Research Review

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

TREBUCHET CLEMENT
AIND-ISOLATION

Goals

AlphaGO agent must be able to beat opponents when playing the GO game.
The GO game is extremely challenging in reason of an high branching factor and a deep search tree (b≈250, d≈150). In order to solve this the deepmind team describes their implementation in terms of their understanding of the problem, and by combining the bests performing techniques.

Techniques involved and stages of the training pipeline

Summary

They train the neural networks using a pipeline consisting of several stages of machine learning They begin by training a supervised learning (SL) policy network pσ directly from expert human moves. This provides fast, efficient learning updates with immediate feedback and high-quality gradients. They train a fast policy pπ that can rapidly sample actions during rollouts. Next, they train a reinforcement learning (RL) policy network pρ that improves the SL policy network by optimizing the final outcome of games of selfplay. This adjusts the policy towards the correct goal of winning games, rather than maximizing predictive accuracy. Finally, they train a value network vθ that predicts the winner of games played by the RL policy network against itself. Their program AlphaGo efficiently combines the policy and value networks with MCTS.

Step 1: Supervised learning of policy networks

The SL policy network pσ(a| s) alternates between convolutional layers with weights σ, and rectifier nonlinearities. A final softmax layer outputs a probability distribution over all legal moves a. The inputs to the policy network is a simple representation of the board state.

The policy network is trained on randomly sampled state-action pairs (s, a), using stochastic gradient ascent to maximize the likelihood of the human move a selected in state s

The deepmind team trained a 13-layer policy network, from 30 million positions from the KGS Go Server.

The network predicted expert moves on a held out test set with an accuracy of 57.0% using all input features, and 55.7% using only raw board position and move history as inputs

Step 2: Reinforcement learning of policy networks

This step aims at improving the policy network by policy gradient reinforcement learning. The RL policy network pρ is identical in structure to the SL policy network.

They play games between the current policy network pρ and a randomly selected previous iteration of the policy network. Randomizing from a pool of opponents in this way stabilizes training by preventing overfitting to the current policy

They evaluated the performance of the RL policy network in game play, sampling each move from its output probability distribution over actions.

When played head-to-head, the RL policy network won more than 80% of games against the SL policy network

Step 3: Reinforcement learning of value networks

This step focuses on position evaluation, estimating a value function vp (s) that predicts the outcome from position s of games played by using policy p for both players2.

This neural network has a similar architecture to the policy network, but outputs a single prediction instead of a probability distribution.

They train the weights of the value network by regression on state-outcome pairs, using stochastic gradient descent to minimize the mean squared error (MSE) between the predicted value, and the corresponding outcome z.

Step 4: Searching with policy and value networks

The Alphago agent combines the policy and value networks in an MCTS algorithm that selects actions by lookahead search.

Evaluating policy and value networks requires several orders of magnitude more computation than traditional search heuristics. To efficiently combine MCTS with deep neural networks, AlphaGo uses an asynchronous multi-threaded search that executes simulations on CPUs, and computes policy and value networks in parallel on GPUs.

The SL policy network performed better in AlphaGo than the stronger RL policy network, presumably because humans select a diverse beam of promising moves, whereas RL optimizes for the single best move. However, the value function derived from the stronger RL policy network performed better in AlphaGo than a value function derived from the SL policy network.

Results

Evaluating the playing strength of AlphaGo

To evaluate AlphaGo, they ran an internal tournament among variants of AlphaGo and several other Go programs, including the strongest commercial programs Crazy Stone and Zen, and the strongest open source programs Pachi1 and Fuego.

The results of the tournament suggest that singlemachine AlphaGo is many dan ranks stronger than any previous Go program, winning 494 out of 495 games (99.8%) against other Go programs. To provide a greater challenge to AlphaGo, we also played games with four handicap stones (that is, free moves for the opponent); AlphaGo won 77%, 86%, and 99% of handicap games against Crazy Stone, Zen and Pachi, respectively. The distributed version of AlphaGo was significantly stronger, winning 77% of games against single-machine AlphaGo and 100% of its games against other programs.