Research Summary: Mastering the game of GO with Deep Neural Networks and Tree Search

Due to its enormous search space and difficulty in evaluating the game state, action and reward (s,a,r) triplet, the game playing of Go has been viewed as one of the most challenging aspects of AI. Due to a very high branching factor in Go, implementing minimax /alpha-beta pruning algorithms are practically infeasible. Google came up with a novel structure that combines tree search with deep neural networks to develop a superhuman game playing AI agent. In this algorithm, effective move selection and position evaluation functions are developed using Deep Neural Networks trained by a combination of supervised and reinforcement learning. These evaluation functions are then combined with Monte Carlo Tree Search to estimate the value of each state and help in reducing the overall search space. The detailed explanation is provided below.

The first step in training pipeline consisted of training a supervised learning of policy network to predict expert moves. A representation of the board state is fed to this policy network that uses a convolutional neural network (13 layers) with a final softmax layer that outputs the probability distribution of all the legal moves available to the player. This policy network is then trained using **stochastic gradient ascent** to **maximize** the likely hood of selecting the best moves. The second stage in the training pipeline aims at improving the policy network by policy gradient reinforcement learning. Experience replays are used in this reinforcement learning section to break the serial correlations when playing the game which reduces overfitting. In the final stage of training pipeline, the value networks are used which helps in predicting the outcome from the current state of the board using the strongest policy chosen by the policy networks. This means that value function approximates the best moves based on the action policy chosen by the policy network.

Once policy and value networks are trained, the AlphaGo combines them in a Monte Carlo Tree Search (MCTS) algorithm that selects the action using lookahead search. AlphaGo implements an asynchronous policy and value MCTS algorithm, to integrate large policy and value networks. Each edge of the search tree stores an action value function Q(s, a), prior probability P and visit count. The game tree is searched in a large number of simulations and is composed of 4 steps:

Selection: The MCTS travels the game tree by selecting the edge with the maximum number action value Q(s,a) plus a bonus that depends on a stored prior probability P

Expansion: If any node is expanded, it is processed once by Supervised Learning policy network to get prior probabilities P for each legal action.

Evaluation: Each node is evaluated in two ways first by value network and then by running a Fast Rollout policy at the end of the game.

Backup: Action values Q are updated by values collected during evaluation step that tracks mean values of all evaluations and value network output in the subtrees below the corresponding action

When time dedicated for AlphaGo is over, it chooses the best move by the highest action value Q found based on above evaluation steps.

AlphaGo was a major breakthrough in the field of AI and game playing as it overcame many conventional problems faced in AI, which included dealing with an intractable search space, finding an optimal function which seems to be infeasible to approximate using action and value functions and finding solutions for a highly nonlinear task. AlphaGo won against a human professional player, Fan Hui who was the winner of multiple European Go championships, and also won 99.8% matches against other Go programs like Crazy Stone, Pachi, and Zen. During the match against Fan Hui, AlphaGo evaluated thousands of times fewer positions than Deep Blue did in its chess match against Kasparov; and compensated by choosing its actions more intelligently using policy network and evaluating the actions more precisely using value networks which led to choosing an approach which more resembles the game play of humans and not just taking advantage of the computational power.