Review of AlphaGo paper

This is a review of "Mastering the game of Go with deep neural networks and tree search" by Silver et al., 2016.

Research goal

The goal of this research was to create the best Go playing computer possible, using state-of-the-art hardware and AI techniques. More formally, their goal was to create the best possible approximation of an algorithm that plays perfect Go. However, playing perfect Go is infeasible due to the large (both wide and deep) search space of the game. Therefore, they instead set their sights lower to an achievable, but still high, goal of creating a Go program that could compete with the highest ranked Go players in the world, a feat predicted to not be attained for at least another decade. These goals were reached by the authors.

New techniques used

Several new techniques were used by the authors to achieve their goals. The two major innovations in this study were: creating a pipeline for training neural networks that combines several machine learning techniques, and combining these neural networks with Monte Carlo tree search (MCTS). Both techniques are summarized below.

AlphaGo (the name they gave their program) used multiple neural networks to aid in evaluation of board states while exploring future moves. A neural network is a machine learning technique that uses a network of nodes (neurons) to make predictions after being trained on a data set. In this project, a neural network was trained to predict the next move that the average Go expert would make given a specific board configuration. Once fully trained, they used this neural network as the starting model for their next neural network, which used a different machine learning technique, reinforcement learning. This neural network played games of Go against random, previous versions of itself and used gradient ascent to constantly improve the next version. Finally, they used the reinforcement-learning policy network to train a value network. Instead of predicting the next move, like the expert-trained and self-trained networks, this value network instead predicted the outcome of the game from the current board state. In the final version of AlphaGo, only the neural network trained on expert data, and the value network were used for evaluating game moves.

The other new technique developed was combining these neural networks with Monte Carlo tree search (MCTS). MCTS is an optimization technique commonly used when creating game-playing AI that limits the number of moves that must be explored, similar to alpha-beta pruning. MCTS works by repeatedly applying random moves to a board until an endgame state is reached, then backpropagating the results of that path back up the tree. By doing this many times, you can approximate a probability distribution of moves that are likely to result in victory. They were able to combine the neural networks they developed with MCTS by using the neural networks to weight the likelihood that a node in the game tree will be expanded based on how likely it is that it's a good move.

Summary of results

Using the techniques described, the researchers were able to achieve their goal and create a Go playing computer that could consistently beat any existing Go program, and even beat human, professional Go players. Since the paper was published, AlphaGo has gone on to defeat the best Go players in the world. Specifically, the best version of AlphaGo they created, which used distributed computing, beat the existing Go programs 100% of the time, and beat Fan Hui, a professional 2 dan Go player, 5 games to 0 in a formal match.