AIND Project 2 / Josef Schodl / 09/2017

Research review on:

Mastering the game of Go with deep neural networks and tree search

https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf

Goals

Goal of the research project has been to master the game of Go with deep neural networks and tree search. The DeepMind team aimed to identify and apply new techniques that would be successful in game playing for games with huge search spaces. They called their product AlphaGo.

The specific degree of complexity is not exactly defined, though the DeepMind team refers to the Go search space being massively larger compared to the search space of chess. A game of Go using the typical 19x19 board has a branching factor of 250, a state-space complexity of 10^170 and a game tree complexity of 10^250, while the values for chess are 'only' 35, 10^47 and 10^123.

These huge numbers seemed to make the game unconquerable by AI techniques for a very long time. However, the DeepMind team accepted the challenge in order to progress the entire field of AI for also other problems beyond game playing to solve.

Several new techniques, respectively their combinations and way of use have been introduced with AlphaGo:

- 'Value networks' to evaluate board positions and 'policy networks' to select moves.
- Combination of supervised learning from human expert games and reinforcement learning from games to train deep neural networks.
- The Monte Carlo Tree Search combined with the value and policy networks.

Results

AlphaGo for the first time in history defeated a professional human Go player, years ahead of anyone expected it to happen. Previously considered as maybe the biggest challenge for AI in game playing, AlphaGo plays constantly at the level of the strongest human players.

AlphaGo showed a 99.8% winning rate against other computer Go programs, leaving all those programs way behind in results. Research of the DeepMind team showed also the level of computing power required for this achievement: The final version of AlphaGo used 40 search threads, 48 CPUs and 8 GPUs. Beyond that, a distributed edition with 40 search threads, 1202 CPUs and 176 GPUs has been implemented.