Build a Game-Playing Agent, research review: Mastering the game of Go with deep neural networks and tree search

Octavio Navarro-Hinojosa

1 Goals or techniques introduced

All games of perfect information may be solved by recursively computing an optimal value function in a search tree containing approximately b^d possible sequences of moves, where b is the game's breadth (number of legal moves per position) and d is its depth (game length). By reducing the depth using position evaluation, and the breadth by sampling actions with a given policy, some approaches allowed improved performance some games, but not in Go. The game of Go is one of the most challenging of classic games for artificial intelligence because of its enormous search space and the difficulty of evaluating board positions and moves.

In order to allow a computer program to play Go at the level of the strongest human players, the authors introduced AlphaGo, a new program based on 'value networks', which evaluate board positions, and 'policy networks', which select moves. These deep convolutional neural networks were trained with a specific pipeline of supervised learning (SL) human expert games, and reinforcement learning (RL) from self-play games, in order to learn better strategies and select better moves.

For the first stage of the training pipeline, the authors used a SL policy network to predict expert moves in Go. The 13-layer policy network was trained from 30 million positions from the KGS Go Server, so that it maximized the likelihood of selecting a human move a in state s. This stage could be thought of as if AlphaGo learned to play from human teachers.

The second stage of the training pipeline aimed at improving the policy network by policy gradient RL. The RL policy network was identical in structure to the SL policy network, and its weights were initialized to the same values. The authors played games between the current policy and a randomly selected previous iteration of the policy network in order to prevent overfitting.

The final stage of the training pipeline focused on position evaluation, estimating a value function $v^P(s)$ that predicts the outcome from position s of games played by using policy p for both players. The authors estimated the value function using a RL policy network. The weights of the value network were trained by regression on state-outcome pairs (s, z), using stochastic gradient descent to minimize the mean squared error (MSE) between the predicted value, and the corresponding outcome.

AlphaGo combined the policy and value networks in a Monte Carlo tree search (MCTS) algorithm that selects actions by lookahead search. Evaluating policy and value networks required several orders of magnitude more computation than traditional search heuristics. To efficiently combine MCTS with deep neural networks, AlphaGo used an asynchronous multi-threaded search that executes simulations on CPUs, and computes policy and value networks in parallel on GPUs.

2 Results

To evaluate AlphaGo, the authors ran an internal tournament among variants of AlphaGo and several other Go programs. All of these programs were based on high-performance MCTS algorithms. The results of the tournament suggest that single machine AlphaGo is many dan ranks stronger than any previous Go program, winning 494 out of 495 games (99.8%) against other Go programs.

To provide a greater challenge to AlphaGo, the authors also played games with four handicap stones (that is, free moves for the opponent); AlphaGo won 77%, 86%, and 99% of handicap games against the softwares Crazy Stone, Zen and Pachi, respectively. The distributed version of AlphaGo was significantly stronger, winning 77% of games against single-machine AlphaGo and 100% of its games against other programs.

Finally, during a match against Fan Hui, a professional Go player, AlphaGo won 5 games to 0 by selecting positions more intelligently, using the policy network, and by evaluating them more precisely, using the value network. The authors claimed that this was an approach that is closer to how humans play.