AlphaGo is a AI computer program, developed by Google DeepMind, that plays the board game "GO". "GO" is a popular board game played by 2 players taking turns with no random element involved. Go starts with an empty board and at each turn a player places a stone on the board. The goal is to capture as much territory as **possible.** The board game can be thus visualized as a game tree with each board position called as a game state. Nodes are the positions in the game and edges are moves.

Techniques Used:

It uses a combination of three deep convolutional neural network and Monte Carlo Tree Search algorithm to choose a move. The 3 neural networks are grouped into Policy Network (SL Policy network and RL Policy Network) and Value network. The input to all the neural networks is the board position as 19*19 image. The SL network is a 13-layer neural network which uses a final softmax layer to output a probability distribution over all legal moves. Stochastic gradient ascent was used to update the weights in the network during backpropagation. It was trained using supervised learning from 30 million positions from the KGS Go Server. The idea of RL policy network is to fine tune the SL policy network using Reinforcement Learning. Here random games were played between the current policy network and a randomly selected previous iteration of policy network. This random selection prevents overfitting. If the game ended in Win then the terminal state was given a score of +1 and -1 in case of loss. Reward was 0 for all other nodes. Weights are then updated by stochastic gradient ascent in a direction that maximizes the expected outcome.

Policy network is used to choose the best move among all possible moves from a board position. The value network, for predicting the outcome from any position of the game played, is used to evaluate a board position. It helped to decide whether a position is "good" or "bad". Based on that AlphaGo can decide whether to read more deeply along a particular branch. It has a similar architecture to the other two but uses a nonlinear activation function to output a single value. It is trained using regression on state-outcome pairs and using stochastic gradient descent to update the weights by minimizing the mean squared error between the predicted value and the outcome.

Search Process:

Searching is done using the Monte Carlo Tree Search algorithm where a search tree is built, node by node, according to the outcomes of simulated game using rollout network. Each edge in the search tree stores an action value Q(s,a), visit count N(s,a) and prior probability P(s,a). P(s,a) comes from SL Policy network. A leaf node gets a value estimate which is a combination of value network and by the outcome of a random rollout played out until the terminal step. At the end of simulation, the action values(Q) and the visit count (N) of all traversed edges are updated. Tree traversal policy selects actions that maximise Q value plus exploration bonus(proportional to P but inversely proportional to N). Once the search is complete the algorithm chooses the most visited move from the root position

Results:

AlphaGo achieved a winning rate of 99.8% against other Go programs. It was evaluated against Fan Hui, the winner of 2013,2014 and 2015 European Go championship. AlphaGo won the match 5-0. This is the first time a computer Go program has defeated a human professional player.