Research Review: AlphaGo

Tianpei Xie

Nov. 30th., 2017

1 Paper review: Mastering the game of Go with deep neural network and tree search

Learning to play games such as chess and Go is a challenging task, since the searching space of these games are very large. The search tree which specify the process of recursive evaluation of the game contains approximately b^d nodes to be visited, where b stands for the branching factor (i.e. number of legal moves for each position) and d denotes the depth of search (i.e. game length). In the paper, "*Mastering the game of Go with deep neural network and tree search*" by David Silver, Aja Huang, etc, the authors provide an efficient learning architecture as well as algorithms to defeat the best human Go expert in the Game.

- **Main contribution**: Traditionally, to train an AI agent to play game, one need to reduce both depth and breadth of the search tree. To reduce the depth, one could replace a sub-tree below some state s by approximate value function of that state to predict the outcome. Also to reduce the breadth, one could use pruning (e.g. alpha-beta pruning in the class).

  In this paper, the authors proposes to use deep convolution neural network to perform pruning and value approximation. This leads to two deep network: a set of policy networks, which perform action selection and sampling and a value network, which provide value approximation.

  Another core technique used is the Monte Carlo Tree Search (MCTS), which uses Monte Carlo rollouts to estimate the value of each state. Both the approximate of the relevant values and the policy to sample the action improves as the tree grow larger. In one aspect, the use of deep neural network is to replace the linear class of value functions to non-linear functions.

- **Pipelines of training**: First, a supervised learning (SL) policy network p_σ is trained using expert human moves. Also a fast policy p_π is trained for MCTS to rapidly sample actions during rollouts. Next, a reinforcement learning (RL) policy network p_ρ is trained to improve p_σ by adjusting the objective from maximizing predictive accuracy to winning games. Here actions are sampled according to the distribution of p_ρ. Finally, a value network v_θ is trained that predicts the winner of game played by the RL policy network.

  When both networks are trained, the searching is performed using MCTS. Each node represents a state-action (s, a) pair as well as an action value $Q(s, a)$. An action is selected by maximizing $Q(s, a) + prior(s, a)$. Here the SL network provides an estimate of prior probability for leaf nodes. The value $V(s)$ of each leaf state is computed using a combination of value network and a fast policy. Then action function $Q(s,a)$ is an empirical mean of the value function $V(s)$.

- **Performance:** Run against other Go program win 99.8%. Run against top human players (Fan Hui, Sedol Lee) It achieves human-level intelligence.
- **Conclusion:** The benefit of using AlphaGo in playing Go is its effectiveness in pruning, action selection and value approximation. It greatly reduce the depth and breadth of the search tree, allowing the system to react in fast and consistent way. It faces a typical challenge of AI: "a challenging decision-making task, an intractable search space and an optimal solution so complex it appears infeasible to directly approximate using a policy or value function." The success of AlphaGo lies in a combination of searching trees and the deep neural network in policy and value approximation.