**Review of the AlphaGo Paper:**
**Mastering the game of Go with deep neural networks and tree search**

This paper was published in Nature in 2016 and it describes the interesting work done by DeepMind team in developing an artificial intelligence (AI) agent playing the game Go that achieved significantly improved performance over conventional methods and existing AI player programs. Most significantly, "this is the first time that a computer Go program has defeated a human professional player, without handicap, in the full game of Go—a feat that was previously believed to be at least a decade away"

Go is a game with extremely large search space for moves and difficult board position evaluation. The state-of-the-art method is Monte Carlo tree search (MCTS), which basically run a set of Monte Carlo simulations (rollouts) to the end of the game to get the average outcome of certain leaf node in the tree search. In order to help narrow down the search space, policies or value networks are used to select high-probability actions and to prioritize actions during rollout.

AlphaGo improves overall existing computer programs in several aspects. Firstly, it uses convolutional layer to condense the board into a 19 x 19 image to reduce the breadth and depth of the search tree, an important factor to consider in time-limited game-play.

Secondly, it combines deep neural networks with tree search in an effective way. The best-performing AlphaGo variant employs several trained deep neural networks. For action sampling in the Monte Carlo rollout it uses a policy network $p_\sigma$ that is trained by supervised learning (SL) with training data from human expert games. A reinforcement learning (RL) based policy work did perform as well compared to the SL based network possibly due to the fact human select a more diverse set of promising moves. In exploring each leaf node, it uses a combination of a fast roll-out policy network $p_\pi$ trained with small patter features by SL and a value network $v_\theta$ trained through reinforcement learning. Based on internal tournament against other AlphaGo variants, the best performing heuristic score turned out to use a combination of $p_\pi$ and $v_\theta$ with equal weight, suggesting the two position-evaluation mechanisms are complementary

Finally, AlphaGo is realized based on highlight parallel computing scheme using CPU for executing roll-out simulations and GPU for evaluation policy and value networks. The version defeated human professional player used 1,202 CPUs and 176 GPUs.

Compared to Deep Blue for chess playing, AlphaGo choose positions more intelligently and resulted in evaluating significantly fewer positions. This is partly due to the use of value network, which resembles human player's. assessing current board state, and partly due to deep neural network trained from gameplay data rather than handcrafted rules in Deep Blue.

By achieving professional player strength in Go – a previously-thought impossible feat, AlphaGo sheds light on other seemingly intractable problems that could be solved by artificial intelligence.