AlphaGo

(review by Adwaith Gupta)

Go games can be solved by recursively traversing the search tree of possible sequential moves (board states) till the end game is not reached. However for a Go game, number of legal moves possible per position (breadth) are ~35 and the length of the game (depth) is typically ~80, making a total of 35^{80} board states in the search tree. Traversing such a huge tree is infeasible with the current state of computational power. In order to reduce the computational times, the DeepMind team used two types of deep neural networks (Ref. 1) in AlphaGo.

1. **Policy Networks -** These are used for reducing the breadth of the tree by using Convolutional Neural Networks or CNN (Ref. 2) in combination with Reinforcement Learning (RL). A policy network provides a probability distribution over all legal moves in a current board state.The higher probability for a move indicates higher likelihood that a human would play the same move. The CNN was trained using 30 million positions from the KGS Go Server. The CNN policy network was then further improved by using reinforcement learning, where the games are being played between the current policy network and a randomly selected previous iteration of the policy network. The outcome is set to +1 for a win and -1 for a loss. The table below shows the win rates of CNN policy network and CNN + RL policy network against Pachi (strongest open-source Go program).

Policy network	**CNN**	**CNN + RL**
Win Rate against Pachi	11%	85%

2. **Value Networks -** These reduce the depth of the tree. Value networks have the same architecture as the policy networks, expect for the output. Instead of the probability distribution, the output here is the predicted outcome of the game using the policy from the policy network. When trained on the KGS data set, the network tended to memorize the outcomes because of lack of variability in the input data set (many board states only vary by a single move). Therefore, a new self-play data set was created consisting for 30 million distinct positions, each sampled from a separate game. This resolved the problem of memorizing (overfitting).

AlphaGo combines the policy and value networks in the Monte Carlo algorithm that selects actions based on the lookahead search. Every node of the search tree stores an action value (based on the value networks), prior probability (based on policy networks) and node visit count. The action that maximizes the action value plus a bonus, is selected. Bonus is proportional to the prior probability, but decays if the node visit count is high, encouraging exploration to other nodes.

It was interesting that CNN policy networks performed better than CNN + RL policy networks during the search. A possible explanation is that humans choose diverse set of moves but CNN + RL aims for the single best move. However, CNN + RL value networks derived from CNN + RL policy networks outperformed CNN value networks. Using all these techniques, AlpgaGo was able to beat other Go programs at a win rate of 99.8% and defeated the human European Go champion by 5 games to 0.

References:

1. Mastering the game of Go with deep learning neural networks and tree search, DeepMind team
2. Un-convoluting the Convolutional Neural Networks (CNN), Adwaith Gupta