Over the past several years we have seen a tremendous growth in the field of Artificial Intelligence. It has progressed up to the point where computers can now defeat the best human players in a game of Go. Go is considered to be simplistic in its rules, yet can be complex in the strategies it employs. And because of its potential to be complex, it has been considered as the "most challenging of classic games for artificial intelligence" *(Silver et al, 2016)*. David Silver, the lead researcher at DeepMind and his colleagues go into greater detail, describing how their program AlphaGo was built and designed, beating the European Go champion with a perfect record.

The use of Monte Carlo rollouts have been the choice of algorithm in designing advanced Go AIs. Using rollouts allowed the program to sample actions and determine the best course of action based on the values it returned after a simulation. However the advancement in it's intelligence was limited by the shallowness of the value functions that were used *(Silver et al, 2016)*. And with the advancement of deep learning, DeepMind's AlphaGo was architected deep convolutional neural networks. Silver and his colleagues utilized a supervised learning policy network, reinforcement learning policy network, and a value network with Monte Carlo tree search to train.

Within its first layer, AlphaGo is trained to predict moves from expert Go players by sampling "from 30 millions positions from the KGS Go Server" *(Silver et al, 2016)*. And was able to achieve a greater than 50% accuracy when it was asked to predict the next move from the sample. This part of the network was essentially helping the program differentiate between what was right and what was wrong by giving it the answer after it made it's prediction. The trained output of the supervised learning layer is then connected to a reinforced learning network, which plays against itself utilizing the move predictor (of expert moves) from the previous layer. Of course winning a game is what reinforces the model. When the model from this layer was put against the supervised learning model from the previous layer, it won with a win rate of 80%, and 85% against Pachi, "the strongest open-source Go program" *(Silver et al, 2016).* Finally, on its last layer, AlphaGo was trained to determine the value of its position on the board (including its opponent) based on the predicted outcome from those positions.

All of these layers are then combined with a Monte Carlo tree search. It looks ahead by simulating the state if it were to take a move in that direction, returning the value for each of the simulated actions. The values are evaluated, and whichever seems favorable, it will take, and continuing this process throughout the game.