Expert Iteration in OXO using UCT as an Expert Agent

Robert Marc James-Stroud
School of Computer Science and Electronic Engineering
University of Essex
Colchester, UK
Email: rj18801@essex.ac.uk

Abstract

I. Introduction

General Game Playing (GGP) is a field of research in AI involving programs that play games such as chess and Go [1]. Historically GGP focuses on two-player zero sum games with finite states. Programs developed for playing games used to be specialised (Deep Blue), only able to play that one game often with handcrafted rules.

Programs such as Deep Blue have little value in AI research, they are able to evaluate a state for one game only before making a decision about the moves [1]. However more recently researchers have begun developing AI agents that are able to take a ruleset and state at runtime and play games to successful completion. Genesereth describes general game players as:

> '*Systems able to accept descriptions of arbitrary games at runtime and able to use such descriptions to play those games effectively without human intervention*' [1]

As general game playing agents are unable to predetermine any policies about the game it is going to be playing, it is essential that a classifier is determined quickly, an algorithm that has seen great success in this area is Monte Carlo Tree Search (MCTS). Neural networks on the other hand are not trained quickly, they require lots of training data until they can output a result with high success.

Humans approaching a game for the first time use two kinds of thinking - dual process theory. The first aspect of this system is a fast thinking, intuition. The second aspect is slow thinking, reasoning [2]. Exploiting dual processes theory for should result in strong agents independently trained from human influence removing any human play bias from the agents learning.

Reinforcement learning (RL) agents take actions without looking ahead and tree searches, to determine the best action must evaluate branches, with even simple games like OXO having a high branching factor. Expert Iteration uses MCTS to train a neural network, which in turn guides the tree search [2].

II. Background

Supervised learning has been common place in AI, using datasets provided by human experts which agents will then try to replicate. Having humans involved in deployment of agents, according to Hui is troublesome [3]. Therefore it is advisable to remove the human bias from training in all aspects. The benefit of this is two-fold, not only is the human removed from the agent, the agent will also be able to discover non-human approaches which might be of more strategic value than imitating and trying to beat a human based agent. Another benefit to using non-human expert datasets is that expert human datasets are 'often expensive, unreliable or simply unavailable' [4].

If the human datasets are available. and considered reliable; using them provides a ceiling on the performance of an agent. RL removes this by being trained on their own experience, allowing them to surpass human expertise [4].

Both [2] and [4] use a very similar RL process to training their agents. [2] coined the term Expert Iteration (ExIt). Expert Iteration mimics the human learning process, dual-process theory. Dual-process theory has two processes. An automatic implicit happening at the subconscious level - intuition, and an explicit conscious logical and reasoning process. Humans when first encountering a new game exploit both processes [5].

Imitation Learning (IL) focuses on copying an expert demonstrations [3]. Previously stated in this report is that humans are not a desirable component in the training of these agents, therefore an expert will need a form of intuition to be able to provide a demonstration the neural network can imitate. [2] and [4] use MCTS as their expert agent, MCTS is suited for this as it will simulate as many state-actions as it can before reaching an execution budget, this lookahead eliminates part of the problem with neural networks - neural networks do not [2].

System 1 in dual-process theory can be thought of as a heuristic based process. MCTS is heuristics based therefore is an ideal candidate for this. Other tree search algorithms are available: Monte Carlo, α-β or a greedy search [2]. A greedy search is not ideal for looking ahead at future states as it will take the highest value move it can take now, ignoring potentially better nodes that might exist further down the tree on another branch.

α-β pruning seeks to limit the search space by cutting off parts that do not look promising early, part of the training is giving the neural network examples of bad play which by cutting off poor actions early, α-β does not do.

MCTS on the other hand over time converges on mini-max, while still exploring some poor state-action pairs, which can be stored for future use by the neural network as an example of what not to do. MCTS given enough time will find a good move if one exists.

IL revolves around having an expert, which for [2] and [4] is MCTS (probably UCT) and an apprentice, the neural network that will aid in evaluating state-action pair determined from the expert. As training the neural network continues it will effectively generalise the policy determined by MCTS, with corrections to inaccurate intuitions [2].

The policy network is capable of biasing the search towards promising areas of the search space through modification to the UCT formula through addition of a bonus [2].

III. Methodology

Instead of using a policy network to evaluate a state-action value, this project seeks a classifier which can be created quickly. Neural networks cannot be trained quickly so are not suitable for this purpose. However with the modern hardware MCTS can quickly establish good and bad moves, building a decision tree which can be used to do essentially the policy network's job in ExIt and Alpha Go Zero.

IV. Experiments

The first experiment run was generating some base data to compare with which future experiments could be compared. Over ten thousand games of OXO were played with two UCT agents, agent one having 1000 iterations and agent two 100. The effectiveness of UCT is present in this data, with over 90% of the games ending in a draw as can be determined from the logs of each game. Agent one had a 7% victory rate, with agent two losing all games that did not end in a draw.

V. Conclusion

Decision trees would be an effective policy for agents that are facing a game for the first time. The policy can be quickly built without domain knowledge through experimentation of state-action pairs using MCTS. The action from a state can be recorded and turned into a decision tree to bias future searches towards more promising areas of the search space, eliminating wasted search time increasing the effectiveness of both the poly, the decision tree and MCTS, creating a closed learning loop.

This fully eliminates any human interference with the learning process, allowing for super-human capabilities as described by [4]

References

[1] M. Genesereth, "Overview of general game playing," 2005.

[2] T. Anthony, Z. Tian, and D. Barber, "Thinking fast and slow with deep learning and tree search," *CoRR*, vol. abs/1705.08439, 2017. [Online]. Available: http://arxiv.org/abs/1705.08439

[3] J. Hui.

[4] 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, p. 354, 2017.

[5] J. S. B. Evans, "Heuristic and analytic processes in reasoning," *British Journal of Psychology*, vol. 75, no. 4, pp. 451–468, 1984.