A summary of "Mastering the game of Go with deep neural networks and tree search" by Diego Gomez

This paper introduces a new approach, made by Google Deepmind, to create an artificial intelligence capable of playing the game of Go in an expert level. It works by implementing deep convolutional neural networks that work together in a machine learning pipeline, and combining it with Monte Carlo Tree Search (MCTS).

The first main neural network was labeled the SL (Supervised Learning) policy network and was trained to predict the probabilities of taking all legal moves *a*, given a state *s*. This network predicted expert moves with a max accuracy of 57%. A faster and less accurate neural network named Rollout Policy was also trained, which selected an action using just 2 μs, compared to 3ms with the Policy Network. It achieved an accuracy of 24.2%.

A second stage of the pipeline trains a Reinforcement Learning (RL) Policy Network, by policy gradient reinforcement learning, to improve the previously created SL Policy Network. It's structure is identical to the SL policy network and its weights are initialized to the same values. The current policy network is played against a randomly selected previous iteration of itself, in order to train the network in such a way that prevents overfitting to the current policy. The final network was evaluated by playing against the SL policy network and the strongest open-source Go program, Pachi, winning 80% and 85% of games, respectively.

As the last part of the training pipeline, a neural network was implemented, which focuses on position evaluation. This network estimates estimates the outcome of games played by using RL Policy network, from position *s*. Predicting game outcomes from complete games leads to overfitting, so it had to be trained with a randomly generated dataset. Given approach led to a lower error than if using complete games during training.

Using previously created policy and value networks AlphaGo uses a MCTS algorithm to search and choose actions. This algorithm simulates games starting from the root state, and at each time step an action (edge) is selected from a given state based on an action value, bonus, visit count and prior probability. These values are updated with the policies and value networks after reaching leaf nodes, in order to maximize the probability of selecting the best decisions. After simulation is over, the algorithm chooses the most visited move from the root position.

The final version of AlphaGo used 40 search threads, 48 CPUs and 8 GPUs. It was tested against various Go programs, including the strongest commercial (Crazy Store and Zen) and open source programs, by running an internal tournament. Results show that AlphaGo is better than all other programs, winning 494 out of 495 games (99.8% win rate). It also won 77%, 86% and 99% of handicap games against Crazy Stone, Zen and Pachi, respectively. A distributed version of AlphaGo was also implemented, with 40 search threads, 1202 CPUs and 176 GPUs. It was evaluated in a five-game match against Fan Hui, the winner of the 2013, 2014 and 2015 Go championships, over 5-9 October 2015. AlphaGo won all five games, becoming the first machine to win against a professional Go player, in the full game of Go, without handicap; a breakthrough in artificial intelligence.

Bibliography:

Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. and Hassabis, D. (2016) ‘Mastering the game of go with deep neural networks and tree search’, Nature, 529(7587), pp. 484–489. doi: 10.1038/nature16961.