As the paper introduces, all games with perfect information can be studied under an optimal value function that determines the outcome of the game. This function must be evaluated in every possible state of the game, it could be done by using a search tree. The problem is, that this game tree grows exponentially b^b where, for the game of GO $b \approx 250 \; and \; d \approx 150$. Hence, this method seems to be impossible to apply. At this point, the authors are facing the typical problem that artificial intelligence have to face: challenging decision-making (especially in go where the difficulty to evaluate board positions is outstanding) an intractable search space and a complex optimal solution. Let´s see how the team face the problem.

Basically, they introduce a new approach by using deep neural networks. They relay in the power of convolutional layers to construct the representation of the position by passing a 19x19 image of the board. By using neural networks, they reduce the depth and the breadth of the search.

In order to do that, they present the "value network" which evaluates the board position and afterwards the "policy network" starts to play and select the right moves. All these combined with a new search algorithm that combines Monte Carlo simulation with value and policy networks

They start the process by trying with supervised learning (SL) the policy network (p_σ) from expert human moves. Afterwards they prepare, as was done before in previous works, a fast policy p_π that can sample actions during rollouts and also train a reinforcement learning policy network p_ρ which optimizes the final outcome.

The 13 layer supervised policy network has been trained by using 30 million positions, predicting expert moves with an accuracy of 57% improving by 11% the current algorithms. The reinforcement learning policy network(RL) which has the same structure as the supervised one, is supposed to increase the performance by making it play against a previous state of the policy network and using a reward function (+1 for winning and -1 for losing). The performance showed by the RL policy network was amazing, beating previous algorithms in the 85% of the cases.

The final stage of training is focused on the value networks to evaluate the position. This network, has a similar structure than the policy one but outputs a single prediction. They trained the network on the KGS dataset by using stochastic gradient descent to minimize the mean squared error (MSE) between the predicted value, and the corresponding outcome. However some lack of generalization was found, in order to avoid this, a new self-play data set was generated consisting on 30 million distinct positions from separated games. Thanks to this, the over fitting almost disappeared.

Those both networks, the policy and the value networks, are combined in an MCTS algorithm that selects the actions by looking into the future. Once the search tree is deployed and each leaf is pondered, the algorithm chooses the most visited move from the root position.

Al these outstanding number of calculations are done in the final version by 40 search threads using 48 CPUs (executing simulations) and 8 GPUs (computing policy and value networks in parallel). In a distributed version, these numbers are increased till 1202 CPUs and 176 GPUs.

The results, as was seeing during the previous lines are incredible. The non-distributed version, won 99.8% against other GO programs. Even with handicap AlphaGo won from a range between 77% and 99% depending on the program faced.

The distributed version, won 77% of the games against the non-distributed version, and 100% against the other algorithms.

What is even more remarkable is the winning against the European champion Fan Hui by 5-0 (formal games) and more recently against the world champion Lee Sedol 4-1.