AlphaGO by the DeepMind Team

Techniques introduced:

DeepMind (Google) present a new approach to computer Go that uses:

- 'value networks' to evaluate board positions and
- 'policy networks' to select moves

These deep neural networks are trained by a novel combination of:

- supervised learning from human expert games, and
- reinforcement learning from games of self-play.

Games like Go are solved by recursively computing the optimal value function $v^*(s)$ in a search tree containing approximately b^d possible sequences of moves. Go has an enormous search space and the difficulty of evaluating board positions and moves. In large games, such as chess ($b \approx 35$, $d \approx 80$) and especially Go ($b \approx 250$, $d \approx 150$) , exhaustive search is infeasible, but the effective search space can be reduced by two general principles:

- First, the depth of the search may be reduced by position evaluation: truncating the search tree at state s and replacing the subtree below s by an approximate value function $v(s) \approx v^*(s)$ that predicts the outcome from state s (but it was believed to be intractable in Go due to the complexity of the game)
- Second, the breadth of the search may be reduced by sampling actions from a policy $p(a|s)$ that is a probability distribution over possible moves a in position s.

They also introduce a new search algorithm that combines Monte Carlo simulation with value and policy networks. Depth-first minimax search with alpha–beta pruning has achieved superhuman performance in chess, checkers and othello, but it has not been effective in Go. Monte Carlo tree search (MCTS), alternative approach to minimax search, uses Monte Carlo rollouts to estimate the value of each state in a search tree. As more simulations are executed, the search tree grows larger and the relevant values become more accurate. With Monte Carlo tree search they also simulate thousands of random games of self-play.

They employ a similar architecture (deep convolutional neural networks) for the game of Go, passing in the board position as a 19×19 image and use convolutional layers to construct a representation of the position. Then use these neural networks to reduce the effective depth and breadth of the search tree: evaluating positions using a value network, and sampling actions using a policy network.

Paper's results:

Using this search algorithm, our program AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0. This is the first time that a computer program has defeated a human professional player in the full-sized game of Go, a feat previously thought to be at least a decade away.

During the match against Fan Hui, AlphaGo evaluated thousands of times fewer positions than Deep Blue did in its chess match against Kasparov, compensating by selecting those positions more intelligently, using the policy network, and evaluating them more precisely, using the value network—an approach that is perhaps closer to how humans play. Furthermore, while Deep Blue relied on a handcrafted evaluation function, the neural networks of AlphaGo are trained directly from gameplay purely through general-purpose supervised and reinforcement learning methods.

Source: https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf