REVIEW OF THE ARTICLE MASTERING THE GAME OF GO WITH DEEP NEURAL NETWORKS AND TREE SEARCH

The article is at https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf and written by

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel & Demis Hassabis

This paper introduces a new way of approaching games of perfect information. Board games like Go and Chess are games of perfect information where all players can see the position in the game at all times. Every position has a value function *v(s)* where *s* is the *state or board position.* In theory, then these games may be solved by recursively computing the optimal value function in a search tree. In practice however, the number of positions to evaluate increase exponentially and quickly become impossible to be evaluated in any finite time.

Two methods can be used to reduce the effective search space. Firstly, the depth of the search can be reduced by positional evaluation. That is, stopping at a node in the tree and replacing its value with an approximate value of the subtree below it. Secondly, the breadth of the search can be reduced by considering only probable moves in a given position. This is determined by a *policy* that helps select the candidate moves. Monte Carlo tree search (MCTS) that uses Monte Carlo rollouts has been used in Go programs, but the *policy* to select moves has been shallow.

The team at DeepMind (https://deepmind.com/) came up with an innovative way to solve the search depth and breadth issue through the use of *convolutional neural networks*. The 19 x 19 Go board position is passed to the neural networks. They use a:

- *value network* to evaluate positions
- *policy network* for sampling actions (selecting moves)

The program they created – AlphaGo – combines the policy and value networks with MCTS.

Training the networks

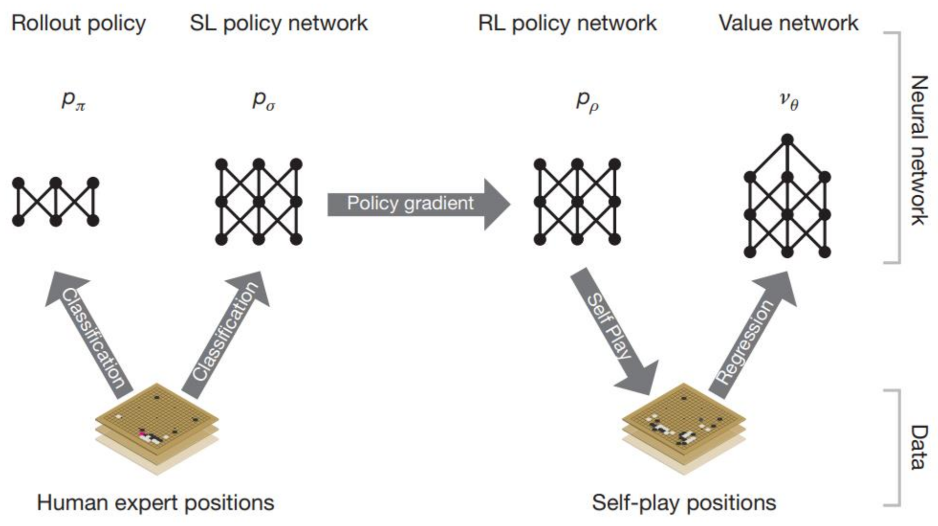
The neural networks are trained in the following stages:

- First a Supervised Learning (SL) *policy* network (p_σ) is trained directly using moves by human experts. This would result in the network that would produce moves that are most likely to be played by human experts in a given position.
- Next, a *policy* (p_π) is trained to rapidly sample actions during rollouts. That is, it selects moves by searching to maximum depth without branching, using the *policy* to select candidate moves

  The SL *policy* network (p_σ) and *policy* (p_π) predict human moves in a given position

- Next a Reinforcement Learning (RL) network *(P_ρ)* is initialized using the SL *policy* network. Both networks are identical in structure. The weights ρ is set to σ. This RL network *(P_ρ)* is improved by playing the current version of *(P_ρ)* with a random prior version of *(Pρ)*. A reward function is used to update the weights so that the evaluation moves in a in a direction that maximizes expected outcome (wins).
- Finally, a *value network* (v_θ) is trained by *regression* to predict whether the current plater wins.

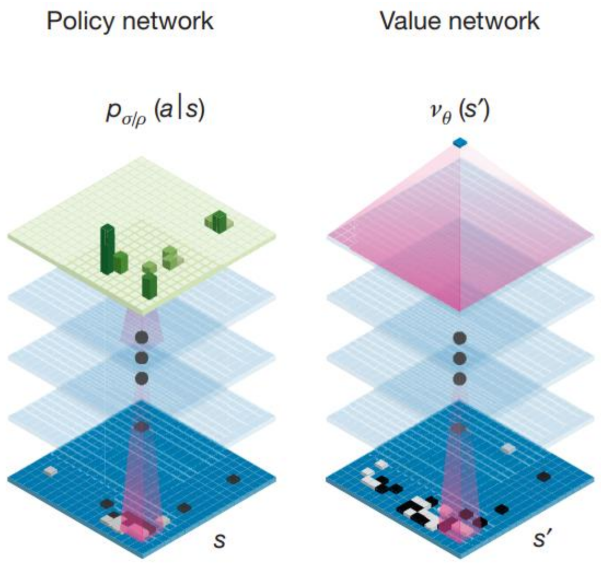
This training method is shown in the following diagram:



(from the article https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf)

Searching with the Policy and Value networks

AlphaGo combines the policy and value networks in an MCTS algorithm that selects actions by lookahead search. The board position is passed to the *policy network (p_ρ) or* (p_σ) that outputs a probability distribution map of legal moves. The *value network* (v_θ) is used to predict the expected outcome (player win/loss) for the moves. The tree is traversed by simulation – that is, complete games without backup. At each position (state) a move (action) is selected so as to maximize the value (probability of winning).



(from the article https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf)

Evaluating the playing strength of AlphaGo

The team found that AlphaGo on a single machine was many *dan* stronger than other Go programs winning 494 out of 495 programs. *Dan* are considered master ranks.

See https://en.wikipedia.org/wiki/Go_ranks_and_ratings

Over 5–9 October 2015 AlphaGo played Fan Hui, a professional 2 *dan*, and the winner of the 2013, 2014 and 2015 European Go championships in a formal five game match. AlphaGo won all five games.

Over 9-15 March 2016 AlphaGo played Lee Sedol an 18-time world champion and 9-dan professional in a formal five game match in Seoul, South Korea. AlphaGo won 4-1. For this the Korea Baduk Association awarded AlphaGo the highest Go grandmaster rank – an "honorary 9-dan"

Interesting References

RL Course by David Silver (DeepMind)
https://www.youtube.com/playlist?list=PLzuuYNsE1EZAXYR4FJ75jcJseBmo4KQ9-

AlphaGo Zero
https://www.nature.com/articles/nature24270.epdf?author_access_token=VJXbVjaSHxFoctQQ4p2k4tRgN0jAjWel9jnR3ZoTv0PVW4gB86EEpGqTRDtpIz-2rmo8-KG06gqVobU5NSCFeHILHcVFUeMsbvwS-lxjqQGg98faovwjxeTUgZAUMnRQ

Alpha Zero https://arxiv.org/pdf/1712.01815v1.pdf

DeepMind https://deepmind.com/