A review of the Alpha Go seminal paper

By Emmanuel Ezenwere

Alpha Go is a computer program developed by the Deep Mind team. It uses deep learning and the Monte Carlo tree search algorithm to play the game of Go with a performance that's better than any human expert player, defeating the human European Go champion by 5 games to 0 in 2015. It also achieved a 99.8% winning rate against other Go programs.

The architecture of the Alpha Go AI involves two deep convolutional neural networks, one called the value network whose outputs are value representing approximations of the outcome of a game given a board state, the other is a Policy network trained to classify board positions.

The game of Go has a very large search space considering the dimensions of the board and possible moves, it has an approximate breath and depth of 250 and 150 respectively. Hence searching to the end game will lead to evaluating approximately 250^{150} move which is not feasible especially given the time threshold of 5 secs for a move.. Besides having a greater search space and possible moves, I realized that the game of Go was not so much a different from the game of chess & Isolation. They all can be represented by a Tree with a branching factor b and depth d. So the challenge with Go was to come up with an evaluation heuristic good enough to beat any human player, the innovation in Alpha Go was the use of deep learning to learn adversarially such a Heuristic.

Here I will like to explain the key implementations in Alpha Go; The Monte Carlo Tree search, the Policy and Value networks.

The Monte Carlo Tree search: a heuristic search algorithm. In Alpha Go, this was used to generate the game tree. I do not have a strong grasp of how exactly it works or how it is implemented. A light wikipedia research reveals that it uses backpropagation to readjust values(weights) assigned to member nodes allowing for a more efficient search in the game of Go when compared to the depth first limited search with alpha-beta pruning as mentioned in the paper. The Monte Carlo Tree search implementation in Alpha Go uses a double approximation of the optimal value function.

The Policy Networks: These are deep neural nets trained using a combination of supervised learning & reinforcement learning to predict moves a human expert would make given a set of positions, by making predictions the Policy Networks reduces the breadth of the search tree. It's architecture consists of a 13-layer deep CNN trained on 30 million Go game positions.

The Value Networks: These are also deep neural nets trained using a combination of supervised learning & reinforcement learning to predict how likely a player would win the game given a board state, the Value Networks reduces the depth of the tree as we do no longer have to search to end game to determine the outcome..

Closing Remark: Despite how remarkable the accomplishment of Alpha Go was, I do feel AlphaGo could have performed much better while implementing fewer algorithms; A flaw I observed in the Architecture of the Alpha Go is in training a deep neural network using samples from Human Expert players (this is a general limitation with supervised learning), I believe this creates an inherent limitation to it's game strength. In other words Alpha Go is only as good as the sum of all it's human expert trainers. My strategy would be; the game of Go I assume has clear rules, and an unambiguous way to determine a win or a loss, therefore with a cost function that reflects win rate we can design a deep neural network to keep playing against itself until the cost function is minimized and hence the win rate is at the very maximum. I am currently working on this for the game of Isolation as a test of this idea.

References: AlphaGo by the DeepMind Team, https://en.wikipedia.org/wiki/Monte_Carlo_tree_search