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

Summary

- Using a new search algorithm that combines Monte Carlo simulation with value and policy networks, a computer program (AlphaGo) has finally defeated a human professional Go player.
- Since Go is a game of perfect information, a game can be solved from every board position, assuming perfect play, by calculating the optimal utility of a move recursively. As it is, however, the search time for the optimal move in Go will be too slow as the number of available moves and game length is too large.
- The search time can however be reduced by narrowing the effective search space:
    - Reducing the depth of the search by position evaluation. This means that the search tree is truncated after a certain point and the subtree is replaced by an approximate value function.
    - Reducing the breadth of the search by sampling actions from a policy that is a probability distribution over possible moves in a position (e.g. Monte Carlo rollouts search to maximum depth without branching at all by using sampling)
- Monte Carlo tree search uses Monte Carlo rollouts to estimate the value of each state in a search tree. Over time, the values become more accurate as the search tree grows and the policy also improves by selecting child nodes with higher values.
- Neural networks (19x19 image with convolutional layers representing the board position) were used to reduce the effective depth and breadth of the search tree.

Training of the neural networks

- Stage 1: Supervised learning of policy networks
    - Trained on randomly sampled state-action pairs and evaluated on the basis on how well it predicted expert moves
    - Purpose: to maximise likelihood of the human move being selected
- Stage 2: Reinforcement learning of policy networks
    - Trained on randomly selected previous iteration of the policy network.
    - Purpose: to maximise expected outcome (reward function: 0 for all non-terminal time steps; +1 for winning; -1 for losing)
- Stage 3: Reinforcement learning of value networks
    - Purpose: to estimate a value function that predicts the outcome from a certain position of games played by using the same policy for both players
        - Value function is approximated using the using the reinforcement learning policy network

Technique: searching with policy and value networks

- AlphaGo selects actions by lookahead search using the policy and value networks.
- Each edge of the search tree stores an action value, visit count and prior probability
    - Prior probability is the output of the supervised learning policy network
- Each leaf node is evaluated in by combining these two evaluations:
    - By the value network
    - By the outcome of a random rollout played until terminal state using the policy
- Algorithm chooses the most visited move from the root position