Summary: A Go game playing agent which uses 2 neural networks, one for value position evaluation and one for move selection, is presented. The networks use a combination of supervised learning from human expert games and reinforcement learning from self-played games to train the network's coefficients. The

Problem: The search space for the Go game (b ~ 250, d ~150) makes it infeasible to use exhaustive search. Although it is possible to truncate the depth of search and approximate the value of the function for the rest of the tree for the Go game this approach is intractable given the game's complexity. Even if a Monte Carlo tree search (MCTS) approach is used (i.e. The branching is avoided by evaluating known sequences which accuracy improves as more sequences are evaluated) limiting the evaluation functions to linear combinations of input features prevents outstanding results for the Go game.

Proposed Solution: A training pipeline is presented, in the first stage a model for expert move prediction is trained using supervised learning from expert human moves. The model was built using a 13 layer convolutional neural network. A faster rollout policy was also trained using a smaller set of pattern features as expected the faster policy responded quicker but had a lower accuracy. The policy used was the likelihood of a human moved in a selected state, as the objective was to maximise this likelihood a stochastic gradient ascent was used during the network coefficient determination.

The second stage is comprised of a Reinforcement Learning network which initially resembles the Supervised Learning network found in the first stage. However, this second stage network updates its coefficients to maximise the number of winning games compared to older policy networks. In a nutshell the agent randomly self-plays games and corrects the coefficients to the ones responsible for the most number of won self-played games.

With the model trained branch selection is based on Monte Carlo simulation, to combine the neural network model and the MTCS selection in an efficient manner the approach uses asynchronous multi-threaded search.

Results: The neural network approach allowed 99.8% win rate for fair games and win rate ranging from 77% - 99% for handicapped games against other algorithms. Using the MTCS selection the approach was also scalable and time constraint compliant.