Austin Totty
19 July, 2017
Udacity AI Nanodegree

Summary of "Mastering the game of Go with deep neural networks and tree search"

The AlphaGo algorithm developed by DeepMind introduced a technique to evaluate an action given a game state for Go. The algorithm uses a weighting of neural networks developed through both supervised and reinforcement learning and Monte Carlo rollouts. A image of the board state is passed to a convolutional neural network which creates a representation of the position. A value network is used to evaluate positions and a policy network is used to sample actions.

The first step in the training pipeline is a supervised learning algorithm that outputs a probability distribution of all legal moves given a game state which is trained from a database of known moves from professional Go players. A 13-layer network is used, which balances the increased accuracy with slower evaluation of larger networks. A separate, faster policy is also trained which is about half as accurate.

In the second step of the pipeline, the previously developed policy network is trained using reinforcement learning to optimize the weights. This training on the optimized policy network is done with self-play against one established policy network which is chosen randomly to reduce overfitting. This reinforcement learning policy won 80% of games played against the supervised learning policy.

The final step of the pipeline consists of training the value function that outputs a single prediction instead of a probability distribution. To prevent the value network from overfitting to known games in Go match databases, a new self-play data set was generating consisting of 30 million distinct positions.

To select a move, the policy and value networks are combined in a Monte Carlo tree search algorithm using lookahead search. A bonus is added to action values that decays with repeated visits to a node to "encourage exploration."

Implementing the algorithm consists of running simulations on CPUs and computing and value networks in parallel on GPUs using 40 threads. A local implementation consisted of 48 CPUs and 8 GPUs. A distributed version used 1,202 CPUs and 176 GPUs.

The result of the technique that was AlphaGo's dominance over previous Go-playing agents; AlphaGo won 494 of 495 non-handicap games against other programs. The value networks alone also outperformed existing game-agents, which led the authors to suggest neural networks are a viable alternative to Monte Carlo tree search methods. With the mixed approach, AlphaGo won a match against a 2 *dan* Go player, Fan Hui, 5 games to 0. Since the original article was published, AlphaGo won a match against Ke Jie, the highest ranked Go player in the world.