Monte Carlo Tree Search in Go Game

Abhijeet and Surupendu

DA-IICT

May 3, 2018

Introduction to the problem

- The search tree of the game of Go is huge.

Introduction to the problem

- The search tree of the game of Go is huge.
- The best way to design an algorithm that can find winning moves is to use Monte Carlo Tree Search(MCTS). Monte-Carlo simulation provides a simple method for estimating the value of positions and moves.

Introduction to the problem

- The search tree of the game of Go is huge.
- The best way to design an algorithm that can find winning moves is to use Monte Carlo Tree Search(MCTS). Monte-Carlo simulation provides a simple method for estimating the value of positions and moves.
- A heuristic function has to be handcrafted (supervised learning) to make a value function that will estimate the outcome of the game.

What is a policy?

A policy $\pi(s, a)$ is the probability of selecting move a in position s. The value function $Q^{\pi}(s, a)$ to be the expected outcome of a self-play game that starts with move a in position s, and then follows policy π for both players.

Monte Carlo Simulation

$$Q(s,a) = \frac{1}{n(s,a)} \sum_{i=1}^{N} I_i(s,a) z_i \tag{1}$$

- where , where z_i is the outcome of the ith simulation: $z = 1$ if the player wins the game and z $=$ 0 otherwise.
- $I_i(s, a)$ is an indicator function returning 1 if move a was selected in position s during the ith simulation, and 0 otherwise.
- $n(s, a)$ counts the total number of simulations in which move a was selected in position s.

Form of MCTS used : Upper Confidence Tree(UCT)

$$Q^{\oplus}(s,a) = Q(s,a) + c\sqrt{\frac{\log n(s)}{n(s,a)}} \tag{2}$$

where ,$n(s)$ counts the total number of visits to position s, and c is a constant determining the level of exploration

What is a Value Function?

$$Q_h(s,a) = \sigma\left(\phi(s,a)^T\theta\right) \tag{3}$$

The value function is approximated by a linear combination of binary features ϕ with weights θ and σ is the logistic function.

$$\begin{aligned} \delta &= Q_h(s_{t+1},a_{t+1}) - Q_h(s_t,a_t) \\ \Delta\theta_i &= \alpha\delta\phi_i(s_t,a_t) \end{aligned} \tag{4}$$

δ is the TD-error and α is a step-size parameter.

Rapid Action Value Estimation(RAVE)

$$\hat{Q}(s,a) = \frac{1}{\hat{n}(s,a)} \sum_{i=1}^{N} \hat{I}_i(s,a) z_i \tag{5}$$

where, $\hat{Q}(s,a)$ is the average outcome of all simulations in which move a is selected in position s.
$\hat{I}_i(s,a)$ is an indicator function returning 1 if position s was encountered at any step k of the i^{th} simulation, and move a was selected at any step $t \geq k$, or 0 otherwise

Combining UCT and RAVE

- The Monte-Carlo value $Q(s,a)$ is unbiased, but may be high variance if insufficient experience is available.
- The RAVE value $\hat{Q}(s,a)$ is biased, but lower variance .It is based more on experience.
- The RAVE value can be relied on initially, and then a gradual shift to the Monte-Carlo value can be made, by using a linear combination of these values with a decaying weight.
- An equivalence parameter k controls the number of simulations when both estimates are given equal weight.

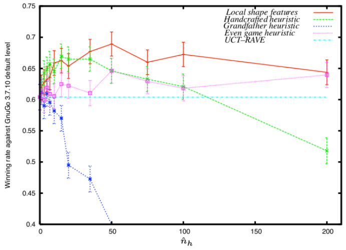
Credit : "Achieving Master Level Play in 9 by 9 Computer Go" , Sylvain Gelly, David Silver

Variance Bias Tradeoff

- The bias-variance tradeoff is a central problem in supervised learning.
- High-variance learning methods may be able to represent their training set well but are at risk of overfitting to noisy or unrepresentative training data.
- Algorithms with high bias typically produce simpler models that don't tend to overfit but may underfit their training data, failing to capture important regularities and therefore models with low bias are usually more complex.
- The solution to this Variance Bias Tradeoff is to use Reinforcement Learning.

Source: Wikipedia Bias-Variance tradeoff

Conclusion and take-away point of the presentation



- The best heuristic was the one in which local features are taken into the value function
- The local shape is the best way to determine the heuristic value function and to train it by Temporal Difference learning.

Credit : "Achieving Master Level Play in 9 by 9 Computer Go" , Sylvain Gelly, David Silver

Scope of Improvement

- Use a better version of MCTS
- Reduce the bias of the algorithm trying to estimate the value function.
- Go from handcrafted heuristic functions (supervised learning) , to unsupervised learning, and then finally to reinforcement learning.
- Increase the size of Monte Carlo simulation episode
- See the results for various sizes of boards like 13x13 or 17x17 or 19x19.

"Achieving Master Level Play in 9 by 9 Computer Go" , Sylvain Gelly, David Silver

"Monte-Carlo Tree Search and Rapid Action Value Estimation in Computer Go" , Sylvain Gelly, David Silver

"MCTS with Information Sharing" , Petr Baudis

"Efficient selectivity and backup operators in Monte-Carlo tree search",R. Coulom