「技巧」アピール文書

2024 年 3 月 31 日

出村　洋介

1. Gumbel AlphaZero を用いた効率的な強化学習

AlphaZero [1]の学習を効率化した Gumbel AlphaZero [2]を用いて，既存の棋譜を使わずに強化学習でゼロから学習を行なっています.

オリジナルの AlphaZero では難しかったような軽量な実験条件（1 手 16〜64 シミュレーション程度）でも Gumbel AphaZero では比較的安定した学習ができるため，学習に要する計算資源が小さく済むようになり，各種実験を行いやすくなりました.

選手権用には，さらに学習時間を増やしたバージョンで参加予定です.

2. 強化学習時の初期局面を多様化

元々の AlphaZero では強化学習時の初期局面は 1 種類（平手初期局面）のみであり，手元の実験では自己対局で似たような展開になりやすい傾向が見られました.

そこで，今年の技巧では，自己対局時にさまざまな局面を積極的に経験させるため，学習時の自己対局で用いる初期局面の多様化を行なっています. 具体的には，2 手ランダムに指した局面（下図（b））や，駒の配置を一定の制約のもと並べ替えた初期局面（下図(c)）を初期局面として強化学習を行なっています（下図(c)の並べ替えでは駒の配置が左右対称を除き約 81 万通りあるため，「チェス 960」[3]にならって「将棋 81 万」と呼んでいます[4]).

このように強化学習時の初期局面を多様化することにより，実験では平手初期局面のみでの強化学習と比較して同一対局数で一定の勝率向上が確認できています. 学習順序としては，学習初期には「将棋 81 万」（下図(c)）で幅広い形を学習させてから，2 手ランダムの初期局面（下図(b)）で学習を行う方法が調べた中では効果的でした.

(a) 平手初期局面　(b) 2手ランダムの初期局面(例)　(c) 将棋81万の初期局面(例)

3. Rust と Python による独自実装

主要な開発言語には Rust を利用しており，既存の将棋ライブラリを使用しない独自実装

となっています．実装上は以下のような工夫がされています．

・Rust 組込みの 128 bit 整数型を用いた Bitboard

・利き数を保持するデータ構造

・利き情報を活用した高速な 1 手詰関数［5］

・ニューラルネットワークへの入力に将棋独自の特徴量を追加

また，学習部の開発は主に Python を用いて行っており，高速化や安定性向上のために次のような実装上の工夫がされています．

・PyTorch Lightning の混合精度学習による学習の高速化［6］

・Torch-TensorRT を用いた推論の高速化［7］

・Python 側と Rust 側のデータの受け渡し時に安全な NumPy 形式で転送［8］

参考文献

[1] D. Silver, et al. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. Science, 362(6419): 1140–1144, 2018.

[2] I. Danihelka, et al. Policy improvement by planning with Gumbel. In International Conference on Learning Representations, 2022.

[3] Wikipedia, https://ja.wikipedia.org/wiki/チェス 960.

[4] 出村，金子．将棋 81 万: 強化学習のための多様性を持った将棋初期局面集．第 28 回ゲームプログラミングワークショップ, pp. 111–118, 2023.

[5] 金子，田中，山口，川合．新規節点で固定深さの探索を行う df-pn の拡張．情報処理学会論文誌, 51(11): 2040–2047, 2010.

[6] PyTorch Lightning, https://lightning.ai/docs/pytorch/stable/common/precision_intermediate.html.

[7] Torch-TensorRT, https://github.com/pytorch/TensorRT.

[8] Rust-numpy, https://github.com/PyO3/rust-numpy.