Alpha Go Paper Summary

AIND Feb Cohort
Pete Peeradej Tanruangporn
Due March 16, 2017

Challenge and Results: AI has achieved superhuman performance in popular board games such as checker, backgammon, and chess, but until 2016, no programs have achieved such a feat for Go. Compared to chess with the approximate breadth and depth of 35 and 80, the search space of Go is much bigger, with its breadth and depth respectively at 250 and 150. Most simple scientific calculators will simply return "infinity" for Go's full depth search space, and about 3.35e^(123) for chess. This paper is the first to offer a program that can achieve superhuman performance, as demonstrated by its formal victory over the European champion, ranked at professional 2 *dan*, five times in a roll. Against other Go programs, its win rate is at 99.8% out of 495 games. The program is also the first to create effective move selection and evolution functions based entirely on generalized algorithms.

Method and techniques

1. Supervised Learning Policy Network (SL PN)
Representing the Go board as a 19x19 image, over 30 million board positions were collected from the expert human players, ranked 6 to 9 *dan* only, on the KGS Go Server. These moves are then put through Convolutional Neural Networks to acquire "abstract and localized" representation of positions and areas of the board, maximizing for the most likely next move. At this point, it could offer a probability distribution for the next move, with an accuracy of 57%)

2. Fast Roll-out Policy (Fast Policy)
The team also trained an extremely quick version of the (SL PN), with less than half the accuracy at 24%, but a particle fraction of execution time (2 nanosecond vs 3 mili-second).

3. Reinforcement Learning Policy Network (RL PN)
The team improved the Policy Network by updating their gradients with reinforcement learning. Reinforcement learning pitches the most current Policy Network against a random pool of previous iterations, updating the weights of the gradients to maximize winning. This helps orient the gradient away from reproducing the most likely learned professional move, to winning them. This achieved an 85% win rate against the previous state-of-art Go program, ranked at 2 amateur *dan,* without having to do any search.

4. Reinforcement Learning Value Network
The value network is an approximate of an optimal value function, used to evaluate a move on the board. It trains on state-outcome pairs (s,z) to minimize the difference between predicted value and true outcome. Trained on the KGS dataset, the function merely remembered the outcomes of past games, but couldn't generalize to new plays. The team thus created 30 million new positions, each from separate games, by playing the RL PN against itself, to create a new value network. The paper claims that its accuracy is superior to Monte Carlo rollouts using Fast Policy, and close to Monte Carlo rollouts using RL PN, but with 15,000 less computation.

5. Searching
Each move is informed by the Monte Carlo Tree Search where the SL PN is first used to select and score a set of moves, after which the program looks further ahead and combine the score from the value network's evaluation and from quick simulation with Fast Policy for *T* steps. The evaluation scores then updates the mean evaluations of a move's subtree, and finally a move is selected based on how many high value actions subtrees there are (in another word how many times move is visited). This helps avoid moves that might have outlier max action value.