Abstract

- Implementing a self-play based algorithm using neural nets has become popular after the huge success of Alpha Zero by Deep Mind.
- Replicating the results for games with larger search space like chess requires scaling.
- We develop a scaled up version of alpha-zero general for the game of Minichess and evaluate our learning algorithm with various baselines.

Keywords: CNN, Reinforcement Learning, Distributed Computing, Monte Carlo search

Introduction

- Self play and improve without any human knowledge.
- MCTS provides provides ground truth to compare and learn.
- 5x5 chess board with Gardner layout will be used for our training.

Neural Network Model

- Single Neural network used for both Policy and Value evaluation.
- We will use the following Loss function
  \[ l = -\gamma (a(s_, n_1) - z_1^2 + step(n_1)) \]

Distributed Architecture

- Three major components in self play and learn are:
  - Training Data Generator - Plays games and generates data for training.
  - Trainer - Consumes the data from Training Data Generator, compares against MCTS and learns.
  - Pitter - Compares two models and publishes a winner model.

Training Performance

- 2.5 times improvement in training speed with distributed setup.

Baseline Comparison

Conclusion

- Trained model beats the random, greedy baselines and performs decently on other layouts.
- Monte Carlo Tree Search and CNN can approximate search space as large as 9 \times 10^{10} as we seen in Minichess.
- Parallelizing self play, training and pitting by leveraging cloud services improves the performance substantially.

Observations

- Learning to Play Othello Without Human Knowledge Surag Nair et al
- Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm. Silver et al. 2017a

Contact Information

- Website: https://github.com/karthikselv/alpha-zero-general/tree/minichess
- Demo Video: https://youtu.be/NxuA5CC7YCE
- Mentor: patcho@stanford.edu