AlphaGo Paper:

Basic Concept:

● Search Tree would normally be constructed for choosing the moves; however, the tree is too big

● To reduce tree depth: Using a position evaluation to truncate the tree, and evaluating its value using an approximate value function

● To reduce tree breadth: In a game state s, use a probability distribution to sample some possible moves from given position

Supervised Learning of Policy Networks:

● First stage consists in training supervised learning policy network using human moves as data

● Weights are trained in a 13 layer neural network on random sampling of state-action pairs from a set of 30 million positions

● Stochastic gradient ascent is used to maximize the likelihood of a human move being selected in a given state

● Input: Representation of the board state

● Output: Probability distribution of all legal moves

Reinforcement Learning of Policy Networks:

● The RL policy network improves the Supervised Learning Policy Network through self-play

● The self-play games are played with a randomly selected previous iteration of the policy network in order to prevent overfitting

● Weights are initially set to sigma, and are trained using stochastic gradient ascent, in order for the policy to optimize for game-winning

Reinforcement Learning of Value Networks:

● Final stage of training involves predicting the outcome of the game from a particular game-state s of all games played by the RL policy network against itself

● The weights are trained by regression on state-outcome pairs, using stochastic gradient to minimize mean-squared error between the predicted value, and the corresponding outcome

● To avoid overfitting, a new data set was generated of uncorrelated self play positions

MCTS searching

● Each edge of the tree stores:

○ The number of times the node has been visited

○ The action value (long-term reward the agent can expect by taking said action from some state)

○ Probability of some action being taken while in some state s

● The steps of the tree search are:

○ The sum of the action value and some bonus value proportional to probability is maximized, and the node maximizing this is selected

○ Once a leaf node is reached in which the visit count exceeds some threshold, the leaf is expanded to a node that is processed by the SL policy network. The output probabilities are stored in the node

○ The leaf node is evaluated as a weighted combination of the RL value network and a random rollout, where a less precise policy plays out the simulation until the game reaches a terminal state. A parameter lambda is used to blend both sources of evaluation

○ A backward pass finally updates the visit count, and the action value of all traversed edges in the tree

Sources:

Youtube Video I found helpful(Relating to understanding the paper):

<https://www.youtube.com/watch?v=Z1BELqFQZVM&t=1s>

Medium Article which was also helpful in understanding the paper:

<https://medium.com/@antoine.louis/mastering-the-game-of-go-with-deep-neural-networks-and-tree-search-3869f85bf8a3>

Youtube video on Monte Carlo Tree Search I found helpful:

<https://www.youtube.com/watch?v=UXW2yZndl7U>