Review of Mastering the Game of Go with Deep Neural Networks and Tree Search[1]

Yuanyuan Pan, June 2017

Brief Summary:

This paper introduced how the team build and trained the deep neural networks, value networks and policy networks, of AlphaGo, as well as use the new search algorithm that combines Monte Carlo simulation with value and policy networks to improve searching efficiency.

Agent Structure:

- Deep Convolutional Neural Networks: They pass in the board position as a 19*19 image, and use convolutional Layers to construct a representation of the position. They use these neural networks to reduce the effective depth and breadth of the search tree: evaluating positions using a value network, and sampling actions using a policy network.
- Monte Carlo Tree Search (MCTS): Monte Carlo rollouts are used to estimate the value of each state in a search tree. The strongest Go agent use MCTS enhanced by Policies that are trained to predict human expert moves, to narrow the search to a beam of high-probability actions and to sample actions during rollouts.

Training Pipeline: Supervised learning of Policy Networks (SL-PN) – Reinforcement Learning of Policy Networks (RL-PN) – Reinforcement Learning of Value Networks (RL-VN):

- SL-PN: Input (s) is a simple representation of the board state, output (a) is a final soft-max layer with probability distribution over all legal moves. The policy network is trained on randomly sampled state-action pairs (s,a), using stochastic gradient ascent to maximize the likelihood of the human move a selected in state s. (13 layer PN trained from 30 million positions)
- RL-PN: identical structured to SL-PN and weights are initialized to the same as SL-PN. Play games between the current PN and a randomly selected previous iteration of the PN, to stabilize training by preventing overfitting to the current PN. Reward function are used and trained towards optimizing the final outcome that is win or lose.
- RL-VN: Train a value function that focus on position evaluation, that predict the outcome from position s of games played by using policy p for both players. This neural network has a similar architecture to the policy network, but outputs a single prediction instead of a probability distribution. They train the weights of the value network by regression on state-outcome pairs

[1] Silver D, Huang A, Maddison C J, et al. Mastering the game of Go with deep neural networks and tree search[J]. Nature, 2016, 529(7587): 484-489.

(s, z), using stochastic gradient descent to minimize the mean squared error (MSE) between the predicted value $v_\theta(s)$, and the corresponding outcome z.

Searching with Policy and Value Networks:

Alpha Go use Monte Carlo methods, combine policy and value networks to select actions. SL-PN's probability is used as prior distribution (which performs better than RL-PN), and the tree is traversed by simulation starting from the root state in Monte Carlo's way. The probability distribution, action values and visit counts are updated during the process to encourage exploration.

Evaluation the Playing Strength:

Alpha Go wins 99.8% games against previous Go agent, and win most games with four handicap stones. Even without rollouts AlphaGo exceeded the performance of all other Go programs, demonstrating that value networks provide a viable alternative to Monte Carlo evaluation in Go. However, the mixed evaluation (λ=0.5) performed best. In 2015, AlphaGo won the match 5 games to 0 competing against Fan Hui, a professional 2 dan.