AlphaGo Research Paper Review

Udacity Artificial Intelligence Nanodegree

Will Trott

Go is an ancient Chinese board game where players take turns placing black and white stones on a 19x19 board until both players pass their turns because they can make no more profitable moves. Because of the number of legal moves per play and length of the game are so high, traditional methods such as truncating the search tree and replacing leaf nodes with approximate values and Monte Carlo rollouts were unable to beat professional Go players. The team behind AlphaGo leveraged advances in performance of convolutional neural networks to create representations of board positions. The neural network is the major improvement to the methodology, as the network that plays the game still takes advantage of the Monte Carlo Tree Search method that most previous solutions leverage.

The first step in their process was to train two policy networks. The first of these networks is a supervised learning policy network that alternates between using convolutional layers with specified weights, rectifier nonlinear layers, and a softmax layer that outputs a distribution considering all legal moves. The second policy network is called a rollout policy that can be used to quickly get values, but is not very accurate, which is trained in a quicker manner using linear softmax and responses that have very few patterns for the network to look for.

For the next step, the policy layer is improved using reinforcement learning where the current policy layer plays against a previous version of itself, using a reward of +1 if the current version wins, and -1 if it loses. After this, stochastic gradient ascent is used to update weights. Using this method, the team then compared their reinforcement learning policy network against the best open-source Go program, Pachi. This version of the algorithm won 85% of the games against they played against Pachi when the previous best result was 11%.

Now that a robust policy network was trained, the team used it to estimate a value function that is based on the RL policy network. The resulting neural network behaves similarly to the RL policy network, but returns a single value instead of a distribution. The weights are trained via stochastic gradient descent. Once this value network has been trained, AlphaGo uses it and the policy network in a MCTS algorithm. This algorithm selects a node at each level that maximizes the sum of an action value with a bonus that is proportional to the prior probability, but decreases every time that that node is visited. When the lookahead search reaches a leaf node, the leaf node is evaluated using the weighted sum of the value network with the result of the fast rollout policy. It turns out the that the optimal value for the weights is to use 0.5 for both.

After the network was trained, it played against all the best commercial programs as well as a professionally ranked player. In order to efficiently use both MCTS and the CNN,

AlphaGo uses asynchronous multi-threaded search which resulted in victory 100% of the time against all of top commercial products and a 5-0 series against the three time winner of the European Go championships.