Brief summary of the paper's goals

AlphaGo paper's goal is to find an optimal solution for a complex problem that cannot be approximated using a policy or value function in and enormous search space.
The game Go involves a deep search which has a width of 250 and depth of 150, for these types of game optimal solution cannot be provided just by recursively computing the optimal value function in a search tree containing approximately b^d possible sequences of moves. Thus, objective is to reduce the effective search space by reducing the depth and breadth of search.

Position evaluation may reduce the depth search by truncating the search tree at a specific state and replacing the subtree below the state by approximate value function that predicts the outcome of the state. However, this approach was not viable for AlphaGo. Breadth of the search may be reduced by sampling actions from a policy that is a probability distribution over possible moves in a state.

Techniques Introduced

The paper introduces a new approach 'value networks' to evaluate board positions and 'policy networks' to select moves. These deep neural networks are trained by a combination of supervised learning from human expert games, and reinforcement learning from games of self-play.

AlphaGo uses Monte Carlo tree search (MCTS) and Deep Convolutional Neural Networks (D-CNN).
MCTS uses Monte Carlo rollouts to estimate the value of each state in a search tree. An effective position evaluation is achieved by searching to maximum depth by sampling long sequences of actions for both players from a policy and without branching. As the search tree grows larger with increase in number of simulations, higher accuracy value is achieved. By selecting children with higher values, the policy used to select actions during search is also improved over time. This policy converges to optimal play, and the evaluations converge to the optimal value function. D-CNN uses many layer of neurons, each arranged in overlapping tiles, to construct increasingly abstract, localized representations of an image.
For AlphaGo following are sequence:

1. A board position image of size 19 by 19 is passed to construct representation of position, this is achieved by using convolutional layers.
2. Position (value network) and actions (policy network) are evaluated, these are used by neural network to reduce effective depth and breadth.
3. Neural network is trained using a pipeline consisting of several stages of machine learning.
4. Next training supervised learning policy network directly from expert human moves. This provides fast, efficient learning updates with immediate feedback and high-quality gradients.
5. Train a fast policy that can rapidly sample actions during rollouts.
6. Next, train reinforcement learning policy network that improves the supervised policy network by optimizing the final outcome of games of self-play. This adjusts the policy towards the correct goal of winning games, rather than maximizing predictive accuracy.
7. Finally, train a value network that predicts the winner of games played by the reinforcement learning policy network against itself.
8. MCTS uses the above networks to evaluate the value of each game position in search tree for best move

Summary of the paper's results

AlphaGo team has introduced a new search algorithm that successfully combines neural network evaluations with Monte Carlo rollouts. It integrates these components together, at scale, in a high-performance tree search engine. Using this search algorithm AlphaGo achieved a 99.8% winning rate against other Go programs, and defeated the human European Go champion by 5 games to 0.

Reference: https://storage.googleapis.com/deepmind-media/alphago/AlphaGoNaturePaper.pdf