Research Review:

Mastering the game of Go with Deep Neural Networks and Tree Search

Rizwaan Adil

# Summary

Go is a classic 2 player strategy board game where players try to surround each other while capturing territories on the board. Due to a high branching factor, enormous search space and difficulties in evaluating positions, Go was considered highly complex for AI agents. This research paper presents unique approach to overcome this complexity in Go gameplay. The approach outlines a search algorithm which combines Monte Carlo simulation with deep learning and reinforcement learning networks.

# Techniques

The key to AlphaGo’s success is the unique combination of value & policy networks and Monte Carlo simulation. Value networks are used to evaluate board positions and policy networks for move selection. Human expert games are used for training the value networks and reinforcement learning is employed for games of self-play.

The approach manifests in the form of a pipeline of machine learning stages as below

## Policy Network Training

A 13 layer policy network (SL) was trained on randomly sampled state-action pairs from 30 million position from KGS Go server (a popular online game host). This network produced the probability of each move being the actual next move on the board during gameplay.

While being only 55% accurate on held out test sets, the SL policy network was comparatively more accurate than the then state-of-the-art result of 44.4%.

## Reinforcement Learning for Policy Network

To further improve the SL policy network, another policy network (RL) was constructed with the same 13 layers and even the same initial weights. Overfitting was prevented by randomizing from a pool of opponents (one of the previous RL iterations) and weights were adjusted using stochastic gradient ascent.

In head to head games with the SL policy network, RL network posted 80% win rate. RL policy network also played against Pachi, the strongest open source Go player, which was backed by a sophisticated Monte Carlo search algorithm and boasted of 100k simulations per move.

## Position Evaluation Network

# Results