Learn to play Go

Proposal

Albert Liu *albertpl@stanford.edu*

1 Introduction

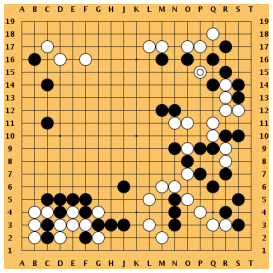

Figure 1: a snapshot of the Go board

We formulate Go as a turn-taking, two player, zero-sum game of perfect information. The state space is all possible placements of the stones on the board and the player who plays that turn. The player is either black stone or white stone. The state is fully observable as input. The action space is any legal position a stone can be, given the current state, and a pass action. The goal is to learn to win the game, by observing the board only. Figure 1 shows a concrete Go board and white stone is making a move at $P15$.

The main challenge is the enormous search space, large number of legal move per state, i.e. branching factor (~ 250), and the difficulty to handcraft a heuristic evaluation function for positions and moves. DeepMind team has made tremendous improvements, particularly AlphaGo [1], AlphaGoZero [2] and AlphaZero [3] with deep reinforcement learning approaches and self-play.

2 Evaluation

Pachi [4] is our game engine and we build the simulation environment based on OpenAI's Gym [5] implementation. We plan to have the agent play against the build-in Pachi engine for n games and report the average utility. We assign a scalar utility for our agent, 0 for tie, -1 for loss, and +1 for win.

As for our baseline, we propose two approaches

1. Random agent, i.e. uniformly choose one of the legal moves. We have random agent play 1000 games against Pachi engine.

   The win rate is 4/1000, average utility is -0.92, average time per game is 22.7 seconds and average time steps per game is 132. We note that the average step of all the non-loss games is 18, i.e. it is due to pass action.

2. We train a 4-layer convolutional neural network by supervised learning to predict the moves made by expert players. The dataset is from Fox Go Server [6] and contains $9K$ game records of professional players. Following AlphaGoZero, our input feature is a 19x19x17 array, representing 8 boards with each board in two channels, one for current player and the other for opponent. The last channel represents current player. We train on single GPU for ~ 36 hours and the train accuracy is 0.08 and the training is ongoing.

   The win rate is 0/100 as we write the report, i.e. agent lost all 100 games.

We see both play poorly. There are more works for the supervised training baseline. We have to explore more advanced Deep Reinforcement Learning approaches.

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[2] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, *et al.*, "Mastering the game of go without human knowledge," *Nature*, vol. 550, no. 7676, p. 354, 2017.

[3] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint arXiv:1712.01815*, 2017.

[4] P. Baudiš and J.-l. Gailly, "Pachi: State of the art open source go program," in *Advances in computer games*, pp. 24–38, Springer, 2011.

[5] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[6] FoxGoWebsite, "http://www.foxwq.com/."