**Trash**

and one value neare applied. First, a supervised learning

Solving the board game Go represents for computer

In the so-called “AlphaGo algorithm” presented, a Monte Carlo tree search (MCTS) is used in combination to a traditional minmax search, breaking the usual roll-out before reaching terminal state, and returning a simulation of the value function, instead of a terminal score. The value function relates to some policy, namely a probability distribution of possible moves at some given position.

Rollout policy: similar as policy network

Policy network classification: used to classify positions according to expert moves, takes board state as input and gives select human action as output. Input features were pre-computed at each position. Training took around 3 weeks for 340 million training steps.

Reinforcement learning on policy network trained for one day on 50 GPUs (10’000 mini-batches of 128 games)

Value network: trained during one week on 50 GPUs, on 50 million mini-batches of 32 positions each

Such policy is evaluated by making use of deep neural network so as to achieve a more powerful representation than linear representation at the cost of a slower computation that is ran asynchronously to the game playing time. A fast rollout policy is used in absence of search tree at given state.

The leaf position value is calculated by the value network

The best action at some given state is obtained as the one maximizing the value function and some additional exploration bonus. The algorithm however retains the action that maximizes the visit counts and not the one that maximizes value.

using a Monte-Carlo Tree Search (MCTS)