A Research Review of the AlphaGo game-playing AI

by Teruyuki Takahashi

Summary
The game of Go has long been viewed as the most challenging of classic games for artificial intelligence owing to its enormous search space and the difficulty of evaluating board positions and moves. Here we introduce a new approach to computer Go that uses ‘value networks’ to evaluate board positions and ‘policy networks’ to select moves. These deep neural networks are trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play. Without any lookahead search, the neural networks play Go at the level of stateof-the-art Monte Carlo tree search programs that simulate thousands of random games of self-play. We also introduce a new search algorithm that combines Monte Carlo simulation with value and policy networks. Using this search algorithm, our program AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0. This is the first time that a computer program has defeated a human professional player in the full-sized game of Go, a feat previously thought to be at least a decade away.

All games of perfect information have an optimal value function, v*(s),which determines the outcome of the game, from every board positionor state s, under perfect play by all players. These games may be solved by recursively computing the optimal value function in a search tree containing approximately bd possible sequences of moves, where b is the game’s breadth (number of legal moves per position) and d is its depth (game length). In large games, such as chess (b≈35, d≈80)1 and especially Go (b≈250, d≈150)1, exhaustive search is infeasible2,3, but the effective search space can be reduced by two general principles. First, the depth of the search may be reduced by position evaluation: truncating the search tree at state s and replacing the subtree below s by an approximate value function v(s)≈v* (s) that predicts the outcome from state s. This approach has led to superhuman performance in chess4, checkers5 and othello6, but it was believed to be intractable in Go due to the complexity of the game7. Second, the breadth of the search may be reduced by sampling actions from a policy p(a|s) that is a probability distribution over possible moves a in position s. For example,Monte Carlo rollouts8 search to maximum depth without branching at all, by sampling long sequences of actions for both players from a policy p. Averaging over such rollouts can provide an effective position evaluation, achieving superhuman performance in backgammon8 and Scrabble9, and weak amateur level play in Go10.

Supervised learning of policy networks

For the first stage of the training pipeline, we build on prior workon predicting expert moves in the game of Go using supervisedlearning.

Reinforcement learning of policy networks

・The second stage of the training pipeline aims at improving the policynetwork by policy gradient reinforcement learning

Reinforcement learning of value networks

The final stage of the training pipeline focuses on position evaluation,estimating a value function vp(s) that predicts the outcome from positions of games played by using policy p for both players

Evaluating the playing strength of AlphaGo

To evaluate AlphaGo, we ran an internal tournament among variantsof AlphaGo and several other Go programs, including the strongestcommercial programs Crazy Stone13 and Zen, and the strongest open source programs Pachi14 and Fuego15.

Discussion

In this work we have developed a Go program, based on a combination of deep neural networks and tree search, that plays at the level of the strongest human players, thereby achieving one of artificial intelligence’s grand challenges.