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

Review of AlphaGo research paper by DeepMind, completed as a part of
Udacity AI Nanodegree

Problem Statement

Until recently the game of Go remained as one of the only classical games not yet solved by artificial intelligence. Similar to chess the exhaustive search of actions space is not feasible due to both large breadth (in fact order of magnitude larger than chess) and depth of the game. Two primary techniques used before this study were the position evaluation to reduce breadth and Monte Carlo tree search (MCTS) for optimum policy to reduce depth. Both were sufficient to create Go agents able to play only at amateur human level. It was widely believed that it will take another decade or more to reach professional Go player levels.

Novel Techniques

Authors employ deep convolutional neural networks to analyze the Go board (19 x 19 image) and reduce both the depth, by using a policy network, and the breadth, by using value network to assess position, of the game.

The program consists of four models, connected with each other:

1. *Supervised Learning (SL) policy network* is a 13-layer neural network trained on 30 million of positions from KGS Go Server. The goal is to be able to predict the human expert move in the given position. After training this model managed to achieve the 57% accuracy using all input features (total of 48) or 55.7% using only raw board position, which is a dramatic improvement over 44.4% of prior existing models.
2. *Rollout policy* is a shallow version of SL policy network, consisting of only linear softmax layer. It achieved a much lower accuracy of 24.2%, however was three orders faster than SL policy network (2μs vs 3ms)
3. *Reinforcement Learning (RL) policy network* was employed in the 2nd stage of training pipeline. Its structure is identical to SL network. It was trained by playing games between randomly chosen different iterations of itself to further improve the SL network. The games were played to the terminal state and then result was back propagated to modify weight correspondingly. After training the RL policy network achieved 80% win rate against SL network and 85% win-rate against some of the available open-source Go

programs.

4. Final state of the training pipeline to create a *value network* to perform a position evaluation, i.e. provide a +1 or -1 value for the current state. Once again it used the similar structure as policy networks. To train it without overfitting (as it appeared to memorize all KGS games) authors generated a new self-play data set of 30 million positions. The MSE was around 0.22-0.23 for both training and test datasets. This model allows evaluating position with accuracy similar to Monte Carlo rollouts, but with 15,000 times less computation.

Combining Everything

AlphaGo uses four described networks in standard MCTS algorithm. During the *selection* phase the we select the edge of tree with the highest action value based on policy network. Once we reach the leaf node it may be expanded and is evaluated in two ways - by running the rollout to the end with the fast rollout policy and by using the value network. Both estimates are then mixed with some weight. The obtained result is then back propagated to all edges on the path. It is interesting that at least in the first games of AlphaGo with human experts the weaker SL policy was used instead of stronger one RL, as RL network was more focused on providing a single move, instead of giving a variety of promising moves and thus limiting exploration during MCTS.

Results

Single-machine AlphaGo has 99.8% success rate against available Go programs, and is at least few dans stronger. At the time of the publication the described model managed to win a 5 game match (5:0) with 2 professional dan (2p) player on a full board without handicap, which was never achieved before. It is known that shortly after the similar model with minor modifications managed to win 5 game match against 9p human player.

Overall it may be concluded that authors combined several previously known techniques (CNN, RL, MCTS) into an effective framework, which managed to provide a step change in performance.