Masterthesis

Investigation of possible improvements to increase the efficiency of the AlphaZero algorithm.

Christian-Albrechts-Universität zu Kiel
Institut für Informatik

angefertigt von: **Colin Clausen**
betreuender Hochschullehrer: Prof. Dr.-Ing. Sven Tomforde

Kiel, 20.7.2020

Selbstständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Literatur und Hilfsmittel angefertigt habe.

.....
Colin Clausen

Contents

1 Introduction

Games have been used for a long time as a standin of the more complex real world in developing artifical intelligence. Beating humans at various games has often been viewed as a milestone.

(TODO a few sentences here about progress on various milestone games).

In March 2016 a computerprogram called AlphaGo for the first time in history has defeated a top human player in the board game Go [1]. Go had eluded attempts at super human level play for a very long time, Louis Victor Allies attributes [2] this to the large game tree size of 10^{360} possible games, compared to 10^{120} in chess [4], but also to the way humans use their natural pattern recognition ability to quickly eliminate most of the often 200 or more possible moves and focus on few promising ones.

This combination of the usage of hard-to-program pattern recognition with an extremely large game tree has prevented computers from reaching top human strength through game tree search algorithms based on programmed heuristics. AlphaGo solved this issue by using the strong pattern recognition abilities of deep learning and combining them with a tree search, allowing the computer to learn patterns, similar to a human, but also search forward in the game tree to find the best move to play.

Further development of the AlphaGo algorithm yielded the AlphaZero algorithm, which significantly simplified AlphaGo, allowing learning to start with a random network and no requirements for human expert input of any kind.

In the following thesis I want to investigate further possible improvements to reduce the computational cost of using AlphaZero to learn to play games.

(TODO here one could state some key results, once they exist...)

This thesis is structured as follows: First I will look at previous work, starting with the basis of AlphaZero, Monte Carlo Tree Search, moving onto the various versions of AlphaZero with previously suggested improvements. Then a list of novel improvements will be described. Finally an extensive set of experiments will be presented, first establishing a baseline performance, then showing the results on the novel improvements.

2 Previous work

In this section previous work relevant to AlphaZero will be presented.

2.1 Monte Carlo Tree Search

2.2 AlphaZero

The AlphaZero algorithm [6] is a simplification of AlphaGoZero [7] and AlphaGo [5]. AlphaGo used a complicated system involving initialization with example games, random roleouts during tree searches and used multiple networks for different tasks. AlphaGoZero drastically simplified the system by only using a single network for all tasks, and not doing any roleouts anymore, instead the network evaluation for the given positions is directly used.

The difference between AlphaGoZero and AlphaZero is mainly that AlphaGoZero involved comparing the currently trained network against the previously known best player by letting them play a set of evaluation games against each other. Only the best player was used to generate new games. AlphaZero skips this and just always uses the current network to produce new games, surprisingly this appears to not give any disadvantage, learning remains stable.

The main advantage of the "Zero" versions is that they do not require any human knowledge about the game apart from the rules, the networks are trained from scratch by self-play alone. This allows the algorithm to find the best way to play without human bias which seems to slightly increase final playing strength. Additionally it allows to use the algorithm for research of games for which no human experts exist, such as No-Castling Chess [3].

2.3 Extensions to AlphaZero

Many improvements to the AlphaZero algorithm have been proposed, typically aiming at reducing the extreme computational cost of learning to play a new game.

3 Evaluated novel improvements

3.1 Network modifications

3.2 Playing games as trees

3.3 Automatic auxilary features

4 Experiments

4.1 Baselines

4.1.1 AlphaZero implementation

4.1.2 Extended AlphaZero

4.2 Results on novel improvements

4.2.1 Network modifications

4.2.2 Playing games as trees

4.2.3 Automatic auxilary features

4.2.4 Evolutionary hyperparameters

[6]

References

[1] http://www.straitstimes.com/asia/east-asia/googles-alphago-gets-divine-go-ranking. Accessed: 2020-05-17.

[2] Louis Victor Allis et al. *Searching for solutions in games and artificial intelligence*. Ponsen & Looijen Wageningen, 1994.

[3] Vladimir Kramnik. Kramnik and alphazero: How to rethink chess. https://www.chess.com/article/view/no-castling-chess-kramnik-alphazero. Accessed: 2019-11-29.

[4] Claude E Shannon. Xxii. programming a computer for playing chess. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 41(314):256–275, 1950.

[5] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[6] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[7] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017.