

畳込みニューラルネットワークを用いた デジタルカーリング AI作成の試み

松井 亮平^{1,a)} 保木 邦仁¹

概要：不確定性を持ち、ゲーム状態や行動が浮動小数点数で表されるデジタルカーリングを題材に、行動価値関数を畳込みニューラルネットワークで近似する方法を検討する。ランダムな方策で進行するエンドの得点確率を関数近似し、この近似関数の得点の期待値を最大化するグリーディ方策によってランダム方策を改善した。改善した方策は単純なルールベース AI には及ばないが、ランダム方策よりは優れたものとなった。簡易な実験条件ではあるが方策改善がなされたことが示された。

1. はじめに

デジタルカーリングは 2015 年に初の AI 大会^{*1}が開催されるなど、思考ゲーム AI の研究対象としては比較的新しいゲームである。これは二人零和有限不確定完全情報ゲームで、ゲーム状態や行動が浮動小数点数で表現されるのが特徴の一つである。

過去のデジタルカーリング AI 大会ではモンテカルロ木探索を行う歩 (あゆむ) [1] や、不確定性を考慮したミニマックス探索を行うじりつくん [2] などが上位入賞してきた。歩では行動価値関数を、じりつくんでは局面評価関数を用いる。現在のデジタルカーリングにおいて、強い AI を開発するためには何らかの評価関数が必要になると考えられる。評価関数の設計は、他のゲームでも重要であり、例えば囲碁 AI の Alpha Go [3] は、畳込みニューラルネットワークを用いてゲーム状態や行動の価値関数を近似する手法により大きな成果を挙げた。

本研究ではデジタルカーリングを題材に、ゲーム状態と行動を入力として行動の価値を出力する行動価値関数を畳込みニューラルネットワークで近似する方法を検討する。行動価値関数を強化学習することを念頭に置き、まずはランダムな方策をモンテカルロ法によって改善することを目的とする。

本研究の内容は現状では強いデジタルカーリング AI を開発することには直結していないが、デジタルカーリングの AI で十分実用可能な高精度価値関数開発への足がかり

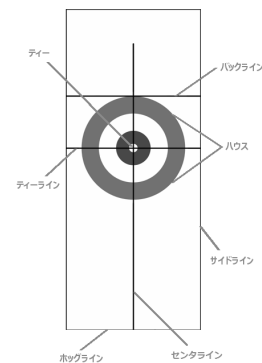


図 1 シートの拡大図とカーリングの用語

になるのではないかと期待される。

2. デジタルカーリングの概要

デジタルカーリングは、ウィンタースポーツの一種であるカーリングの戦略を議論するために北清ら [4] によって提案された二人零和有限不確定完全情報ゲームである。

本章ではまずカーリングのルールを説明し、続いてデジタルカーリング特有のルールを紹介する。

2.1 カーリングのルールと用語

カーリングは氷上で行うゲームであり、ストーンを交互に投げることで進行していく。カーリングを行う氷の面をシートと呼ぶ。図 1 にカーリングで用いるシート上の用語を示す。ホッグラインとバックライン間の距離は約 8.23m、サイドライン間の距離は約 4.75m となる。

以下にカーリングに関するいくつかの用語を記載する。

¹ 電気通信大学
The University of Electro-Communications

^{a)} m1311180@edu.cc.uec.ac.jp

^{*1} 第 1 回 UEC 杯

プレイエリア バックライン, サイドライン, ホッグラインに囲まれた領域. ホッグラインを超えないストーン, サイドラインに触れたストーン, バックラインを超えたストーンは除外される. ストーンの除外されない範囲をプレイエリア内と呼ぶ.

ティー ティーラインとセンタラインの交点.

ハウス ティーを中心とする半径 1.82m の円. あるストーン中心のティーからの距離を r , ハウスの半径を $R_h = 1.82\text{m}$, ストーンの半径を $R_s = 0.145\text{m}$ としたとき, $r < R_h + R_s$ を満たすならばそのストーンはハウス内に含まれる.

ショット ストーンを投げる (氷上を滑らせる) こと. ショットの起点はティーから約 36m ホッグライン側へ離れたセンタライン上となる.

テイクアウト プレイエリア内のストーンをプレイエリア外へ弾き出すこと.

カーリングの 1 エンドは基本的には以下のように進行し, 8 または 10 エンドの合計得点で勝敗を決める.

カーリングの 1 エンド

- (1) 先攻から交互にストーンをショットする.
- (2) お互いに 8 投した時点で得点計算を行う.
 - ハウス内のストーンのうち, 最もティーに近いストーンを持つ側が得点権を得る.
 - 得点権を得なかった側の最もティーに近いストーンよりもティーに近いハウス内のストーンの数 が得点となる.
 - ハウス内にストーンが存在しない場合にはどちらも得点しない.
- (3) ここまでを 1 エンドと呼び, 得点した側が先攻となって次のエンドを行う.

なお, 1 ゲーム通じての持ち時間が設定されており, 持ち時間を使い切った時点で負けとなる.

2.2 デジタルカーリングのルール

デジタルカーリングではゲームの状態を以下の 6 つの要素で表現している.

デジタルカーリングのゲーム状態

- 現在のショット数 (整数)
- 現在のエンド数 (整数)
- 最終エンド (整数)
- 各エンドの得点 (整数)
- どちらの手番か (真偽値)
- 各ストーンの座標 (単精度浮動小数点数)

ショットを以下の 3 つの要素で表現している.

デジタルカーリングのショット

- 初速度の x 成分 (単精度浮動小数点数)
- 初速度の y 成分 (単精度浮動小数点数)
- 回転方向 (真偽値)

デジタルカーリングのサーバは次のような順序で動作する.

デジタルカーリングサーバの動作

- (1) サーバにゲーム状態とショットの情報を与える.
- (2) ショットの初速度の x, y 成分に正規乱数による補正が加えられる.
- (3) ショットの物理シミュレーションを行う.
- (4) シミュレーション後のゲーム状態を返す.

2.2.1 実際のカーリングとの主な違い

実際のカーリングではシートの摩擦は一様ではなく, ショットの度に変化していくが, デジタルカーリングでは一様かつ一定である.

実際のカーリングでは投げられたストーンの進行方向をブラシで擦る (スウィーピング) ことでストーンの軌道の微調整を行うが, デジタルカーリングではスウィーピングは実装されていない. 北清ら [4] はスウィーピング及びショットミスの影響はショットの不確実性によって吸収できると主張している.

3. 先行研究

本章ではデジタルカーリングの評価関数に関する先行研究について述べる.

3.1 じりつくん

じりつくんは加藤ら [2] の提案したデジタルカーリング AI であり, 不確実性を考慮したミニマックス探索によりショットを決定する.

ミニマックス探索を行うゲーム木の末端ノードの局面を s としたとき, 局面評価関数 $e(s)$ を用いてノードの評価を行う. $e(s)$ は式

$$e(s) = W(s) + \mu \sum_i u(s, i) N(s, i) \quad (1)$$

で定義される. $W(s)$ は局面 s でエンドを打ち切った場合の勝率であり, 残りエンド数と得点差と自分の手番のルックアップテーブルを参照して勝率を取得する. μ は結合の重みを表す. $u(s, i)$ は局面 s における i 番目に投げられたストーンが自分のストーンであれば +1, 相手のストーンであれば -1 を出力する. $N(s, i)$ は i 番目に投げられたストーンの位置や他のストーンとの位置関係を評価する関数であり, 数百回程度の四則演算と数の大小比較, ストーン数と同数程度の三角関数の評価を行う, 計算時間とパラメタ数を抑えるよう設計された表式である. パラメタは自己

対戦と手調整により決めたと考えられるが、詳細は報告されていない。

3.2 歩

歩は大渡ら [1] によって提案されたデジタルカーリング AI であり、ゲーム状態やショットの連続性を考慮したモンテカルロ木探索を行う。また、シミュレーション中の方策に行動価値関数を利用している。

方策は線形重み和からなる行動価値関数の softmax であり、重みの数は 40000 程度、交差エントロピーを最小化する学習によって重みを決定している。訓練データには過去大会の上位 AI4 種による対戦で現れた局面局面を用いている。

4. モンテカルロ法による得点分布の予測

デジタルカーリングにおけるゲーム状態の集合を S とし、ある状態 $s \in S$ におけるショットの集合を $A(s)$ と表記する。また、状態 s における方策を $\pi(s, a)$ ($a \in A(s)$) と表記する。 $\pi(s, a)$ は状態 s においてショット a を選択する確率であり、 $\sum_{a \in A(s)} \pi(s, a) = 1$ を満たす。

1 エンドの得点を元に自分が 3 点以上から相手が 3 点以上の 7 クラスの中からクラス c ($1 \leq c \leq 7$) を求める。ゲーム状態 s とショット a に対して、以降方策 π に従った場合の真の得点の確率分布を

$$\mathbf{f}^\pi(s, a) = [f_1^\pi(s, a), \dots, f_7^\pi(s, a)]$$

とし、近似された得点の確率分布を重みベクトル $\mathbf{w}$ を用いて

$$\tilde{\mathbf{f}}^\pi(\mathbf{w}, s, a) = [\tilde{f}_1^\pi(\mathbf{w}, s, a), \dots, \tilde{f}_7^\pi(\mathbf{w}, s, a)]$$

と表記する。このとき、方策 π で進行するエンドで現れる状態とショットの対 (s, a) の集合を X^π として、交差エントロピーの標本平均を

$$E(\mathbf{w}, s, a) = -\frac{1}{|X^\pi|} \sum_{x \in X^\pi} \sum_{r=1}^7 f_r^\pi(x) \log \tilde{f}_r^\pi(\mathbf{w}, x) \quad (2)$$

と定められる。この $E(\mathbf{w}, s, a)$ が小さいほど $\tilde{\mathbf{f}}^\pi$ が $\mathbf{f}^\pi$ の良い近似となると考える。

$\mathbf{f}^\pi$ は未知であるが、方策 $\pi(s, a)$ を用いて N エンドブレイシ、モンテカルロ法によって評価すれば、交差エントロピーの標本平均は

$$\begin{aligned} E(\mathbf{w}, s, a) &= -\frac{1}{|X^\pi|} \sum_{x \in X^\pi} \sum_{r=1}^7 f_r^\pi(x) \log \tilde{f}_r^\pi(\mathbf{w}, x) \\ &\approx -\frac{1}{N} \sum_{n=1}^N \sum_{r=1}^7 d_{nr} \log \tilde{f}_r^\pi(\mathbf{w}, s_n, a_n) \quad (3) \end{aligned}$$

と近似できる。ただし、 n エンド目の結果クラス c に分類されたとき $d_{nc} = 1$, $d_{nr} = 0$ ($r \neq c$) とする。

式 3 の交差エントロピーを最小化するような $\mathbf{w}$ を求めるために、学習率を $\epsilon > 0$ として

$$\mathbf{w} \leftarrow \mathbf{w} - \epsilon \nabla_{\mathbf{w}} \left(-\frac{1}{N} \sum_{n=1}^N \sum_{r=1}^7 d_{nr} \log \tilde{f}_r^\pi(\mathbf{w}, s, a) \right) \quad (4)$$

なる更新を行う [5]。

クラス $c = 1, 2, \dots, 7$ が順に、相手が 3 点以上、相手が 2 点、 $\dots$ 、自分が 3 点以上と表すものとする。このとき、 $\tilde{\mathbf{f}}^\pi$ を用いて、得点の期待値を

$$Q^\pi(s, a) = \frac{1}{7} \sum_{r=1}^7 (r-4) \tilde{f}_r^\pi(\mathbf{w}, s, a)$$

と求められる。ただし、+3 点以上は全て +3 点、-3 点以下は全て -3 点として扱う。

ゲーム状態 s において

$$a^* = \operatorname{argmax}_{a \in A(s)} Q^\pi(s, a)$$

なるショット a^* を選ぶ戦略をグリーディ戦略 [6] と呼ぶ。グリーディ戦略に従う方策 π^{new} は $\pi^{\text{new}}(s, a^*) = 1$ かつ $\pi^{\text{new}}(s, a) = 0$ ($a \neq a^*$) となり、 π^{new} は元の方策 π よりは良い方策となることが期待される。

5. 関数近似の方法

本章では得点確率の関数近似に用いたニューラルネットワークについて説明する。

ニューラルネットワークは脳の神経細胞であるニューロンを模したユニットを複数個結合させたものである。1 つのユニットは複数の入力に対して 1 つの出力を行う。あるユニットに対する入力を $\mathbf{x} = \{x_1, x_2, \dots, x_n\}$ 、重みを $\mathbf{w} = \{w_1, w_2, \dots, w_n\}$ 、バイアスを b としたとき、そのユニットの出力 u は

$$u = \sum_{i=1}^n w_i x_i + b \quad (5)$$

となる。

5.1 順伝播型ネットワーク

複数のユニットからなる層を何層も重ね、隣接する層間でのみユニットが結合され、情報が入力側から出力側への一方向のみに伝播するニューラルネットワークを順伝播型ネットワークと呼ぶ。

一般に N 層のネットワークにおいて隣接する層間の全てのユニットが結合している場合、 l 層目の i 番目のユニットの出力を $u_i^{(l)}$ 、 l 層目の i 番目のユニットから $l+1$ 層目の j 番目のユニットへの重みを $w_{ji}^{(l+1)}$ 、 $l+1$ 層目の j 番目のユニットのバイアスを $b_j^{(l+1)}$ 、 $l+1$ 層目の j 番目ユニットの出力 $u_j^{(l+1)}$ は

$$u_j^{(l+1)} = \sum_i w_{ji}^{(l+1)} u_i^{(l)} + b_j^{(l+1)} \quad (6)$$

となる ($l = 1, 2, \dots, N-1$)。ただし、ニューラルネッ

トワークの入力を $\mathbf{x} = (x_1, \dots, x_i, \dots)$, 出力を $\mathbf{y} = (y_1, \dots, y_i, \dots)$ としたとき, $u_i^{(1)} = x_i$, $u_i^{(N)} = y_i$ とする.

5.2 畳込みニューラルネットワーク

本節では, 各ユニットが直前の層の一部のユニットとのみ結合し, 複数の結合で重みを共有する畳込み層について説明する. 重みが共有されていることで, 全結合層よりも重みの総数が削減される.

前項まではグレースケールの画像 1 枚を 1 種類のカーネルで畳込むことを考えたが, 一般に入力画像は多チャンネルからなる. 畳込み層では N チャンネルからなる入力画像を NM 種類のカーネルで畳込み, M チャンネルの画像を出力する.

$l+1$ 層目が NM 種類のサイズ $K_w \times K_h$ のカーネルを持つ畳込み層であるとする. l 層の出力が N チャンネルからなるサイズ $W \times H$ の画像であるとし, この画像の n チャンネル目の各画素の値を $u_{i,j,n}^{(l)}$ ($i = 0, 1, \dots, H-1$, ($j = 0, 1, \dots, W-1$)) とする. また, n チャンネル目に対応する m 番目のカーネルの各画素の値を $h_{p,q,n,m}^{(l+1)}$ ($p = 0, 1, \dots, K_h-1$, ($q = 0, 1, \dots, K_w-1$), ($n = 0, 1, \dots, N-1$), ($m = 0, 1, \dots, M-1$)) とおき, m 番目のカーネルに対応するバイアスを $b_m^{(l+1)}$ とし, スライド s ずつカーネルを動かすとする. 以上を用いて, 畳込み層の計算は

$$u_{i,j,m}^{(l+1)} = \sum_{n=0}^{N-1} \sum_{p=0}^{K_h-1} \sum_{q=0}^{K_w-1} h_{p,q,n,m}^{(l+1)} u_{si+p,sj+q,n}^{(l)} + b_m^{(l+1)}$$

と表される. このとき, この層の出力画像の m チャンネル目の各画素の値は $u_{i,j,m}^{(l+1)}$ となる.

5.2.1 プーリング層

プーリング層は主に畳込み層の直後に現れる層であり, 畳込み層によって抽出される特徴の位置感度を低下させる働きをする.

プーリング層への入力画像の n チャンネル目の各画素の値を $z_{i,j,n}^{(l)}$ ($i = 0, 1, \dots, H-1$, ($j = 0, 1, \dots, W-1$)) とする, 大きさ $K_w \times K_h$ のカーネルを考える. プーリングにはいくつかの方法があるが, 本研究では次の式

$$u_{i,j,m}^{(l+1)} = u_{si+p^*,sj+q^*,m}^{(l)} \quad (7)$$

で表される最大プーリングを利用する. ただし, p^* と q^* は $0 \leq p < K_h$, $0 \leq q < K_w$ で $u_{si+p,sj+q,m}^{(l)}$ に最大値を与える p, q である. カーネルをスライド s ずつ動かしながらこの式を繰り返し適用することで, プーリング前の画像とチャンネル数が等しく, 各画素の値が $u_{i,j,m}^{(l+1)}$ の画像を得られる.

6. グリーディ戦略を求める方法

ショットの初速度が浮動小数点数であるために, 全て

のショットを比較してグリーディ戦略に基づくショットを厳密に求めることは現実的ではない. 本研究では, グリーディ方針に従うショットを求める際に得点期待値 $Q^\pi(s, v_x, v_y, \text{curve})$ を最大化する初速度 (v_x, v_y) を探索するための手法として以下の 4 つの方法を検討する. ただし, $v_x, v_y \in [0, 1]$ であり, curve は右回転または左回転を表すものとする.

直線探索付き最急降下法 初期点を 1 点とする直線探索付きの最急降下法 [7]. 初期点は後述の格子点評価によって決定する. 直線探索には 2 分割法とアルミホの基準 [7] を用いる. 位置の更新距離が微小となったときに反復を打ち切る.

初期点複数の最急降下法 初期点を 4 点とする最急降下法. 初期点は後述の格子点評価の上位 4 点とする. 反復 1 回あたりの更新距離は固定で, 反復回数も 4 回で固定する.

格子点評価 8×8 の格子の格子点を評価する方法. $Q^\pi(s, v_x, v_y, \text{curve})$ に最大値を与える $v_{xi} = i/8 + 1/16$, $v_{yj} = j/8 + 1/16$ ($i, j = 0, 1, \dots, 7$) をショットとして選択する.

ランダムサンプリング 一様乱数によって選んだ 100 点を評価する方法.

6.1 2 階以上の偏導関数の利用

ニューラルネットワーク $\tilde{f}^\pi(s, v_x, v_y, \text{curve})$ は, 初速度 v_x, v_y の連続関数とみなせるので, 最急降下法のような勾配を用いる手法の利用が可能であると考えられる. しかし, ReLU 層や最大プーリング層で $\max$ 関数を用いている関係で v_x と v_y の 1 階偏導関数は不連続となるため, 2 階以上の偏導関数を用いる手法の利用は困難である.

図 2 に, 本研究で学習したニューラルネットワークを用いて得点確率を近似した関数 $\tilde{f}_4^\pi$ において, ゲーム状態 s , ショットの初速度の y 成分 v_y , ショットの回転方向 curve を固定し, ショットの初速度の x 成分 v_x のみを変動させた際の出力の様子を示す. また, 図 3 に, この入力に対する初速度の x 成分による 1 階偏導関数の様子を示す. これらの図から, $\tilde{f}_4^\pi$ が微分不可能点を持つものの連続である一方で, $\frac{\partial \tilde{f}_4^\pi}{\partial v_x}$ は不連続であり激しく上下している様子が確認できる.

以上の理由から本研究においては 2 階以上の偏導関数を利用しないものとする.

7. 1 エンドの得点予測

本研究ではニューラルネットワークの学習に深層学習フレームワークである Caffe*2 を用いて, 図 4 に示す構成のニューラルネットワークを学習する. また, 訓練データの

*2 <http://caffe.berkeleyvision.org>

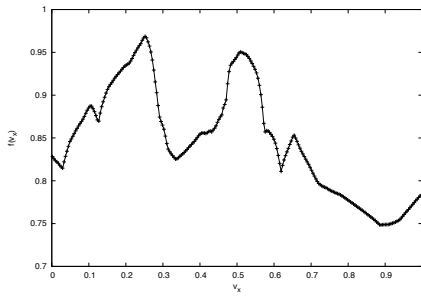


図 2 $\tilde{f}_4^\pi$ の出力例

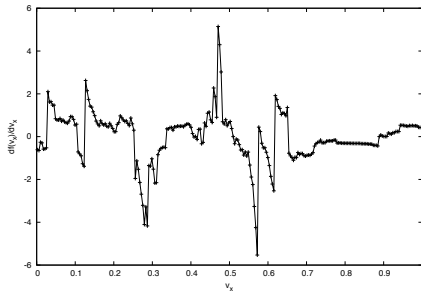


図 3 $\frac{\partial \tilde{f}_4^\pi}{\partial v_x}$ の出力例

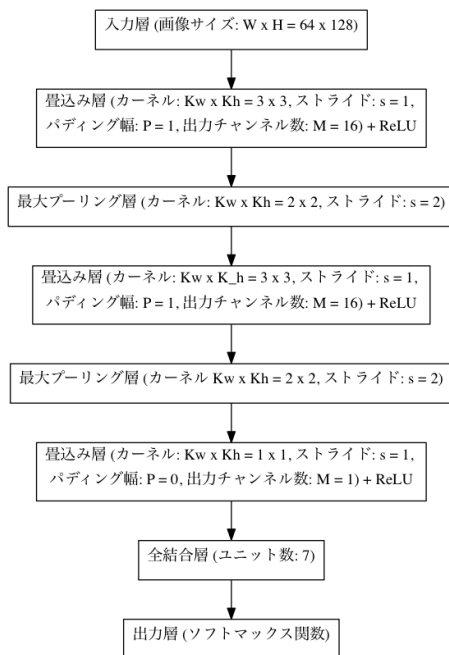


図 4 本研究で用いた畳込みニューラルネットワークの構成

作成や学習済みニューラルネットワークの性能確認実験の際には、動作の高速化のため公式のデジタルカーリングサーバを模した自作のシミュレータを用いる。

7.1 ランダム方策 AI

本節では、訓練データの作成や一部実験の際に用いたランダム方策 AI の挙動を説明する。

ランダム方策 AI は、初速度の x, y 成分をそれぞれ区間 $[X_1, X_2]$, $[Y_1, Y_2]$ の一様乱数を用いて決定し、回転方向は

右回転か左回転のどちらかを等しい確率で選択することでショットを生成する。ここで、 $X_1(X_2)$ はプレイエリアの左下隅(右下隅)を右回転(左回転)で狙うショットの初速度の x 成分に等しい。また、 Y_2 はホッグラインの中央を狙うショットの初速度の y 成分であり、 Y_1 はデジタルカーリングで許容される最も大きな初速度の y 成分である。なお、ゲーム上で意味のあるショットの初速度の y 成分は負の値となっている。

7.2 学習内容

本研究においてニューラルネットワークで学習する内容は、 n 投目のゲーム状態とそのゲーム状態でのショットを入力として、そのエンドでの最終的な得点のクラス分類を出力するものとする。1 エンドで取り得る得点は後攻 8 点から先攻 8 点の 17 通り存在するが、3 点以上得点することは稀であるため、本研究においては後攻 3 点以上から先攻 3 点以上の 7 クラス分類としている。

訓練データは学習と並行してランダム方策 AI 同士で対戦することで生成し、1 度学習に用いた訓練データは再利用しない。学習時のミニバッチの大きさは 32 とし、学習の反復回数は 500000 回とする。学習率 ϵ は学習の反復回数によらず $\epsilon = 0.01$ で固定する。

8. 実験

本研究では 3 つの実験を行う。実験 1 では、16 投目の得点確率関数を学習することで、入力画像の構成による近似精度への影響を調べ、以降の実験での学習に用いる入力画像構成を決める。実験 2 では、16 投分の得点確率関数を学習し、その性能をテストデータを用いて評価する。実験 3 では、実験 2 で学習したニューラルネットワークを用いた AI を作成し、実際にデジタルカーリングで対戦させることでその性能を調べる。

8.1 実験 1

入力画像の構成による近似精度への影響を調べるために、表 1 の 3 種類の入力画像の構成 a, b, c でそれぞれ学習し、性能を比較する。ただし、ストーンの配置画像とはストーンが含まれる画素が 1、ストーンが含まれない画素が 0 の値を持つ画像であるとする。ハウス小とはハウスを構成する 4 つの同心円のうち半径が 2 番目に小さいものであり、ハウス中は 3 番目に小さい同心円、ハウス大は半径が最も大きい同心円を意味する。また、入力に用いるショットの初速度は、実際の初速度 v_x^*, v_y^* を、

$$v_x = \frac{X_2 - v_x^*}{X_2 - X_1}$$

$$v_y = \frac{Y_2 - v_y^*}{Y_2 - Y_1}$$

として変換した $v_x, v_y \in [0, 1]$ とする。

16 投目のゲーム状態とショットを訓練データとして、各

表 1 入力画像の構成

入力画像			内容
a	b	c	
o	o	o	全てのストーンの配置画像
o	o	o	先攻のストーンの配置画像
o	o	o	後攻のストーンの配置画像
o	o	x	ハウス小の内側を 1 で塗りつぶした画像
o	o	x	ハウス中の内側を 1 で塗りつぶした画像
o	o	x	ハウス大の内側を 1 で塗りつぶした画像
o	o	o	初速度 v_x
o	x	o	$\min(1, 2v_x)$
o	x	o	$\min(0, 2v_x - 1)$
o	x	o	$\min(1, \max(2v_x - 1/2))$
o	o	o	初速度 v_y
o	x	o	$\min(1, v_y)$
o	x	o	$\min(0, 2v_y - 1)$
o	x	o	$\min(1, \max(2v_y - 1/2))$
o	o	o	回転方向 (0 or 1)

表 2 [実験 1] 各入力画像構成の正答率平均

入力画像構成	正答率平均
a	0.63(10)
b	0.61(12)
c	0.36(6)

入力画像毎にニューラルネットワークを学習した。学習結果は重みの初期値や学習に用いる訓練データに依存する。平均的な性能を評価するために、各学習はこれらの初期値やデータを変更して 16 回行った。

性能比較は同一のテストデータを入力とした際の正答率を比較することで行う。ここで、正答率とはテストデータの総数に対して正しく分類したデータの個数の割合である。正しく分類するとは、値が最大の出力を与えるクラスが実測されたクラスと一致することである。テストデータは各クラス毎に 1000 個ずつ、計 7000 個用意し、内容はランダム方策 AI 同士で 1 エンド対戦した際の 1 投毎のゲーム状態とショットの情報及びそのエンドの最終得点とする。

表 2 に各入力画像構成で学習したニューラルネットワークのテストデータに対する正答率平均を示す。括弧内の誤差は標準誤差から求めた 95%信頼区間により見積もる。

画像構成 c が a, b に対して有意に劣る結果となった。ハウスの範囲を表す画像によって、ストーンの位置関係からの得点認識能力が向上したと考えられる。一方で、画像構成 a, b 間の正答率平均には有意な差は見られない。a の構成はショットに関する入力を増やすことで、ショットの初速度の細かい違いを認識しやすくすることを狙いとしていたが、本実験ではその優位性は確認できない。

8.2 実験 2

表 1 の a の入力画像構成を用いて 1 投目から 16 投目までそれぞれ対応するニューラルネットワークの学習を行い、実験 1 と同様のテストデータに対する正答率を調べる。

表 3 に n 投目の学習を行なったニューラルネットワー

表 3 [実験 2] 各ニューラルネットワークの正答率

n	正答率	n	正答率
1	0.1454	9	0.2781
2	0.1684	10	0.3200
3	0.1634	11	0.3393
4	0.1866	12	0.3996
5	0.1956	13	0.4519
6	0.2079	14	0.5266
7	0.2460	15	0.5757
8	0.2696	16	0.6961

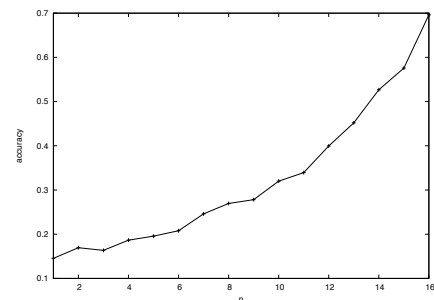


図 5 [実験 2] 各ニューラルネットワークの正答率

クの正答率を示す。また、図 5 に表 3 の内容を横軸を n 、縦軸を正答率としてグラフにしたものを示す。

3 投目を除き、 n が大きくなるに従って正答率も上昇していく様子が見られる。エンド後半のゲーム状態やショットの方が直接得点に結びつく可能性が高いためであると考えられる。

テストデータに対して、入力に無関係に一樣な確率でランダムにクラス分類を行なったときその正答率は $1/7 \approx 0.1429$ となる。いずれのニューラルネットワークの正答率もランダムなクラス分類の正答率を上回っていることから、学習によって一樣分布よりは良い確率分布を表現できることが確認された。

8.3 実験 3

実験 2 で重みを調整したニューラルネットワークを用いて得点の期待値を最大化するショットを選択する AI を作成する。ショットの探索には 6 章で述べた 4 つの手法を用いる。これらの AI とランダム方策 AI 及び単純なルールベース AI との対戦を行い性能を比較する。

単純なルールベース AI は以下の様にショットを選択する。ただし、ショットの回転方向は左右どちらかを等しい確率によって決定する。

表 4 [実験 3] 1 エンド目先攻の勝利点平均

後攻 先攻	ランダム				
	直線探索付	初期点 4 点	格子点	サンプリング	ランダム方策
直線探索付	*	-0.12(10)	0.40(9)	-0.28(10)	0.73(7)
初期点 4 点	0.26(10)	*	0.15(10)	0.02(10)	0.75(7)
格子点	-0.23(10)	0.03(10)	*	-0.46(9)	0.57(8)
ランダム					
サンプリング	0.21(10)	0.40(9)	0.36(9)	*	0.27(9)
ランダム方策	-0.54(8)	-0.76(6)	-0.65(8)	-0.57(8)	*

ルールベース AI

- ハウス内にストーンが存在しない場合にはティーを狙うショットを選択.
- ハウス内に存在するストーンのうち, 最もティーに近いストーンが相手のストーンであれば, そのストーンをテイクアウトするショットを選択.
- ハウス内に存在するストーンのうち, 最もティーに近いストーンが自分のストーンであれば, そのストーンをガードするためにそのストーンの 2m 下方 (ホッグライン側) を狙うショットを選択.

表 4 に 1 ゲーム 10 エンドとして各 AI の組合せで 200 ゲームずつ対戦を行なった際の 1 エンド目先攻の勝利点平均を示す. 括弧内の誤差は標準誤差により求めた 95%信頼区間により見積もる. 勝利点は先攻が勝った場合に 1, 引き分けで 0, 負けた場合に -1 を与えるものとする. また, *は対戦を行っていない組合せであることを表す. なお, ルールベース AI は今回検討した他の全ての AI に対して全勝したため表では省略する.

表 4 より強い順に並べると, ルールベース AI, ランダムサンプリング, 初期点 4 点の最急降下法, 直線探索付きの最急降下法, 格子点評価, ランダム方策となる.

格子点評価よりも最急降下法の方が良い結果となったことから, 格子点を勾配方向に改善することで性能が向上したと言える.

ランダムサンプリングが格子点評価より良い結果となったのは, 単純にランダムサンプリングの方が評価するショットの数が多いことが要因の 1 つであると考えられる. また, $v_x = 0, 1$ 付近や $v_y = 1$ 付近が良いショットとなることはゲームの性質上少ないが, 格子点評価ではこの近辺に多くの評価回数を割り当てていることも影響しているだろう.

最急降下法がランダムサンプリングに劣る結果となった. 図 3 に示されているように確率関数が激しく上下しているために, 勾配ベクトルが降下方向として良い指標となっていない可能性がある.

学習した Q^π の性能確認のために, ある典型的なゲーム状態でショットが左回転であるときの 16 投目の学習を行ったニューラルネットワークによる得点の期待値を解析したものを図 6 に示す. このゲーム状態ではプレイエリア内にはティーにある相手のストーン 1 つのみ存在し, このストーンに衝突するようなショットを行えば高確率で自分が得点できる. このゲーム状態においては得点の期待値は最

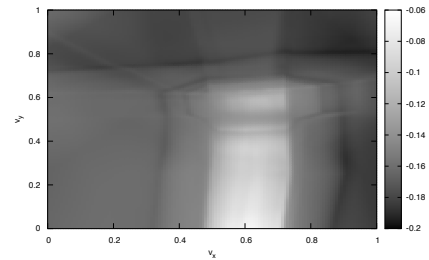


図 6 ある典型的なゲーム状態 s における $Q^\pi(s, v_x, v_y, \text{curve})$

大の点では正となることが望ましいが, 図 6 では得点の期待値が最大の点でも負となっているため, 得点期待値が十分に正しくは表現されていないと考えられる. このことは, ルールベース AI に劣る程度の性能となっている原因の 1 つだろう.

図 6 からは (0.6, 0.6) 付近, (0.6, 0.5) 付近, (0.6, 0) 付近等複数の極大値を与える点が存在する様子が読み取れる. これらの点は概ねティーにある相手のストーンに衝突するショットを表しているため, 相手のストーンに衝突させることで自分が得点できるという概念はある程度は習得していると期待できる.

9. おわりに

本研究ではランダム方策 $\pi^{(0)}$ の得点期待値 $Q^{\pi^{(0)}}$ を畳込みニューラルネットワークで表した. この $Q^{\pi^{(0)}}$ から得たグリーディ方策 $\pi^{(1)}$ はランダム方策 $\pi^{(0)}$ よりは優秀であるが, 現状ではとても弱いものである. 今後は本研究で行ったような方策改善を $Q^{\pi^{(1)}} \rightarrow Q^{\pi^{(2)}} \rightarrow \dots$ と繰り返すことで実用的な価値関数の開発を目指したい.

10. 付録 1 階偏導関数の計算法

グリーディ方策に従うショットを最急降下法によって探索する際に用いる勾配の計算法を記す. 以下に各層の順伝播の際の計算式と, 入力 x による 1 階偏導関数を求める際の計算式を掲載する. L 層のニューラルネットワークにおいて, $u^{(1)}$ を入力として以下の基本形の式を $l = 1, 2, \dots, L-1$ と層毎に順に計算することで得られる $u^{(L)}$ が出力となる. また, $\frac{\partial u^{(1)}}{\partial x}$ を入力を偏微分した値として, 以下の 1 階偏導関数の式を $l = 1, 2, \dots, L-1$ と層毎に順に計算することで得られる $\frac{\partial u^{(L)}}{\partial x}$ が出力の 1 階偏導関数値となる. なお, 厳密には ReLU 層と最大プーリング層には微分不可能点が存在するが, 本研究においては片側微分を用いる.

- 全結合層

– 基本形

$$u_j^{(l+1)} = \sum_i w_{ji}^{(l+1)} u_i^{(l)} + b_j^{(l+1)}$$

– 1 階偏導関数

$$\frac{\partial u_j^{(l+1)}}{\partial x} = \sum_i w_{ji}^{(l+1)} \frac{\partial u_i^{(l)}}{\partial x}$$

変数説明

$u_j^{(l+1)}$ $l+1$ 層の j 番目のユニットの出力
 $w_{ji}^{(l+1)}$ l 層目の i 番目の出力から $l+1$ 層の j 番目のユニットへの入力にかかる重み
 $b_j^{(l+1)}$ $l+1$ 層の j バイアス

畳込み層

基本形

$$u_{i,j,m}^{(l+1)} = \sum_{n=0}^{N^{(l)}-1} \sum_{p=0}^{K_h^{(l+1)}-1} \sum_{q=0}^{K_w^{(l+1)}-1} h_{p,q,n,m}^{(l+1)} u_{s^{(l+1)}i+p, s^{(l+1)}j+q, n}^{(l)} + b_m^{(l+1)}$$

1 階偏導関数

$$\frac{\partial u_{i,j,m}^{(l+1)}}{\partial x} = \sum_{n=0}^{N^{(l)}-1} \sum_{p=0}^{K_h^{(l+1)}-1} \sum_{q=0}^{K_w^{(l+1)}-1} h_{p,q,n,m}^{(l+1)} \frac{\partial u_{s^{(l+1)}i+p, s^{(l+1)}j+q, n}^{(l)}}{\partial x}$$

変数説明

$u_{i,j,m}^{(l+1)}$ $l+1$ 層 m チャンネルの位置 (i, j) の画素値
 $N^{(l)}$ l 層の画像のチャンネル数
 $K_h^{(l+1)}, K_w^{(l+1)}$ $l+1$ 層のカーネルの高さ, 幅
 $s^{(l+1)}$ $l+1$ 層のスライドの大きさ
 $h_{p,q,n,m}^{(l+1)}$ l 層の画像の n チャンネルから $l+1$ 層の画像の m チャンネルに対応するカーネルの位置 (p, q) の重み
 $b_m^{(l+1)}$ m チャンネルに対応するバイアス

最大プーリング層

基本形

$$u_{i,j,m}^{(l+1)} = u_{s^{(l+1)}i+p^*, s^{(l+1)}j+q^*, m}^{(l)}$$

1 階偏導関数

$$\frac{\partial u_{i,j,m}^{(l+1)}}{\partial x} = \frac{\partial}{\partial x} u_{s^{(l+1)}i+p^*, s^{(l+1)}j+q^*, m}^{(l)}$$

変数説明

$u_{i,j,m}^{(l+1)}$ $l+1$ 層 m チャンネルの位置 (i, j) の画素値
 $K_h^{(l+1)}, K_w^{(l+1)}$ $l+1$ 層のカーネルの高さ, 幅
 $s^{(l+1)}$ $l+1$ 層のスライドの大きさ
 p^*, q^* $0 \leq p < K_h, 0 \leq q < K_w$ で $u_{s^{(l+1)}i+p, s^{(l+1)}j+q}^{(l)}$ に最大値を与える p, q のうち, $pK_w + q$ の値が最大となるもの

ReLU 層

基本形

$$u_i^{(l+1)} = \max(0, u_i^{(l)})$$

1 階偏導関数

$$\frac{\partial u_i^{(l+1)}}{\partial x} = \begin{cases} 0 & (u_i^{(l)} < 0) \\ \frac{\partial u_i^{(l)}}{\partial x} & (u_i^{(l)} \geq 0) \end{cases}$$

変数説明

$u_i^{(l+1)}$ $l+1$ 層目の i 番目の出力

ソフトマックス層

基本形

$$u_i^{(l+1)} = \frac{e^{u_i^{(l)}}}{\sum_j e^{u_j^{(l)}}}$$

1 階偏導関数

$$\begin{aligned} \frac{\partial u_i^{(l+1)}}{\partial x} &= u_i^{(l+1)} (1 - u_i^{(l+1)}) \frac{\partial u_i^{(l)}}{\partial x} \\ &\quad - \sum_{j \neq i} u_i^{(l+1)} u_j^{(l+1)} \frac{\partial u_j^{(l)}}{\partial x} \end{aligned}$$

変数説明

$u_i^{(l+1)}$ $l+1$ 層目の i 番目の出力

参考文献

- [1] 大渡勝己, 田中哲朗: カーリング AI に対するモンテカルロ木探索の適用, ゲームプログラミングワークショップ 2016 論文集, Vol. 2016, pp. 180–187 (2016).
- [2] 加藤修, 飯塚博幸, 山本雅人: デジタルカーリング AI 「じりつくん」, *GAT2016* (2016).
- [3] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T. and Hassabis, D.: Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol. 529, pp. 484–503 (online), available from <http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html> (2016).
- [4] 北清勇磨, 伊藤毅志: デジタルカーリングシステムの提案と構築, 第 9 回 E&C シンポジウム, pp. 13–16 (2015).
- [5] 岡谷貴之: 深層学習, 講談社 (2015).
- [6] Sutton, R. S. and G. Barto, A.: 強化学習, 森北出版 (2000).
- [7] 田村明久, 村松正和: 最適化法, 共立出版 (2002).