Mastering Go with AlphaGo Article Summary

The game of Go is one of the examples where applying a search tree algorithm, due to the number of legal moves per position and its game length, would take millions of years to visit all nodes and find the best path. Because of this issue, the team decided to join neural networks and the tree search algorithms. This made possible to have and agent capable of winning against other Go programs with a rating of 99.8%, and was able to defeat the human European Go champion by 5 games to 0.

As the depth and the breadth of the game is enormous the team decided to use deep convolutional neural networks that using the board with positions as input creates a board representation of the position.

As a first step the agent uses a supervised leaning policy network (SL). The policy network alternates between convolutional layers with weights and rectifier nonlinearities with a final softmax layer that outputs the probability distribution for all legal moves. that is trained using expert human moves, that generates high-quality gradients.

The generated gradients from the SL are then passed to another neural network, policy network (PL), that randomly selects previous iteration of the current policy and using reinforcement leaning improves the supervised network by optimizing the final outcome of games of self-play.

To help the previous two networks a third one is added, value network, that focusing on the position evaluation, predicts the winner of the games played by the reinforcement learning network against itself. The value network is based on the RL policy network but instead returning a probability distribution returns a single prediction. The problem with this network was that it started to memorize the game outcomes leading to overfitting. As solution was to generate a self-play data set consisting in 30 million distinct positions, each sample from a separate game.

To complete the agent AlphaGo efficiently combines this networks with Monte Carlo tree search algorithm (MCTS), a heuristic search algorithm that combines the generality of random simulation with the precision of tree search.

The strength of AlphaGo was evaluated running tournaments against several Go programs including Crazy Stone, Zen, Pachi and Fuego, that are considered the strongest Go programs based in high-performance MCTS algorithms and to which of them was allowed a computation time of 5 seconds per move. As result from the tournament AlphaGo got the greatest rank than other previous Go program, winning 494 out of 495 games.

For this first time AlphaGo proved that combining neural networks with search algorithms, in this case Mote Carlo, it is possible to create Agents that are able to play at the human level. This opens the door to apply neural networks to other game agents and improve them.