree Search will be used to achieve a lookahead search of the game, while the classifier (Decision Tree or Neural Network), shall replace the rollout function of MCTS, and help in predicting the best move at the particular stage of the game.

This document is structured as follows. Firstly, a brief explanation of Othello, the game, along with a thorough discussion of related work done in the past is presented in section II. The data generation process and the core techniques utilised is explained in III. The approach taken to compare the various agents, as well the evaluation conditions taken are described in detail in sections IV and V respectively. Finally, section VI concludes the paper.

II. Background

A. Othello

Othello or Reversi is a popular, two player board game that was first invented around 1883. The players are assigned coloured tokens, black and white. The game is played on an 8x8 board and starts off with the initial configuration as shown in 1. When it is a players turn, they can place their respective token on the board, and in the process, if the tokens outflank the opponent's tokens, then those are flipped over. The game typically ends when either the board is filled, or both the players pass. The goal of the game is for a player to have a majority of their colour tokens displayed on the board.

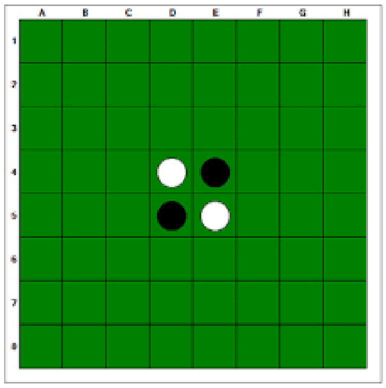

Fig. 1. Othello board and initial configuration

B. Literature Review

Games and developing agents for games has been a major part of AI research history. Board games in general have precise rules, which are easy to formalize and usually have perfect information; which makes it a good choice for research and development of various AI algorithms.

One of the most popular algorithms used to create game agents is the Monte-Carlo Tree Search (MCTS) algorithm. MCTS is a tree search based technique which is capable of searching over many iterations and providing the best possible move for the player. A main feature of MCTS is the fact that it is 'anytime'. That is, the algorithm can be stopped at anytime, and will still have a solution ready to the problem.

Benbassat and Sipper [2] implemented a combination of MCTS and evolutionary algorithm to play the game of Othello. While the MCTS was running, the current board configuration was given to the evolutionary algorithm to choose best possible moves and help guide the MCTS algorithm. Jain, Verma et al [3] proposed an agent which made use of the XCS (eXtended Classifier System) framework to create an agent. The XCS framework uses rules that evolve over time, while playing the game multiple times. It was also shown to offer advantage over other Reinforcement Learning techniques due to its rule based format.

In recent years, the idea of combining MCTS with other techniques has really caught on, especially after AlphaGo defeated the World Go Champion. AlphaGo [1] combined Reinforcement Learning with Deep Neural Networks. The deep Neural Networks implemented helped guide the MCTS algorithm, and predicted its own move selection, rather than doing it randomly. Similar work was also done by [4], where they presented Expert Iteration; an approach which combined the problems of generalisation and planning, by combining MCTS with a convolutional neural network.

Liskowski et al. [5] implemented a convolutional neural network that would take as input the board configuration for Othello, and output the best move that should be taken. The model trained was able to beat all the existing Othello agents, including Edax [6], the best open source Othello playing agent. Soemers, Piette and Browne [7] proposed a biased MCTS which made use of a linear function approximator, instead of the typical Deep Neural Network. While this approach doesn't exhibit exceptional results, it certainly has a few advantages such as interpretability, low utilization of resources and a shorter training time.

III. Methodology

A. Dataset

Data was generated by letting pure MCTS agents play 30 games of Othello. Each 10 games in the 30 games used 500, 300, and 200 iterations respectively. Also both players (agents) used the same number of iterations while generating data, so that both of them have an equal chance of winning. If any one player has less number of iterations, since they are not exploring as much, there is a possibility that they have a higher probability of losing the game.

The data collected includes features such as the player number, the current status of the board and the best move that can be played. Each position on the board is recorded as a feature, and is encoded by a number; 0 if empty, 1 if Player 1 has a token there, and 2 if Player 2 has a token there. The data can be viewed by accessing the GitHub repository for the Data Science Lab, within the Assignments sub-directory. The Repository can be accessed either by following this Link or by following the link mentioned in the Project Plan Section.

Since an 8x8 Othello board is used, there are effectively 64 positions, however 4 of them are used for the initial configuration. So to create a classifier to predict the best move, we need to create a multi-class classifier for 60 classes.

B. Techniques Implemented

The project will start by using pure MCTS agents to play the game of Othello. These agents will also be used to generate data necessary for the future stages.

The main aim of this project is to build a classifier that can predict the best possible moves from a given game state; so as to bias the Monte Carlo Tree Search Algorithm, and improve its performance. By biasing the MCTS algorithm, the rollout function shall now be replaced by a classifier that is able to predict the best results and choose the appropriate move, rather than doing so randomly; which will result in a more guided search.

The project will be done in a certain number of iterations. That is in each iteration, previous agents shall play the game and generate data, and this data shall be used to build the next classifier for the next agent. At the end, each of the agents shall be analysed to see the impact of biasing the MCTS algorithm with a classifier, as well as to see the degree of improvement amongst the various agents.

IV. Results

Initially, Decision Trees as classifiers shall be used to bias the MCTS algorithm. The new agent shall be trained on previous data, and then be used to generate more data, so as to build more classifiers in each iteration. This step will also be implemented by using a combination of Neural Networks and Decision Trees to bias MCTS.

In each iteration, the current agent shall be made to play with all the previous generation agents, and analysis shall be done to see the improvement of the agents through the iterations, and also between pure MCTS, MCTS and Decision Trees, and the combination of MCTS, Neural Networks and Decision Trees.

V. Discussion

Since this is not a purely Supervised Learning problem, we cannot use any of the standard metrics to evaluate the performance of the agents. Due to this, to evaluate performances, the idea of a competition shall be used.

A competition shall consist of around 20 games of Othello played by various Agents, and the winning rates for each agent shall be used to compare if its improving or not. That is, newer agents shall be expected to have a higher winning rate when playing against an older version of itself.

Pitting different agents against each other in a competition and analysing the winning rates should provide a clear understanding of the ability of each agent, and should show the degree of improvement.

We expect to see each new generation of the biased MCTS agent to perform better than its older version. Also the MCTS, Neural Network and Decision Tree approach is expected to perform better than the MCTS and Decision Tree approach. Both of the above mentioned approaches should be remarkably better than the pure MCTS agent.

VI. Conclusions

The game of Othello, while not rated on the same level of difficulty as Go, is still considered to be one of the most difficult strategic games, that can take humans years to master. Since it is a board game, and has no hidden information, clear and easy rules can be formalized, by which an agent can be created to play the game.

While majority of the earlier agents used approaches such as Brute Force, MiniMax search, or even MCTS, this project aims to combine MCTS with a classifier to help guide the search. The resulting agent is expected to be able to outperform existing MCTS agents, and even be able to win in games against humans. The future of artificial intelligence in games definitely lies in the utilising both supervised learning and reinforcement learning together, in such a way, so that they both complement each other, and can produce remarkable results.

References

[1] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[2] A. Benbassat and M. Sipper, "Evomcts: Enhancing mcts-based players through genetic programming," in *2013 IEEE Conference on Computational Inteligence in Games (CIG)*, pp. 1–8, IEEE, 2013.

[3] S. Jain, S. Verma, S. Kumar, and S. Aggarwal, "An evolutionary learning approach to play othello using xcs," in *2018 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, 2018.

[4] T. Anthony, Z. Tian, and D. Barber, "Thinking fast and slow with deep learning and tree search," in *Advances in Neural Information Processing Systems*, pp. 5360–5370, 2017.

[5] P. Liskowski, W. Jaśkowski, and K. Krawiec, "Learning to play othello with deep neural networks," *IEEE Transactions on Games*, vol. 10, no. 4, pp. 354–364, 2018.

[6] R. Delorme, "Edax (othello program)," 2004.

[7] D. J. Soemers, É. Piette, and C. Browne, "Biasing mcts with features for general games," in *2019 IEEE Congress on Evolutionary Computation (CEC)*, pp. 450–457, IEEE, 2019.

Project Plan

This project plan can be seen depicted in the Gantt Chart below in Fig 2. Work on the project started in early February, and is expected to finish by the middle of April, along with a full detailed report on all the techniques implemented, and analysis of the results. The link to the repository can be accessed by either clicking here or following this url: https://github.com/Rohitv97/Data-Science-CE888

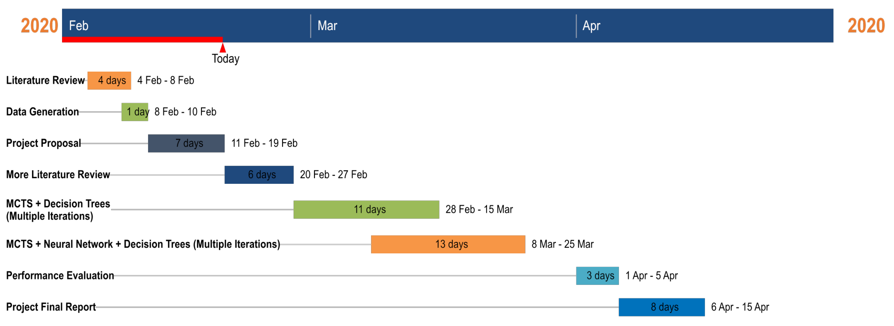

Fig. 2. Gantt Chart Depicting Project Plan