Research Report

In 2016, DeepMind (a Google-owned AI research lab) announced the creation of a new AI agent for the board game Go, called AlphaGo. Go, like the Isolation game used in our project, is a finite, two-player game of perfect information, however, unlike Isolation, Go has a high branching factor that leads to an enormous search space. In fact, many experts believed that Go was too complex for current AI techniques to perform well. AlphaGo was built with neural networks trained through reinforcement learning along with Monte Carlo tree search (MCTS) to analyze game boards and select good moves. AlphaGo far exceeded the performance of all earlier Go engines, and bested the European Go champion 5 games to 0.

Fundamentally, AlphaGo performs the same actions as our Isolation agent. AlphaGo searches through the game tree and estimates the value of the most likely end states under the assumption of an adversarial opponent seeking to minimize the possible rewards. However, minimax search is intractable for Go because the high branching factor leads to an extremely large game tree even at shallow depths. AlphaGo substitutes Monte Carlo tree search -- an approximate search algorithm based on repeated simulations -- to achieve greater depth along promising lines of play than minimax could provide.

In order to guide the search, AlphaGo uses a convolutional neural network consisting of convolutional layers and rectifier nonlinearities leading to a final softmax layer for output probabilities. [Silver, et.al] This acts as a policy network to predict lines of play that are consistent with human expert play and optimized to select for winning the game. Additionally, AlphaGo uses a value network to predict the outcome of moves based on games simulated using the policy network for both players.

While these networks significantly outperform other Go agents (the trained policy network alone won 85% of games against other state-of-the-art Go engines [Silver, et.al]), the downside is that evaluating the policy and value networks requires several orders of magnitude more computation than traditional search heuristics. AlphaGo uses an asynchronous multi-threaded search that executes simulations on CPUs, and computes policy and value networks in parallel on GPUs. The distributed version of AlphaGo runs on 1,202 CPUs and 176 GPUs.

AlphaGo was a remarkable advance in the state-of-the-art for game playing agents. Successfully combining policy and value networks with MCTS resulted in expert-human level performance for the first time in any Go agent. The trade-off of increased depth with an improved heuristic compared to traditional methods like minimax also contributed to computational efficiency without sacrificing agent performanc