Research Review of AlphaGo paper

GOAL: This paper introduce a new approach to computer Go that uses 'value networks' to evaluate board positions and 'policy networks' to select moves.

Techniques overview: These deep neural networks are trained by a novel combination of supervised learning from human expert games, and reinforcement learning from games of self-play. This paper also introduces a new search algorithm that combines Monte Carlo simulation with value and policy networks. It uses 19x19 images to construct a representation of the position and use convolutional neural networks for the game to reduce the effective depth and breadth of the search tree: evaluating positions using a value network, and sampling actions using a policy network.

The first technique is supervised learning (SL) policy network directly from expert human moves and a fast policy p_π that can rapidly sample actions during rollouts. As a **result**, the network predicted expert moves on a held out test set with an accuracy of 57.0% using all input features, and 55.7% using only raw board position and move history as inputs, compared to the state-of-the-art from other research groups of 44.4% at date of submission

The second technique is policy network by policy gradient reinforcement learning (RL). As a **result**, when played head-to-head, the RL policy network won more than 80% of games against the SL policy network. Tested against the strongest open-source Go program, Pachi a sophisticated Monte Carlo search program, ranked at 2 amateur *dan*
on KGS,that executes 100,000 simulations per move. Using no search at all, the RL policy network won 85% of games against Pachi.

The final stage of the training pipeline focuses on position evaluation, estimating a value function $v_p(s)$ that predicts the outcome from position s of games played by using policy p for both players. The naive approach of predicting game outcomes from data consisting of complete games leads to overfitting. So they use **the third technique** to solve the overfitting issue, generated a new self-play data set consisting of 30 million distinct positions, each sampled from a separate game. Each game was played between the RL policy network and itself until the game terminated. **Results:** Training on this data set led to MSEs of 0.226 and 0.234 on the training and test set respectively, indicating minimal overfitting.

AlphaGo combines the policy and value networks in an MCTS algorithm that selects actions by lookahead search. Evaluating policy and value networks requires several orders of magnitude more computation than traditional search heuristics. To efficiently combine MCTS with deep neural networks, AlphaGo uses **the fourth technique** of an asynchronous multi-threaded search that executes simulations on CPUs, and computes policy and value networks in parallel on GPUs. **The results** of the tournament suggest that single machine AlphaGo is many *dan* ranks stronger than any previous Go program, winning 494 out of 495 games (99.8%) against other Go programs. To provide a greater challenge to AlphaGo, they played games with four handicap stones (that is, free moves for the opponent); AlphaGo won 77%, 86%, and 99% of handicap games against Crazy Stone, Zen and Pachi, respectively. The distributed version of AlphaGo was significantly stronger, winning 77% of games against single-machine AlphaGo and 100% of its games against other programs.