A summary on

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

By Noyin Lawal

Current Go programs are based on Monte Carlo search tree (MCST) that are enhanced by policies trained to predict human expert moves. These policies are used to narrow the search in a search tree to a beam of high-probability actions and to sample actions during rollouts. This method produces strong amateur play. However this produces shallow policies or value functions based on linear combinations of input features.

AlphaGo program tackles this problem by combining deep neural networks and MTCS. The deep neural networks are composed of two parts, a policy network to handle move selections and a value network to evaluate board positions.

The Policy network was trained in two stages. First by using supervised learning (SL) of expert moves and second using reinforcement learning (RL) by game of self-play. The data for SL comprised of 30 million positions from the KGS Go Server. The large policy network predicted expert moves on a test set with an accuracy of 57.0% using all input features compared to 44.4% achieved by other research groups. A faster but smaller SL policy network with a speed of 2μs and accuracy of 24.2% was used in place of the larger SL policy network with a much slower speed of 3ms . For the second stage, the policy network was trained by using RL which involved testing against randomly selected previous iterations of itself. An 80.0% success rate was achieved when the RL Policy network played against the SL policy network. Also tested against the strongest open-source Go program, Pachi, the RL policy network achieved an 85.0% success rate. Previous state of the art, based only on supervised learning of convolutional networks won 11.0% of games against Pachi.

The value network was also trained using RL with a generated self-play data 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 was terminated. This produced an evaluation accuracy that was consistently more accurate compared to the Monte Carlo rollouts using the fast rollout policy. A single evaluation of the value network combined with the RL policy network also matched the accuracies produced by Monte Carlo rollouts but used 15,000 times less computation.

AlphaGO combines the policy network and value networks in an MTCS algorithm that selects actions by lookahead search. Each edge in the search tree stores an action value, visit count and prior probability. The tree is traversed by simulation starting from the root node. At the end of each simulation, the value actions and visit counts of all traversed edges are updated. Once search is complete, the algorithm chooses the most visited move from the root position.

To handle the computations for policy and value networks, Alpha Go employs an asynchronous multi-threaded search that executes simulations on CPUs and computes policy and value networks in parallel on GPUs. The final version of AlphaGo runs on 40 search threads, 48 CPUs and 8 GPUs. Also a distributed version of AlphaGo was implemented that used several machines, 40 search threads, 1202 CPUs and 176 GPUs.

For Evaluation, AlphaGO was tested in a tournament against variants of itself and several Go programs including the strong commercial programs Crazy Stone and Zen and the strongest open source programs Pachi and Fuego. All of these programs are based on high-performance MCTS algorithms. During each game, programs were given 5s of computation time per move. The result 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%). The distributed version of AlphaGo was tested against Fan Hui , a professional 2 dan , and the winner of the 2013, 2014, and 2015 European Go championships. Alpha Go won the match 5 games to 0. This is the first time a computer Go program defeated a human professional, without handicap, in a full game of Go.

AlphaGo is able to attain human level performance with the combinations of neural networks and search trees. AlphaGo evaluated a thousand times fewer positions than Deep Blue did in its chess match against Gary Kasparov. These fewer evaluations by AlphaGo are made possible by intelligently selecting positions using policy networks and evaluating them more precisely using value network