Research Review

For Udacity AIND Game playing project

In this report we will be looking at AlphaGo by DeepMind Team who were able to create a game playing agent for game Go which is able to beat the world' best Go players. So how did they were able to create such an agent? In our Game Playing Isolation agent, we used the tree search for getting all the possible moves for our game isolation and used iterative deepening and alpha beta pruning to prune some of the tree branches to reduce the search space. THen we used different heuristics to give a value to each branch. But In our case the branching factor was very small which made it possible to get all the possible branches. Whereas in the game of Go the branching factor is around 250 and its depth is around 150 which makes it impossible to seintroducedlect a move using the techniques of Isolation agent.

Techniques Introduced

Here two major techniques introduced are :

Monte Carlo Tree Search : This technique has helped in getting an optimal value function. As previous methods were limited to shallow value functions based on a linear combination of inputs.

Reinforcement Learning : This technique helped the AlphaGo to compete against the other instances of itself which helped in building a better player than from only using the supervised learning.

System Design

AlphaGo uses four neural networks and Monte Carlo Tree Search (MCTS). MCTS uses Monte Carlo rollouts to estimate the value of each state in a search tree. With more simulations as search tree grows and improvement in the policy as it converges to give optimal value function. The neural networks used are trained by a combination of supervised learning from human expert moves and reinforcement learning from thousands of random games of self play which are simulated by the state of the art Monte Carlo tree search programs. And finally a new algorithm is made by combining the Monte Carlo Simulations with the value and policy networks.

Training

The neural networks are trained using a pipeline having several stages of machine learning. The first stage is the **Supervised Learning Policy Network** which is a 13 layer deep CNN trained on the 30 million game positions. For a single game position, it predicts the most likely move. The second stage is the **Rollout Policy Network** which can rapidly sample actions during rollouts and helps in narrowing the search to a beam of high probability actions. The third stage is **Reinforcement Learning Policy Network** also predicts the best move or converges to the optimal value function by playing against itself and beating earlier instances of itself. The final stage is the **Reinforcement Learning Value Network** which predicts the winner of the games played by the Reinforcement Learning policy network against itself. Since the network when trained on the Supervised Learning Policy it overfitted. So to improve the generalization it was trained on Reinforcement Learning policy network.

Search with the networks

AlphaGo combines the policy and value networks in an MCTS algorithm that selects actions by lookahead search. In AlphaGo the SL policy network performs better because humans

selects a diverse beam of promising moves whereas RL optimizes the single best move. However the Value function derived from the RL policy network performs better than one derived from the SL policy network. There were four stages in the MCTS algorithm for AlphaGo.

a. *Selection Stage* : In this the tree is traversed to get the edge with maximum Q value.
b. *Expansion Stage* : In this the leaf node is explored by creating a branch and running the SL policy network to get the likely player move.
c. *Evaluation Stage* : In this after simulation is done, leaf node is evaluated in two ways: using the value network derived from RL policy network and by using rollouts till the end of the game using Rollout Policy network, then computes the winner by a evaluation function.
d. *Backup Stage* : In this action value Q is updated to get the mean values of value network and evaluation function for the action in the subtree.

Finally the move with highest Q value is chosen.

Results

a. Both versions of the AlphaGo (Single Machine and Distributed) outperformed all previously existing Go game playing AIs.
b. The Distributed Version was significantly stronger and won 77% of games against Single Machine variant.
c. Even without using the Monte carlo rollouts, the AlphaGo variant with a value network exceeded the performance of all other Go programs.
d. Finally AlphaGo defeated the European champion Go player Fan 5 to 0.

References

a. Mastering the game of Go with deep neural networks and tree search by David Silver et al.