The article talks about previous examples of Go programs and the entirety of their complexity. After providing background information about the inner workings of previous Go programs and directly comparing the setbacks they had to the solutions in AlphaGo. It can be seen that the new Go program, AlpahGo, was performing at a much higher level of play than previous programs. The authors then go on to explain that the how AlphaGo was able to achieve this using deep neural networks and tree searches.

Connections to class are the tree searches and the algorithms used to evaluate them. Once such example that the authors brought up was similar to the A\* search where AlpahGo had the policies and A\* has the heuristic. Due to this, we can see a nice comparison to an example we have done in class. The authors also talk about depth and breadth searches and calculating the optimal values of a search tree. This has a direct comparison to the cannibals and missionaries assignment we had done.

One of the confusion I had throughout the article was how the training and reinforcement sets were teaching the program. Was this done on startup or is it something the program always has access to. Another point of confusion was with the deep neural networks and how they work with the problem.

This article has shown me that Go in indeed a very complex game. I now understand the increasing difficulty of games as more possible sequences of moves and length are introduced. Just a small increase in either will result an exponential increase in computational time.