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

Summary: *This work has introduced a new algorithm that combines Monte Carlo simulation with value and policy networks to be utilized in GO game and that defeated for the first time a human champion!*

Introduction:

- **GO** is the most challenging classic game because:
  - It has an enormous search space.
  - Difficulty of evaluating board positions and moves

- Given infinite resources, the optimal move of a deterministic game is completely determined by recursively searching a tree with possible $\boldsymbol{b^d}$ moves, where **b** is the number of legal moves and **d** is the game length.
- For games with large playing space like **GO**, where $\boldsymbol{b^d}$ soon becomes intractable, certain strategies were introduced to limit the search space and yet produce a reasonable approximation. These include:

    1. Reducing the depth by position evaluation with an approximate scoring functions for deep subtrees.
    2. Sampling actions over possible moves probability distribution, and maybe averaging the outcome.

- Because of the complexity of the **GO** game, only the second solution may provide an acceptable performance, if the sampling strategy was enhanced. In that regard, **Monte Carlo Tree Search** (**MCTS**) is one of the widely utilized techniques to predict human expert moves.

Methodology:

- Utilizing convolutional networks by passing an image board of size 19x19 and allowing the deep network to abstract various information. They constructed two somehow similar networks:
    - "value networks" to evaluate board positions. Outputs probabilities
    - "Policy networks" to select moves. It outputs a single prediction

- ***Training Methodology***
    - supervised learning from human expert games:
        - Series of convolutional layers and nonlinear rectifiers with final softmax layer that outputs the probabilities over all legal moves
        - Stochastic gradient decent was utilized to maximize the likelihood of human move given a selected state.
        - They achieved up to 57% accuracy using all input features.
    - reinforcement learning from games of self-play.
        - To prevent overfitting, the game was played between current selected policy network and a randomly selected one by utilizing a certain rewarding function.
        - Weights are updated using stochastic gradient decent to maximize the expected outcome.
        - For certain games, they achieved around 85% win.
        - They also plotted the MSE value between the predicted value and the actual game.

- ***Searching***
    - Combines policy and value networks in an MCTS algorithm and select actions by lookahead search.
    - The tree is traversed by simulation, starting from the root state.
    - An action is selected in accordance with some prior probability and a visit count for all traversed edges is kept updated.
    - Utilizes multi-threaded search that executes simulations on CPUs and computes policy and value networks in parallel on GPUs.

Results

- They tested their model against various commercial as well as open source programs that already utilizes high-performance MCTS algorithms with a time limit of **5 sec** to compute the next move.
- They reported a wining scenario in 99.8% against other go programs.
- They tried different challenges and their program was winning most of the time.
- They reported that their distributed version is much stronger.
- And for the first time in this game's history, they reported defeating the human European Go champion by 5 games to 0.
- Their algorithm, chooses next moves intelligently and thus requires searching less space

Achievements

- By combining tree search with policy and value networks, the introduced a frame work where human -level performance may be achieved in intractable AI domains. The method paves the way to advances in many other domains like
  - General game playing
  - Classical planning
  - Partially observed planning
  - Scheduling
  - Constraint Satisfaction

Remarks

- **I liked to see how this algorithm would compete with other commercial products if the time per move was extended to like 30s rather than 5. ☺**