カードゲーム型対戦環境への強化学習の適用

1 はじめに

近年, 人工知能に関する研究分野は目覚ましい発展を遂げており様々な分野に応用されている. その中でも人間の脳の働きに近いとされる強化学習と深層学習を組み合わせた深層強化学習は自動運転やロボット, 推薦システムに活用されており実生活の問題解決に寄与している.

深層強化学習はゲームにも盛んに応用されている. 特に将棋や囲碁といった, プレイヤーが意思決定をする段階でそれ以前の意思決定の過程が全て把握可能な完全情報ゲームへの応用が有名である. 最近では麻雀やポーカーのような, プレイヤーに与えられる情報が部分的である不完全情報ゲームへの応用も注目されている.

そこで, 本研究では不完全情報ゲームであるトレーディングカードゲームを参考にカードゲーム型対戦環境を構築し, 構築環境への強化学習の適用と構築環境のゲームバランスの調整を目標とする. 本稿では対戦環境の構築と構築環境への強化学習の適用について述べる.

2 要素技術

2.1 OpenAI Gym

OpenAI Gym [1] は非営利企業 OpenAI が提供する強化学習のシミュレーション用プラットフォームであり, 強化学習の環境として多くのゲームが登録されている. 提供されているインターフェースに沿って, 環境におけるエージェントの行動空間や状態空間, 報酬などを定義, 実装することで自作の環境を登録し利用することができる. シミュレーション環境と強化学習アルゴリズム間のインターフェースが確立されており様々な強化学習用ライブラリに対応しているため比較的容易に強化学習を試すことができる.

2.2 Q 学習

強化学習では, エージェントが行動することで環境から報酬を得る. 強化学習における行動はその直後に獲得する報酬の大きさではなく, 未来に渡っての報酬の総和を見積もった値である「価値」の最大化に繋がるかという観点で評価される. 価値の最大化を目指す場合にはある状態 s において行動 a をとった時の価値が分かればよい. この価値のことを Q 値, あるいは行動価値関数と呼ぶ. Q 学習ではエージェントの 1 ステップごとに (1) 式に示す更新式で Q 値を更新する.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)) \quad (1)$$

なお t は時間, r は報酬, α は学習率, γ は割引率を表す.

2.3 Deep Q Network

Q 学習では実装の際に Q 値のテーブルを作成する. しかし状態空間や行動空間が高次元である, あるいは状態や行動が離散値ではなく連続値で表現される場合には Q テーブルのメモリ量は爆発してしまう. この問題を解決した技術が Deep Q Network (DQN) [2] である. DQN ではニューラルネットワークを用いて, ある状態における行動ごとの Q 値を推定する. エージェントが経験した過去の体験を Replay Memory に一定期間保存しておき, 過去の経験をランダムにサンプリングして学習する Experience Replay や行動を決定する Q 値のネットワークと Q 値の学習を行うネットワークを分けることで Q 値の過大評価を防ぐ Fixed Target Network といった工夫により安定した学習を可能としている.

2.4 モンテカルロ探索

モンテカルロ探索 (Monte Carlo Search : MCS) は Q 学習と同様に Q 値を推定する学習アルゴリズムであるが, Q 学習のように 1 ステップごとに Q 値を更新するのではなく, 1 エピソードをランダムに行動し, 終了状態に到達してから辿ったステップ $t = 1 \sim (\mathrm{T} - 1)$ について Q 値を更新する. Q 値の更新は学習率 α , 割引率 γ , エピソードから得られた割引現在価値 G_t を用いて (2) 式に従う.

$$Q(s_t, a_t) \leftarrow (1 - \alpha) Q(s_t, a_t) + \alpha * G_t \quad (2)$$

$$\text{where} \quad G_t = r_{t+1} + \gamma r_{t+2} + ... + \gamma^{\mathrm{T}-t-1} r_{t+k+1}$$

3　カードゲーム型対戦環境の構築

3.1　トレーディングカードゲーム

今回の実験で構築したカードゲーム型対戦環境は, HearthStone [1] や Shadowverse [2] といったトレーディングカードゲーム (Trading Card Game : TCG) を参考にした. TCG は 2 人のプレイヤーからなるゲームである. 囲碁や将棋のようにプレイヤーは先攻と後攻に分かれ, ターン制で進んでいく. ゲームタイトルごとに細かいルールは異なるが, 多くの場合相手プレイヤーのカードの 1 部分はプレイヤーから観測できない不完全情報ゲームである.

3.2　構築環境

実装したカードゲームのルールと用語を説明する. ゲームは 2 人のプレイヤーからなり, プレイヤーは複数のカードからなるデッキを持つ. プレイヤーは手札, 盤面と呼ばれるカードを保有する領域を持ち, ドローと呼ばれる操作でカードをデッキから手札に加える. また, プレイと呼ばれる操作でカードを手札から盤面に出す. また, デッキからカードが無くなった状態をデッキ切れと呼ぶ.

カードはそれぞれ攻撃力と HP の 2 つの数値を持つ. 盤面にあるカードは対戦相手の盤面にあるカードに攻撃することができる. ただし, 攻撃が可能となるのはカードがプレイされたターンの次のターンからになる. カードが攻撃を行う際には, 相手盤面に存在する攻撃対象のカードの HP へカードの持つ攻撃力分ダメージを与える. またカードが攻撃する際には攻撃対象のカードが持つ攻撃力分, 攻撃するカードもダメージを受ける. カードの HP が 0 になった, あるいは後述する手札と盤面の枚数制限を超えて盤面にプレイされた時はカードは破壊される. 破壊されたカードは捨て札となり盤面や手札から消え, ゲーム中には再出現しない.

3.3　ゲームフロー

実装したゲームの流れを説明する.

1. ゲーム開始時に各プレイヤーは自身のデッキをシャッフルする.
2. デッキからカードを 3 枚ドローする.
3. 先攻プレイヤーは 1 ターン目の行動をする.
4. 後攻プレイヤーはカードを 1 枚ドローして行動する.
5. 2 ターン目以降は先攻プレイヤーもカードを 1 枚ドローしてから行動する.
6. 4 , 5 を繰り返す.
7. どちらかのプレイヤーがデッキ切れになっている状態で, カードをドローしようとしたらゲーム終了となる.

勝利条件は 4.3 節で述べる. 本構築環境では, 一般的な TCG と同様に先攻プレイヤーがカードの行動が早いため有利となる. そのため, 1 ターン目のドローといったメリットを後攻に加えている.

Algorithm 1 対戦相手の行動ルーチン

```
盤面にカードを 1 枚プレイ
for 盤面のカード (プレイ順が古い方から) do
    if 敵の盤面にカードがある then
        ランダムに攻撃対象を選んで攻撃
    else
        何もしない
    end if
end for
ターンを終了
```

4　実験

構築環境へ強化学習が適用できるか検証した. 強化学習手法として DQN と MCS を用いた. 2 つの手法について構築環境において後攻のプレイヤーとして同程度学習し, 学習済みのモデルを用いて 10000 回ゲームを行い勝率を計算した. また, 学習が進んでいるかどうか判断するため学習時の獲得報酬の推移を記録した.

4.1　対戦相手の行動ルーチン

学習, 勝率計算の際は学習するプレイヤーの対戦相手を用意する必要がある. Algorithm 1 に今回の実験における対戦相手の行動ルーチンを示す.

[1] https://hearthstone.blizzard.com/ja-jp
[2] https://shadowverse.jp/

表 1: デッキの内容

攻撃力	HP	枚数
3	3	5
2	4	5
2	3	5

4.2 デッキ

学習側, 対戦相手ともに同じデッキを持つ. デッキの内容は表 1 に示す.

4.3 勝利条件

ゲーム終了時に,
(学習側盤面の枚数) > (対戦相手盤面枚数) ならば, 学習側の勝利,
(学習側盤面の枚数) ≤ (対戦相手盤面枚数) ならば, 対戦相手の勝利とした.

4.4 状態空間と行動空間, 報酬の定義

強化学習では, エージェントの取りうる行動と観測できる状態の空間, 報酬を定義する必要がある. TCG ではドローやプレイ, カードの攻撃による破壊といった行動で盤面や手札の枚数が変化する場合があり, 各ステップ時点でプレイヤー取りうる行動の次元数が変わるため学習が困難である.

そのため本研究では予め手札と盤面の枚数の上限をそれぞれ 9 枚, 5 枚と定め , 手札と盤面に存在するカードに自盤面 1 というように番号をつけ, カードが存在しない場合は状態を 0 とすることで状態空間と行動空間を定義した. 表 2 , 3 に状態空間, 行動空間の定義を示す. なお, ドローやプレイといった操作でカードを追加し枚数の上限を超える場合には追加しようとしたカードを破壊する. また, 学習側は表 3 に基づいて, 保有している手札と盤面のカード全てについて盤面に出すか出さないか, 攻撃するかしないかを行動し終えた段階でターン終了とする.

reward は以下のように設定した.

- 1 ステップ終了後

$$\text{reward} = 0.0$$

- 1 エピソード終了後

$$\text{reward} = \begin{cases} 1.0 & \text{(学習プレイヤーの勝利)} \\ -1.0 & \text{(敵プレイヤーの勝利)} \end{cases}$$

表 2: 定義した状態空間

状態説明	次元数	最小値	最大値
手札 1 ～ 9 の HP と攻撃力	18	0	20
自盤面 1 ～ 5 の HP と攻撃力	10	0	20
敵盤面 1 ～ 5 の HP と攻撃力	10	0	20
自盤面 1 ～ 5 が攻撃可能かどうか	5	0	1
お互いのデッキの残り枚数	2	0	15

表 3: 定義した行動空間

行動説明	次元数
手札 1 ～ 9 を自盤面に出す	9
手札 1 ～ 9 を自盤面に出さない	9
自盤面 1 が敵盤面 1 ～ 5 に攻撃 or 何もしない	6
自盤面 2 が敵盤面 1 ～ 5 に攻撃 or 何もしない	6
自盤面 3 が敵盤面 1 ～ 5 に攻撃 or 何もしない	6
自盤面 4 が敵盤面 1 ～ 5 に攻撃 or 何もしない	6
自盤面 5 が敵盤面 1 ～ 5 に攻撃 or 何もしない	6

4.5 DQN, MCS のパラメータ

表 4 , 5 に実験で用いた 2 種の強化学習アルゴリズムのパラメータを示す.

5 結果

表 6 に実験結果を示す. なお, ベースラインとして対戦相手と同じ戦略のプレイヤーを後攻に配置して 10000 回対戦し勝率を計算した.

図 1 , 2 に MCS と DQN における学習時の平均獲得報酬の推移を示す. 図 1 , 2 において縦軸は reward , 横軸はエピソード数であり, 図中の緑点は 学習時の 100 エピソードにおける平均獲得報酬を表し, 薄緑の領域は標準偏差を表す. DQN , MCS 共にベースラインと比べて遥かに高い勝率を記録した. 特に, DQN は約 9 割もの勝率を記録している. 図 1 から, 学習が進んでいくにつれ平均獲得報酬が増加しており構築環境において強化学習が適用できていると考えられる. また, 図 2 に示す MCS より高い勝率を記録した DQN の推移からは, MCS に比べて短いエピソード数で学習が安定していると考えられる.

表 4: DQN のパラメータ

パラメータ名	値
割引率 γ	0.99
全結合層の活性化関数	ReLU
全結合層の次元	64
最適化アルゴリズム	Adam
方策	ε -greedy
ε	0.1
Target Network 更新重み	0.5
Exprience Memory 開始ステップ数	1.0×10^4
学習ステップ数	5.0×10^6

表 5: MCS のパラメータ

パラメータ名	値
学習率 α	0.5
割引率 γ	0.99
学習エピソード数	8.5×10^4

6 まとめと今後の課題

今回の実験では簡易的なルールを用いたカードゲーム型対戦環境を構築し, 構築環境へ DQN や MCS といった強化学習手法を適用した. 構築環境において DQN と MCS を用いて高い勝率を記録するエージェントを作成することができ, 学習が進んでいくにつれてエージェントがより高い報酬を得るよう学習していることが実験により確認できた.

今後の課題として, 研究の最終目的であるカードゲーム型対戦環境のゲームバランス調整に取り組むことが挙げられる.

また, 一般的な TCG ではプレイヤー自身に HP を定義しカードがプレイヤーに攻撃できるようにする, カードにコストというパラメータを設けて盤面にプレイする際に制約を設けるといった仕様を加えることでよりゲームの戦略性を深めている. その一方で今回作成した環境はゲームの終了条件と勝利条件が直接関係していない, 1 ターンに出そうと思えば何枚もカードをプレイできるといったようにゲームデザインが詰めきれていない. そのため, プレイヤーの HP やカードのコストといった仕様を構築環境に取り入れ, より戦略性の高いカードゲーム型対戦環境を作成することも今後の課題である.

表 6: 実験結果

手法	勝率
DQN	**0.9069**
MCS	0.7257
対戦相手と同じ戦略	0.1294

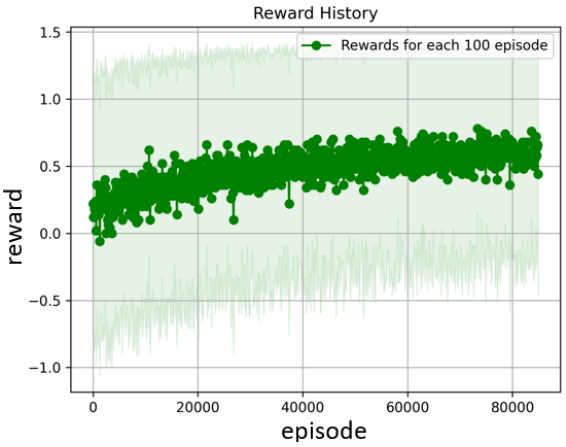

図 1: MCS における平均獲得報酬の推移

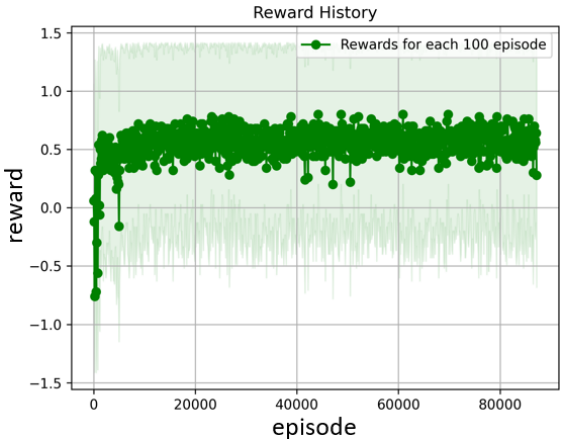

図 2: DQN における平均獲得報酬の推移

参考文献

[1] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv e-prints*, p. arXiv:1606.01540, June 2016.

[2] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling Network Architectures for Deep Reinforcement Learning. *arXiv e-prints*, p. arXiv:1511.06581, November 2015.