Summary of the Paper AlphaGo by the DeepMind Team

Andre Marinho

This is a summary of the article “Mastering the game of Go with deep neural networks and tree search”.

The article starts explaining the challenges of building an adversarial agent for the game of Go, due to its enormous search space and the difficulty of evaluating board positions and moves.

AI based on search methods has been used agents on games like chess, checkers, and othello. In large games (game depth and number of legal moves per position), the exhaustive search is infeasible, and the depth of search needs to be truncated, and replaced by a approximate value function that predicts the outcome from that state.

In more recent state-of-the-art AI game agents for the game of Go, Monte Carlo tree search (MCTS) has been used to estimate the value of each state in the search tree.

The DeepMind team developed the AlphaGo program using a novel approach of having ‘value networks’ to evaluate board positions, and ‘policy networks’ to select moves. These deep neural networks were trained combining supervised learning from human expert games, and reinforced learning from games of self-play.

With this approach they were able to defeat a human professional player on a full-sized game of Go, for the first time.

The neural network is trained using a pipeline of several stages of machine learning:

The first stage of the pipeline uses supervised learning (SL) for training the ‘policy network’.
The second stage aims to improve the policy network by reinforcement learning (RL).
The final stage (RL of ‘value networks’) focuses on position evaluation, estimating a value function that predicts the outcome from position s of games played by using policy p for both players.

The search is done combining the policy and value networks in a MCTS algorithm that select actions by lookahead search. Thus, combining deep neural networks and tree search.

AlphaGo was able to outperform by far all the best Go programs, and was able to defeat by 5 to 0, a professional 2 dan and three times European Champion Go player, Fan Hui.

Another highlight was the use of an asynchronous multi-threaded search. The final version uses 40 search threads, 48 CPUS, and 8 GPUs.

A distributed version, that uses multiple machines, was also developed. .