AlphaGo Summary

The article, “Mastering the game of Go with deep neural networks and tree search” describes a system developed by the authors that combines deep neural networks with state of the art Monte Carlo Tree Search (MCTS) algorithms designed to play the game of Go. The result is a new type of game playing AI, AlphaGo, that vastly outperforms even the best current Go AI, to the extent that it was able to beat the European Go champion 5-0 in an official match, a feat previously thought to be a decade away.

Go, a game with a 19x19 grid board, has for some time stood out in the field of game AI. Common tree search algorithms such as minimax with alphabeta pruning have led to superhuman performance in games such as chess, checkers, and othello. However this approached was believed to be intractable in Go due to the complexity of the game. More recently, the development of the MTCS algorithm has led to Go AIs able to compete in strong amateur play when combined with policies trained to predict human expert moves.

The authors use deep learning methods to train several neural networks that they incorporate into MCTS to produce AlphaGo. These networks include a Supervised Learning (SL) policy network trained directly on expert human moves along with a fast policy that can rapidly sample actions during rollouts, a Reinforcement Learning (RL) policy network directed toward the outcome of winning games rather than maximizing predicted accuracy, and value network that predicts the winner of games played by the RL policy network against itself.

The SL policy network was trained using 30 million positions from the KGS Go server. The resulting network predicted expert moves with an accuracy of 57%, compared to 44.4% for current state of the art from other research groups. The faster but less accurate rollout policy was trained using a linear softmax of small pattern features, resulting in an accuracy of 24.2% using just 2 us to select an action compared to 3 ms for the policy network.

Next the policy network was improved using policy gradient reinforcement learning. The RL network was identical to the SL network in structure and was initialized with the same weights. To optimize the policy network, games were played with the current policy vs a random selection from a previous iteration using a reward function of +1 for winning and -1 for losing. Weights were adjusted after each iteration using gradient ascent to maximize the expected outcome. The resulting RL policy network was capable of winning 80% of games vs the SL policy network.

The final stage of training focused on position evaluation via the value network. The value network was similar in structure to the policy network except that it outputs a single prediction rather than a probability distribution. To avoid overfitting, this network was trained on 30 million distinct positions from games between the RL policy network and itself.

After the networks are trained, AlphaGo combines the policy and value networks in an MCTS algorithm that selects actions by lookahead search. The resulting algorithm is capable of defeating previous Go programs 99.8% of the time on a single machine. A stronger distributed version can defeat the single machine AlphaGo 77% of the time, and was used to defeat the European Go champion, Fan Hui, 5 games to 0. This was a feat considered to be one of AI’s “grand challenges”.