Mastering the game of Go with deep neural networks and tree search

Introduction and Goals:
This paper explains about a new approach used to play the game of Go, which has been known as the most challenging game for artificial intelligence. The main idea for this approach is to use deep neural networks as 'value networks' to evaluate board positions and 'policy networks' to select moves. These deep neural networks are trained by combination of supervised learning from human players, and reinforcement learning from games of self-play. Also a new search algorithm is introduced that combines Monte Carlo simulation with value and policy networks.

Games of perfect information have an optimal value for every board position and may be solved by recursively computing the optimal value in a search tee containing approximately b^d possible sequences of moves where b is the game's breadth and d is its depth. In large games like Go, exhaustive search is infeasible. The effective search space can be reduced by position evaluation to reduce the depth, and sampling actions from a policy $p(s| a)$ to reduce the breadth. The strongest current Go programs are based on Monte Carlo Tree Search (MCTS) and use these reduction techniques.

Methods:
In the new approach the board position is passed as a 19*19 image to a convolutional deep neural network to construct a representation of the position. The neural networks are used to reduce the effective breadth and depth of the search tree, evaluation positions using a value network and sampling actions using a policy network.

The neural networks are trained in a pipeline consisting of 3 stages of machine learning.

The 1st stage is supervised learning (SL) 'policy network' which learns directly from expert human moves. The input to the policy network is a simple representation of the board state. The policy network is trained on randomly sampled state-action pairs (s, a) to maximize the likelihood of the human move a selected in state s.

The 2nd stage is a reinforcement learning (RL) policy network that improves the SL policy network by optimizing the final outcome of games of self-play. Games are played between the current policy network and previously iteration of policy network.

The 3rd stage will be a 'value network' that predicts the winner of games played by the RL policy network against itself. This stage neural network has a similar architecture to the policy network but outputs a single prediction instead of probability distribution.

Finally the AlphaGo program combines the policy and value networks with MCTS algorithm that selects actions by likelihood search. To efficiently combine MCTS with deep neural networks, AlphaGo uses asynchronous multi-threaded search that executes simulations on CPUs and computes policy and value networks in parallel on GPUs.

Result:
Tournament was ran among variants of AlphaGo and several other Go programs including strongest commercial and open source programs, which all were based on high-performance MCTS algorithms. The result of tournament suggests that single machine AlphaGo is stronger than other Go programs, winning 99.8% of the games. Also AlphaGo

won the match 5 to 0 when playing with human professional Go player. This is the first time that a computer Go program has defeated a human professional player in full game of Go, which was previously believed to be at least a decade away.