6Q-06

深層強化学習による事前知識を用いないリバーシ盤面価値評価

山本　将也†　　藤田　悟†

法政大学　情報科学部‡

1. はじめに

二人零和有限確定完全情報ボードゲームは，手の分岐により指数関数的にパターンが増大するため，コンピュータの計算速度では全探索が不可能な場合が多い．人間によるプレイを模倣する方法も存在するが，それらの手が最善である確証はない．本研究では，定石やプロの棋譜などの情報は利用せず，機械同士の自己対局データの活用により強いプレイヤーを実現することを目的とする．

2. 先行研究

AlphaZero[1]は UCT をベースとした木探索により実装され，盤面評価には ResNet[2]を用いている．自己対局で得られたデータを重み更新に再利用し，性能向上を実現している． Neural Network で構築した Value および Policy Network の出力から木の探索を行うことで手を決定するが，この際に推論処理を多数行うため計算量を要する．

そこで，TD-Gammon[3]や Deep Q-Network[4]に代表される TD 誤差を用いた実装方法に着目し，木探索を使わない比較的単純なシステムによりリバーシゲームのプレイヤーを実装する方法を検討した．TD-Gammon では，終局における盤面価値に試合結果を用い，直前の盤面へ予測勝率を伝搬していくことで，盤面価値を更新する．盤面情報と盤面価値(予測勝率)の組を教師データとし，Neural Network で関数近似することで，広い探索空間の表現ができる．Deep Q-Network (以下，DQN) においても，Q 学習における Q テーブルの役割は Neural Network が担う．ビデオゲーム「Atari2600」において一定の成果を示した．リプレイメモリ(以下，RM と表記)にゲーム状態や報酬を保持し，Q 値を学習する際のデータセット構築に用いる方法を採用しており，本研究もこの実装方法を踏襲した．

3. 提案手法

3.1. 実装方針

リバーシにおいて，盤面価値関数(後述のモデル)を深層強化学習により更新する手法を提案する．盤面情報と盤面価値(予測勝率)の組を教師とし，モデルの更新に用いる．予測価値は-1 から 1 の範囲で表され，手番側が確実に勝利する場合を 1，相手が確実に勝利する場合を-1 とする．TD 誤差法の要領で自己対局結果からデータセットを構築することで，ゲーム前半の盤面へと価値を伝搬する．

3.2. モデルの詳細

機械学習モデルは ResNet を参考に構築した(図 1)．盤面情報 $\boldsymbol{s}$ (石の配置)を入力し，予測盤面価値 $V(\boldsymbol{s})$を-1 から 1 の値で出力する．盤面は 8×8 のサイズであるが，手番と相手側の層に分け，自石が存在する箇所を 1，それ以外を 0 とすることで，入力となる盤面情報を[8,8,2]の形状で表現する．残差層を 6 層重ね，出力直前で平坦化して Dense 層に通し，最終的に tanh を活性化関数に採用することで，値を-1 から 1 の範囲に制限している．

3.3. リプレイメモリ構築のための自己対局

データセット作成のため，モデル同士の自己対局(後述)を行い，得られた情報 $\boldsymbol{s}$，$\boldsymbol{s}'$，r_s，t_sを RM に追加する．ただし，$\boldsymbol{s}$ は現盤面の情報，$\boldsymbol{s}'$は手の選択後の盤面情報 $\boldsymbol{s}'$(相手目線)，r_sは-1，0，1 のいずれかで得られる自己対局結果(手番目線の勝敗情報)，t_sは経過したターン数とする．

3.4. V の重み更新に用いるデータセット構築

RM からデータの一部をランダムに選び，Vを更新するためのデータセットを構築する．選んだデータおよび 既存のモデルVの出力を用い，Algorithm1 に示した方法で sと$V(\boldsymbol{s})$の組を構築し，これを V の重み更新の際の教師データとする．$V(\boldsymbol{s})$の値域は-1 から 1 となり，終盤に近いほど盤面価値への報酬 r_sの反映率が高くなるよう設計した．

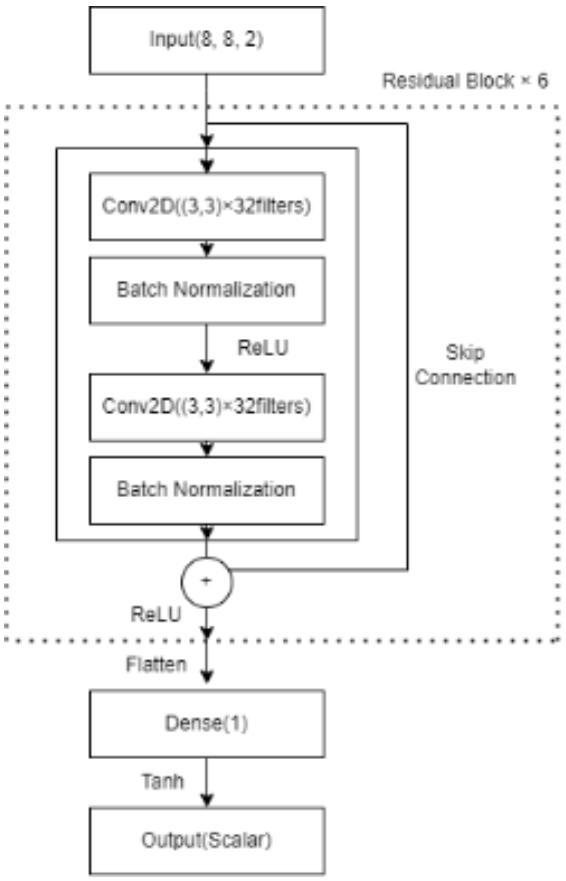

図 1　モデル構造

Reversi Board Value Prediction Using Deep Reinforcement Learning Without Prior Knowledge

† Masaya Yamamoto, Satoru Fujita

‡ Faculty of Computer and Information Sciences, Hosei University

Algorithm1 (Vの更新式)

$\alpha = 0.3$:TD 誤差の反映率 $(0 \leq \alpha \leq 1)$
$\gamma = 0.8$:報酬のターン数による割引率 $(0 \leq \gamma \leq 1)$
r_s :報酬 (1, 0, -1 のいずれかで表現される自己対局結果)
$\boldsymbol{s}$: 自己対局で現れた盤面(RM に保持)
$\boldsymbol{s}'$: $\boldsymbol{s}$から最善手を打った直後の盤面(相手目線)
t_s :自己対局においてその盤面が何ターン目であるか
(後攻 1 ターン目を$t_s = 0$とし，パスの場合も加算する．
t_sが 59 を超えた場合，t_s =59 とする．)
$t_{max} = 59$:ターン数の上限
for all $\boldsymbol{s}$ **do**
　if $\boldsymbol{s}$!= terminal **do** ($\boldsymbol{s}$ が終局状態でない場合)
　　$\beta = \gamma^{t_{max}-t_s}$ (報酬の反映率をターン数から決定)
　　$\mathrm{V}(\boldsymbol{s}) \leftarrow \beta r_s + \alpha(1-\beta)(-\mathrm{V}(\boldsymbol{s}')) + (1-\alpha)(1-\beta)\mathrm{V}(\boldsymbol{s})$
　else **do**
　　$\mathrm{V}(\boldsymbol{s}) \leftarrow r_s$ ($\boldsymbol{s}$ が終局状態なら試合結果自体を代入)
　end if
end for
(新たな $\boldsymbol{s}$, $\mathrm{V}(\boldsymbol{s})$の一部をデータセットとしVの重みを更新)

4. 実験

4.1. 自己対局からモデル更新を行う方法の概要

自己対局から得たデータセットにより重みを更新する操作を繰り返し，Vの性能向上を図る．初期の Vには乱数で重みを初期化したものを用いる．以後，自己対局からモデルの更新までの操作を「1 サイクル」と表記し，初回を cycle=0 として加算していく．

自己対局は，各合法手を打った直後の相手目線の盤面をVで推論し，相手目線の予測勝率が最も低くなる手を選ぶことで行う．ここで，盤面配列を上下，左右，表裏反転を任意に施した計 2^3=8 通りからランダム選択したものを推論することで，手の選択に多様性が生まれる．現れたすべての盤面 $\boldsymbol{s}$ に対し，t_s，r_s，$\boldsymbol{s}'$を記録しておく．これらの特徴量を用いて Algorithm1 の要領で$\boldsymbol{s}$と新たな盤面価値$V(\boldsymbol{s})$の組を得て，これを教師として重み更新を行う．数百から数千程度(250 戦を採用)の自己対局を並列に行い，ターン毎に推論すべき多数の盤面をバッチ処理することで高速化できる．

4.2. 自己対局や学習におけるパラメータ等

自己対局は 1 サイクル毎に 250 戦行い，1000000 盤面を上限に保存できる RM に盤面情報を追加していく．この際，左右，上下，表裏反転を施し，計 2^3=8 通りにデータを拡張する．自己対局結果を RM に追加後，保存上限を超過する場合には，ランダムに破棄する．その後，RM からランダム選択した盤面を用いて，Algorithm1 に従って教師データを作成する．データセットは 250000 盤面を上限に RM からランダムにデータを選択して作成したものを用いる．強化学習は性能向上が収束するまで続ける(650 サイクル(cycle=0~cycle=649)まで繰り返した)．Vの重み更新の Optimizer には Adam を採用し，学習率は 0.01，1 サイクル毎の学習 epoch 数は 3，バッチサイズは 1024 とする．

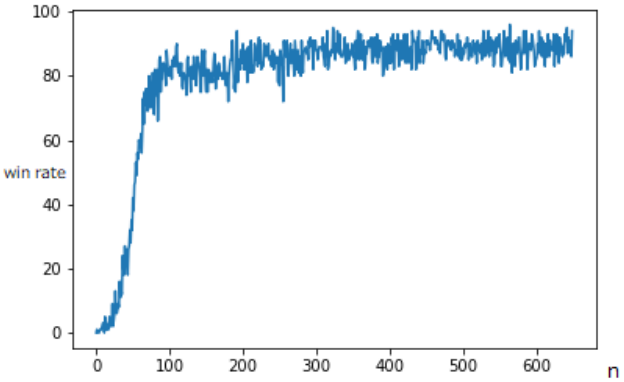

図 2 RL_nエージェントの対 RP 再現エージェント勝率推移
(横軸:サイクル数 n，縦軸: RL_n エージェントの勝利数)

4.3. 強化学習による性能向上の確認

以後，cycle=n での重み更新で得たVにより自己対局の要領で手を選択するエージェントをRL_nエージェントと表記する．サイクル毎に，RL_nエージェントと RP 再現エージェント(後述)の対局を 100 戦(50 戦で先攻と後攻を交代)行い，勝率の推移を記録していく．ただし，比較対象の RP 再現エージェントは強化学習を施していないモデルであり，RL_nエージェントと同じ構造のモデルに「盤面」と「ランダムプレイアウト(お互いにランダムな手を終局まで打つ試行を繰り返す操作)から得られる予測勝率」の組を代わりに学習させた．

5. 結果および考察

RL_nエージェントと RP 再現エージェントの対局結果のサイクル推移(100 戦におけるRL_nエージェントの勝利数)を図 2 に示す．サイクルを重ねるごとに RP 再現エージェントに対する勝率が向上しており，本手法のリバーシへの有用性が確認できる．ただし，比較対象の RP 再現エージェントに対して完勝する前に収束がみられ，本手法の限界も観測された．

6. 結論

TD 誤差法や DQN を基に設計した強化学習により，対戦性能の向上を実現できることを実証できた．自己対局を並列に多数同時に行うことで効率性を向上させた点，推論前に盤面を回転することでランダムな手を混入せずに多様性を確保した点など，実装時の工夫点の提案も行った．ただし，真の評価値に辿り着く前に強化学習が収束し，学習方策等に改善の余地が残る．

文　献

[1] David Siver, et al. “Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm,” 2017. https://doi.org/10.48550/arXiv.1712.01815

[2] K He, et al. “ Deep Residual Learning for Image Recognition,” 2015. https://doi.org/10.48550/arXiv.1512.03385

[3] Gerald Tesauro, “Temporal difference learning and TD-Gammon,” Communications of the ACM, Volume 38, Issue 3, pp 58–68, 1995. https://doi.org/10.1145/203330.203343

[4] Volodymyr Mnih, et al. “Playing Atari with Deep Reinforcement Learning,” NIPS Deep Learning Workshop 2013. https://doi.org/10.48550/arXiv.1312.5602