Review of the DeepMind Team's AlphaGo Paper

AlphaGo goals. While AlphaGo was being developed, the strongest Go programs were based on a technology that achieved strong amateur play, and it was believed that it would be at least a decade until an artificial intelligence could beat a professional player in a full game of Go. This belief was due to the enormous search space of Go and the complexity in evaluating moves. (According to DeepMind, "There are more possible positions in Go than there are atoms in the universe," making it a googol times more complex than Chess). The goal of the AlphaGo project was to leverage deep neural networks, and some of the best techniques used in previous Go programs, in order to develop a competitive player.

Techniques introduced by AlphaGo. The key to dealing with intractable search spaces is finding intelligent ways to *not* search the whole space, without missing optimal or near-optimal solutions. In terms of searching game trees, that means intelligently reducing the breadth and depth of the search. AlphaGo does this by combining Monte Carlo tree search with novel policy and value networks.

The policy network that AlphaGo uses is a neural network that helps to intelligently reduce the breadth of the search for an optimal move by focusing AlphaGo on moves that are most likely to result in optimal play. That is, it assigns probabilities to all moves available to the player from a game state, with higher probabilities given to moves that are more likely to be optimal. AlphaGo uses this information to just search moves that are likely to be optimal for the player. The policy network was first trained with supervised learning (SL) techniques on a dataset of millions of board positions from games played by human experts. The policy network learned to predict moves most likely to be played by a human expert from a board state; it was able to predict expert human moves with 57% accuracy. Then the policy network was further improved by using reinforcement learning (RL) techniques on games of self-play. This move resulted in subtle shift in the goal of the learning process: instead of learning to predict expert moves, AlphaGo was learning to predict winning moves. After being trained on games of self-play, the policy network was played against the earlier version of the network that was trained only on the dataset of expert moves. The fully trained network beat the earlier version in 80% of the games.

The value network is a neural network that helps intelligently reduce the depth of the search for an optimal move. After reaching a certain depth in its search of the game tree, AlphaGo substitutes further search of the tree with an evaluation of the game-tree node by the value network (and combines this evaluation with the results of some fast Monte Carlo rollouts to endgame). (AlphaGo's use of a value network corresponds to our use of heuristic-evaluation functions in the AIND-Isolation project.) The score returned by the value network gives an indication of how strong of a position a game-tree node is for the player, and it informs AlphaGo's game-tree search for an optimal move. According to the AlphaGo paper, the value network "has a similar architecture to the policy network, but outputs a single prediction instead of a probability distribution." The value network was trained on 30 million distinct board

positions, sampled from games between the policy network and itself, and it learned to predict with high accuracy the outcome of a game from a board position.

The development of AlphaGo used data from human and self-play games of Go. And it used general SL and RL techniques, not tailored to the problem at hand, which is impressive because other programs for Go and Chess rely heavily on tailored techniques, with handcrafted expert knowledge being programmed into the system.

AlphaGo results. Before playing AlphaGo against a human champion, the DeepMind team ran a tournament among Go programs, which included variants of AlphaGo and some of the strongest commercial Go programs. The single-machine and distributed versions of AlphaGo dominated the other programs, with the former winning 99.8% and the latter 100% of its games. When the single-machine and distributed versions of AlphaGo were played against each other, the distributed version won 77% of the time. Finally, the distributed version of AlphaGo was played against Fan Hui, winner of several European Go championships. In the end, AlphaGo became the first program to defeat a pro player on a full-size board, winning 5 games to 0.