The paper, Mastering the game of Go with deep neural networks and tree search, presented methods used to surpass prior performance limitations of Chinese Go game-playing algorithms (depth-first minimax search with alpha-beta pruning).

Policy Networks

The policy network contains 13 layers of convolutional weights and rectifier nonlinearities, terminated with soft-max over all legal moves. Initial training was performed using 30 million positions stored on the KGS Go Server.

Reinforcement learning is then used to train an identically structured policy network. The reward function is zero for all non-terminal steps, +1 for winning and -1 for losing. Stochastic gradient ascent updates weights at each timestep reflecting favor towards the direction that maximizes the expected outcome function.

Value Networks

The position evaluation network is a deep neural network similar to the policy network, except the output is a single prediction. The value network's purpose is to estimate the optimal value function. It is trained using state-outcome pairs where the weights are updated using stochastic gradient descent (to minimize the error between the predicted value and corresponding outcome).

The Monte Carlo Tree Search (MCTS)

The MCTS combines the value and policy networks to select actions using lookahead search. Each game tree is descended (without backup) and the edges' action value, visit count, and prior probability is updated. A mixing function is used to combine leaf node evaluations, and each edge accumulates the visit count and mean evaluation of all simulations passing through the edge. The algorithm selects the most visited move from the root position.

Finding Highlights

Policy network structural performance revealed that larger networks achieve better accuracy but were slower to search. Training the value networks initially on full game sequences led to overfitting, so randomly generated games were used to create novelty.