Article's summary - Mastering the game of Go with deep neural networks and tree search (doi:10.1038/nature16961).

The game of Go, one of the classic games, is considered the most challenging in the field of artificial intelligence. Its enormous search space containing approximately b^d sequences of moves with $b \approx 250$ and $d \approx 150$ makes ineffective the solution with minimax and alpha beta pruning search.
In prior works, techniques as reinforcement learning and Monte Carlo tree search (MCTS) have been used. Alpha Go make use of MCTS algorithm coupled with policy network and value network.
Trough the Monte Carlo rollouts is possible to reduce the branching or even using a beam (i.e. avoiding branching) to search to maximum depth by sampling with simulation long sequence of actions according with a policy $p(a \mid s)$. Such policy is a probability distribution over possible move a in position s.
The deep neural networks involved are trained by supervised learning from human expert games and reinforcement learning from games of self-play.
The input of the networks are positions as 19 x 19 images and through the use of convolutional layers is made a representation of those positions.

First a fast rollout policy p_π and a supervised learning (SL) policy p_σ are trained to predict human expert moves in a data set of positions. Then a reinforcement learning (RL) of policy p_ρ, initialized to the SL policy network, is improved from games of self-play with the goal of winning games. Then a new data set is generated and finally a value network v_θ is trained by regression from this data set.

The search is made on a tree where each edge (s, a) stores an action value $Q(s, a)$, visit count $N(s, a)$ and prior probability $P(s, a)$. The tree is traversed by simulation starting from the root state. At each time step a move is selected to maximize the sum of two terms: the action value $Q(s, a)$ and the prior probability $P(s, a)$ divided by $1 + N(s, a)$. The second addend decay with repeated visit to encourage the exploration. When a leaf node is reached during the traversal, that node may be expanded. The leaf position is processed by the SL policy network p_σ. The output probabilities are stored as prior probabilities for each legal action. From this point the evaluation is made as a linear combination of the value network in the leaf node state with coefficient $(1 - \lambda)$ and and a value obtained from fast rollout policy with coefficient λ. This latter value is obtained by a less accurate policy that make use of input with few features and is very fast to obtain.
The SL policy network has an accuracy within 55.7 and 57% depending on the number of input features used spending around 3 ms while the fast rollout policy has an accuracy of only 24.2% but uses just 2 µs. At the end of simulation all the action values and visit counts are updated.

The results demonstrate that using different values of λ, namely balancing the use of the value network and MCTS, is possible to obtain time performance and chances of winning that overcome both other programs and professional human experts.
Go is exemplary because faces several issues of the artificial intelligence: challenging decision making task, an intractable search space and a very complex optimal solution. This result is therefore an important milestone that allows to look with greater confidence in trying to solve intractable artificial intelligence problems.