Go is an ancient board game where the objective to capture your opponent's "stones" by surrounding them. Although the game has simple rules, the number of possibilites are more than the number of atoms in the known universe. Games with this level of complexity makes the search space for AI agents intractable and thus clever heuristics must be used to reduce the search space and action selection. AlphaGo is an AI agent that has been built to play Go at an human professional level. It has proven its professional playing ability by defeating the human European champion by 5-0 and made headlines by defeating World Champion Lee Sodol 4-1. This was a milestone that was not expected to be accomplished within the next decade which makes its accomplishment a big deal for the progress of the Artificial Intelligence field. AlphaGo builds on top of previous research techniques and combines the use of deep neural networks and Monte Carlo Tree Search for its AI agent.

Deep neural networks are used to provide the policy(which action to take) and the value (how good is the move chosen) functions for their AI agent. The policy and value functions are represented by deep neural networks and are referred to as policy network and value network respectively. These networks receive board images as inputs which are passed into a series of convolutional, rectifier nonlinearities and softmax layers. The policy networks output a probability distribution whereas the value network generates a single outcome value. There are two policy networks that are used in AlphaGo. The fast rollout policy network is a less accurate, smaller neural network and its value is combined with either the outcome of a supervised learned or reinforcement learned policy network. Based on trial runs, the Supervised learned policy performed better due to the randomness performed by humans in their search.

These deep neural networks are combined with the Monte Carlo Tree Search algorithm. Each edge in the search tree stores the Action value, visit count, and prior probabilities. These values are updated as the tree is expanded and over iterations of the game. The search starts by selecting an action by the edge with the highest action value. Once the tree reaches a leaf node, that leaf node is expanded by executing the supervised learned policy network. At the end of the simulation, the algorithm evaluates the selected actions based on the outputs of the value network and the fast rollout policy. The action values, visit counts are propagated backup and the algorithm chooses the action with the highest visit count.

The AlphaGo algorithm used 40 search threads, 1202 CPUS, ad 176 GPUs in its distributed version. It was evaluated by executing games against the open source and commercial grade GO agents to date. The distributed version won 77% of the games against a single machine AlphaGo agent and 100% against all other agents. It eventually went on to beat the European champion Fan Hui 5-0 and the World Champion Lee Sodol 4-1 (https://www.wired.com/2016/05/google-alpha-go-ai/).