The adversarial game of Go is often used to test the performance of artificial intelligence due to the difficulty of building an evaluation function capable of assigning values to successive game states within a large state space. AlphaGo, a search algorithm introduced by the DeepMind Team, evaluates moves by utilizing neural networks – "value networks" and "policy networks," specifically – within a Monte Carlo Tree Search (MCTS) algorithm. In doing so, AlphaGo restricts its search space to a subset of game states. Comparatively, the majority of algorithms at the time of AlphaGo's introduction evaluated search spaces that would be considerd supersets of AlphaGo's search space. The approach of the DeepMind Team proved not only more effective than that of their peers, it also resulted in the first AI capable of beating a professional human opponent in a game of Go.

AlphaGo's policy network is a neural network built using supervised learning for the purpose of predicting human expert moves. The KGS Go server, a game server known for hosting both national and international Go tournaments, served as AlphaGo's source for informing its 13-layer policy network. The SL policy network, as the DeepMind Team calls it, is compromised of 12 layers alternating between "convolutional layers with weights, and rectifier nonlinearities" and a final softmax layer which provides a probability distribution over all available legal moves. With these, the SL policy network, given either all input features or solely raw board positions, achieved 55.7% and 57.0% prediction success rates, respectively.

Given the SL policy network as a foundation, reinforcement learning improves AlphaGo's play by addressing potential instances of overfitting. The reinforcement learning neural network (RL policy network) accomplishes this by replicating the SL policy network in structure and initializing all weights to that of the SL policy network. Following this, AlphaGo pits its current policies against a randomly selected previous iteration of its policies' weights. The SL policy network's weights are then adjusted based upon winning/losing terminal game states (i.e. draws have no bearing on this process), and with reference to the weights of the opposition's policy network. AlphaGo's value networks similarly use reinforcement learning, but return a single prediction in lieu of a probability distribution.

A new technique introduced by the DeepMind Team involved the rollout policy of their MCTS implementation. As opposed to the more application-specific rollout policies of previous Go-playing artificial intelligences, AlphaGo hashes the most likely action along with a pattern context. If the rollout matches the pattern context within the hast table, a high probability is attached to the move.

The key results of AlphaGo were revealed by testing. AlphaGo played games against a number of other Go programs, in addition to professional Go player Fan Hui, in a tournament. Probabilities were evaluated using the BayesElo program with the rating of Fan Hui as its foundation. Additionally, Chinese rules of *k*omi were used to score matches. The result was AlphaGo winning 99.8% of matches against other Go programs and, furthermore, winning 77%, 86%, and 99% of matches against three different Go programs with a four-stone handicap. Most significant of all, however, is that of 5 games played against Fan Hui, AlphaGo won all of the matches.