The paper describes the approaches used for the development of the AlphaGo engine into the world’s strongest Go program, as well as the first program to dominate the best human players. It provides the following information:

* A problem definition, outlining the core aspects of Go playing algorithms and AlphaGo’s improvements to these aspect, namely, the use of neural networks in the evaluation of value and policy function
* An explanation of the neural network methods used for the value and policy function
* A discussion of the resulting games of AlphaGo against the existing class-leading software and professional human players

Problem Definition

In Go, the possible consequences of each move are fully known, and a search tree consisting of all possible moves can be constructed. In practice this is impossible, since such a tree for Go would have of order 10^360 elements. State-of-the-art Go programs attempt to reduce the size of the tree by sampling only a subset of the branches of each state according to a *policy* (i.e., reducing the breadth), and truncating the tree after a certain depth using a *value function* that estimates how favorable the current board is for either player. For policies, in particular, it is important to predict the most likely move of a skilled opponent, as all branches corresponding to ‘stupid’ moves can safely be ignored.

Traditionally, value functions and policies have been trivial functions such as linear combinations of certain positions on the board. However, the quality of the predicted value (or likely move of the opponent) has previously not been competitive with human players. AlphaGo uses a supervised learning approach with convolutional neural networks (CNNs) to improve both the value and policy function.

Deep Learning Improvements

Initially, two different policy functions were trained:

* A simple logistic score based on the existence of certain position pattern as input features (the so-called rollout function)
* A complex 13-layer convolutional neural network.

Using supervised training on the same initial data set of 30 million positions taken from the KGS Go server, the trained policies were able to predict the opponents move with an accuracy of 24% within 2µs for the rollout policy, and with 55% accuracy (within a thousand-fold extended period of 3ms) using the deep learning policy. [[1]](#footnote-1)The results achieved represent a significant improvement over state-of-the art research, which so far had achieved 44% prediction accuracy with similar approaches.

For an alternative approach, the same 13-layer neural network was trained in reinforcement learning mode with games played against opponents chosen from the pool of class leading Go programs. Competitions between SL-trained networks pitched against RN-trained networks resulted in a win rate of 80% for RL, which also clearly dominated in competitions against existing program.

;l

1. To illustrate the relative unpredictability of Go, one can compare this complexity to that of the MNIST digit recognition problem. While the input complexity (each digit features a 28\*28 pixel ‘board’ with 256 possible grey scale values for each pixel) and arguably even the number of output classes can be considered comparable, MNIST can be solved by a shallow fully-connected network trained on a comparably tiny set of 60,000 labeled digits to an accuracy of over 96%. [↑](#footnote-ref-1)