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

Goals:

The go game, unlike other game, has a very large search space so that the normal search algorithm and prune method doesn't works well. This paper gives us a brief introduction of how AlphaGo works and provides the algorithm to tackle with this situation. It works by implementing deep convolutional neural networks that work together in a machine learning pipline, and combining it with Monte Carlo Tree Search algorithm.

Techniques:

AlphaGo use the Monte Carlo tree search combined with supervise/reinforcement learning techniques to deal with the search tree spaces. The traditional Minimax algorithm with alpha-beta pruning can't deal with the breadth and depth in go, while the Monte Carlo method can return a score about how well the game looks like by sampling and aim at the best result. The machine learning algorithms are used to train the policy network and value network which are two networks. Policy network is used for playing piece and value network is used for evaluating the situations. These two networks were combined into the Monte Carlo tree search as heuristics. Policy network is trained by with chess manual with supervised learning techniques and refined by reinforcement learning algorithm. Then another reinforcement learning algorithm is applied to build the value network. After that, the whole pipeline was built.

The first supervised learning policy network has about a max accuracy of 57% with action using 3ms. A faster and less accurate neural network named Rollout Policy also trained with accuracy of 24.2% accuracy and 2us action.

The second reinforcement learning policy network works not to overfitting the current policy. The final network has already about 80% accuracy on game against the previous supervised learning policy network.

The last part of the pipeline is another neural network with focuses on position evaluation. The network estimates the outcome of games played by using previous network and predicting game outcomes from complete games. So it is based on the whole game rather than one-time situation to train.

Result:

With all technique being used, the algorithm simulates the games start from root state, and select an action each time based on the value, bonus, count and prior probability. These values are updated with the policy and value networks after reaching leaf nodes in order to maximize the probability to win. After simulation time is over, the algorithm chooses the most possible position to move.

The final version of AlphaGo has 40 search threads, 48 CPUs and 8 GPUs. It was tested against different go game agent and human player. It has on average 99.8% winning rate on other AI agents. And wins all 5 matches against the human go champion.

The whole game agent performs very well in go game and beat the world championship. Also, the calculating time through the game is far less than the deep blue versus Kasparov.

Besides the game playing, AlphaGo gives us the methods and confident to deal with large search space which is a great evolution in AI field.

Reference:

Silver, David, et al. "Mastering the game of Go with deep neural networks and tree search." *nature* 529.7587 (2016): 484-489.

Silver, David, et al. "Mastering the game of go without human knowledge." *Nature* 550.7676 (2017): 354.