2022 年 11 月 22 日 B3 西村昭賢

進捗報告

1 今週やったこと

- Docker の環境構築
- 自作カードゲーム環境の改良と実験

2 Docker の環境構築

先週まで上手く行かなかったが, 先週火曜の B3 実験の時間に先輩方にご教授頂きいろいろ試行錯誤した結果サーバー使って実験できるようになった. 坂川さん岡本さんありがとうございます.

3 カードゲーム環境の改良と実験

先週までは, 予め決まった盤面状況の中でデッキから手札へのドロー, 手札から盤面のプレイなどを考えずに自プレイヤーの行動→敵プレイヤーの行動を 1 エピソードとする学習しか実現できなかった. 今週は先週検討した予め盤面と手札の最大枚数を決定し行動空間と状態空間を予め定めておく方法を軸に実装をし [1], ゲーム開始から終了までを 1 エピソードとした実験を回せるように環境を改良した. また, 改良した環境で実験した.

3.1 カードゲームのルール

先週森先生に頂いたアドバイスを踏まえてプレイヤーの体力やカードのコストは実装せずに簡単なルールのカードゲームを作成した.

- プレイヤー
  プレイヤーは最大 9 枚の手札, 最大 5 枚まで盤面にカードを持つことができる. 枚数制限を超えた場合, 新たに加えようとしたカードは破壊される. ゲーム開始時にデッキから 3 枚ドローする. その後はターンが回ってくると自動でデッキから 1 枚ドローする.
- カード
  カードは攻撃力と HP を持つ. 相手のカードに攻撃することができ, 攻撃した際には攻撃対象のカードの攻撃力分ダメージを受ける.
- デッキ
  15 枚のカードからなる. ゲーム開始時にシャッフルされる.
- 終了条件
  お互いのプレイヤーのデッキ, 手札の両方からカードが無くなったらゲーム終了.
- 勝利条件
  盤面のカード枚数が多いプレイヤーの勝利. カードの枚数が同じだった場合には盤面のカードの攻撃力と HP の総和を計算し, (先攻プレイヤーの総和) $\geqq$ (後攻プレイヤーの総和) ならば先攻プレイヤーの勝利, そうでなければ後攻プレイヤーの勝利.

3.2 行動空間の次元

以前から述べているが, OpenAI Gym に自作環境を定義する際は行動空間の次元, 状態空間, 報酬を定義する必要がある.

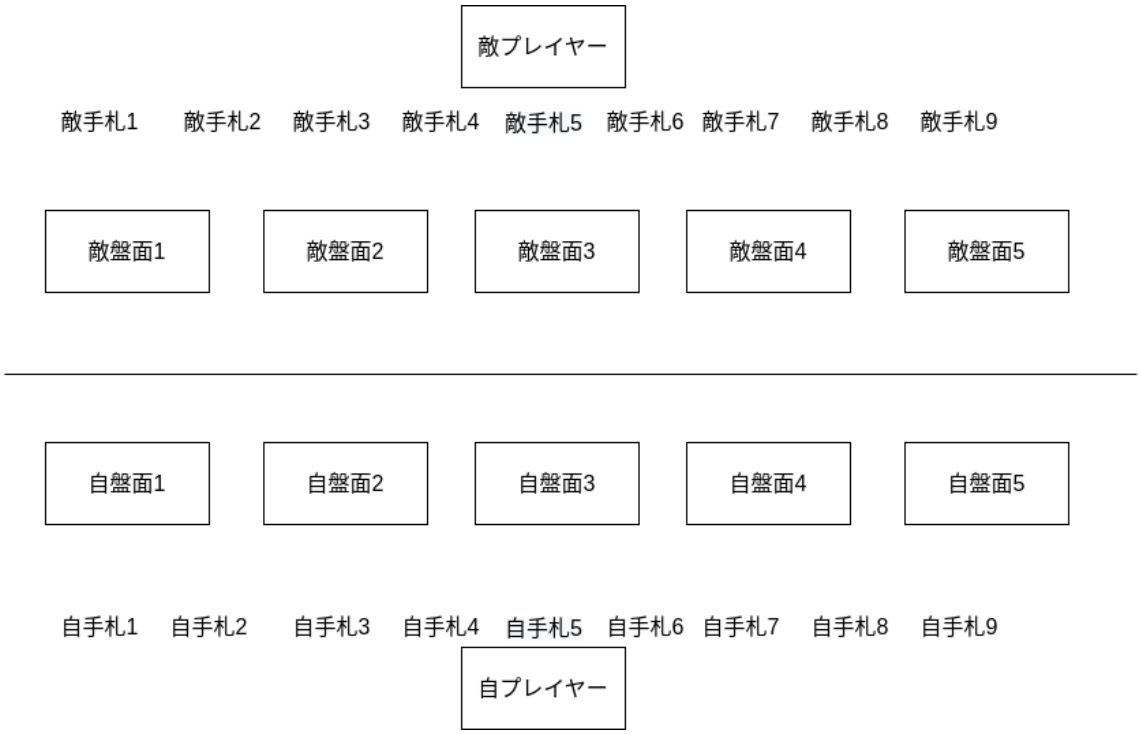

図 1: 定義した空間

先週決めたように今回のルール設定では手札の最大枚数を 9, 盤面の最大枚数を 5 と決めている. そのため, プレイヤーの行動は

- 自手札 1 - 9 を盤面に出す … 9 通り
- 自盤面 1 - 5 が敵盤面 1 - 5 を攻撃 … 5 * 5 = 25 通り
- ターンエンド … 1 通り

の計 35 通りとなり, 行動空間の次元は 35 次元と定義した.

3.3 状態空間の定義

プレイヤーの観測できる状態として以下の計 43 個のパラメータを定義した.

- 自手札 1 - 9 の攻撃力と HP … 18 個
- 自盤面 1 - 5 の攻撃力と HP … 10 個
- 敵盤面 1 - 5 の攻撃力と HP … 10 個
- 自盤面 1 - 5 がターン中行動可能か … 5 個

3.4 報酬

最適な報酬の与え方はまだ定まってはいないが, 以下で述べる実験で採用した報酬設定は以下の通りである. なお, 学習する側のプレイヤーの 1 行動を 1 ステップ, ゲーム開始から終了までを 1 エピソードとしている.

$$1\text{ステップ終了後}\quad reward = \begin{cases} 1.0 & (\text{後攻プレイヤーのターン終了時先述の勝利条件で勝利していた場合}) \\ -1.0 & (\text{後攻プレイヤーのターン終了時先述の勝利条件で敗北していた場合}) \\ -30.0 & (\text{ターンが回ってきたステップで即ターンエンドした場合}) \\ 0.0 & (otherwise) \end{cases}$$

$$1\text{エピソード終了後}\quad reward = \begin{cases} 10.0 & (\text{先述の勝利条件で勝利していた場合}) \\ -10.0 & (\text{先述の勝利条件で敗北していた場合}) \end{cases}$$

カードの枚数が同じだった場合にドローとしない, 即ターンエンドにペナルティをつける, 後攻プレイヤーの行動後に 0.0 以外の報酬を渡す用に設定した理由としては, 後述する実験においてこれら 3 つの処理を行わなかった際に学習が安定しなかったためである. 環境から与えられる報酬が疎となるためと考えられる.

3.5 実験設定

作成した環境で学習できるか, また学習結果は妥当性のある結果となるか検証するために以下の設定で実験した.

学習するのは先攻プレイヤーであり, 後攻プレイヤーは毎ターン 1 枚盤面にカードを出しその後は盤面にある行動可能なカードでランダムに対象を選択して攻撃する. 先攻, 後攻プレイヤーのデッキを表 1 に示す.

表 1: 先攻, 後攻プレイヤーのデッキ
() 内の数字は (攻撃力 , HP) を意味している.

先攻	後攻
(3 , 3) × 5	(3 , 3) × 6
(2 , 3) × 5	(1 , 5) × 3
(2 , 4) × 5	(4 , 2) × 3
	(3 , 2) × 3

また実験では Deep-Q-Network であるステップ回学習を行い, 10000 エピソード検証し勝率を求める. この操作を 10000, 50000, 100000, 150000, 200000 ステップと変更して行い勝率を比較した.

図 2 , 表 2 に実験で使用したモデルとモデルのパラメータ設定を示す.

表 2: DQN のパラメータ [2]

方策	ε -greedy
ε	0.1
全結合層の活性化関数	ReLU
全結合層の次元	16
最適化アルゴリズム	Adam
学習率	1e-3
Experience Replay のメモリ量	1000000

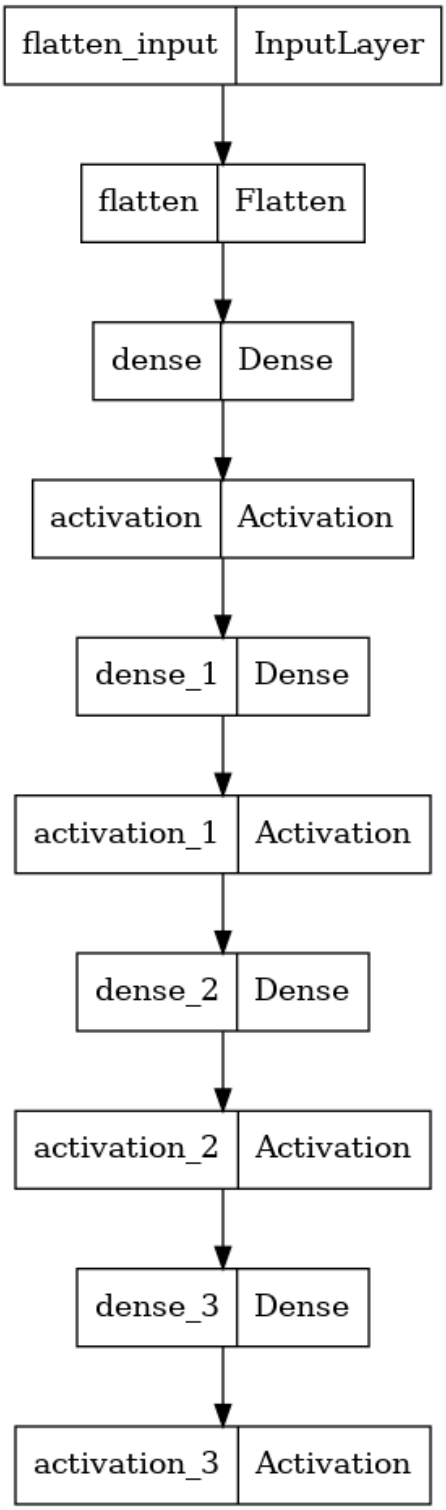

図 2: 実験で用いたモデル

3.6　実験結果

勝率は 10000 エピソードの検証を 5 回行いその平均値を用いた. 結果を表 3 に示す.

表 3: 各ステップ数における勝率

ステップ数	勝率
10000	0.57336
50000	0.63074
100000	0.62650
150000	0.71484
200000	0.70110

また, 各ステップ数の実験時のエピソードごとの reward, step の推移を図 3 - 7 に示す.

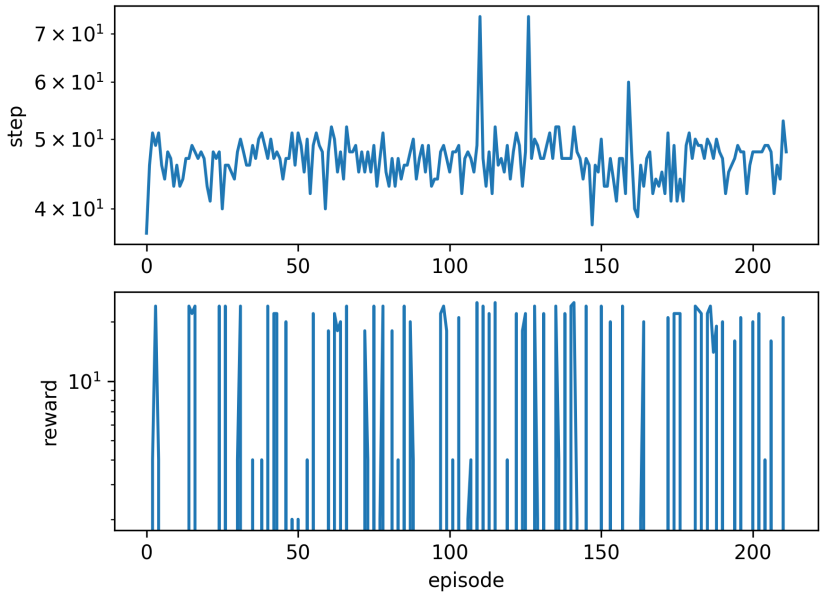

図 3: ステップ数 10000 の際の推移

3.7　考察, 検討

多少の誤差はあれどステップが増えれば勝率は上がっているといえる. ただ, デッキのシャッフルや相手の行動でランダム要素が含まれているとはいえ勝率 7 割という数値が妥当性があるかは検討の余地がある. 学習率のパラメータを調整する必要があるのか, 報酬の与え方に問題があるのか, どちらにせよモデルと環境側それぞれについて検討していきたい. また keras と Optuna を用いてパラメータの調整を行っている参考資料も見つけたので Optuna といったフレームワークも試してみたい [3].

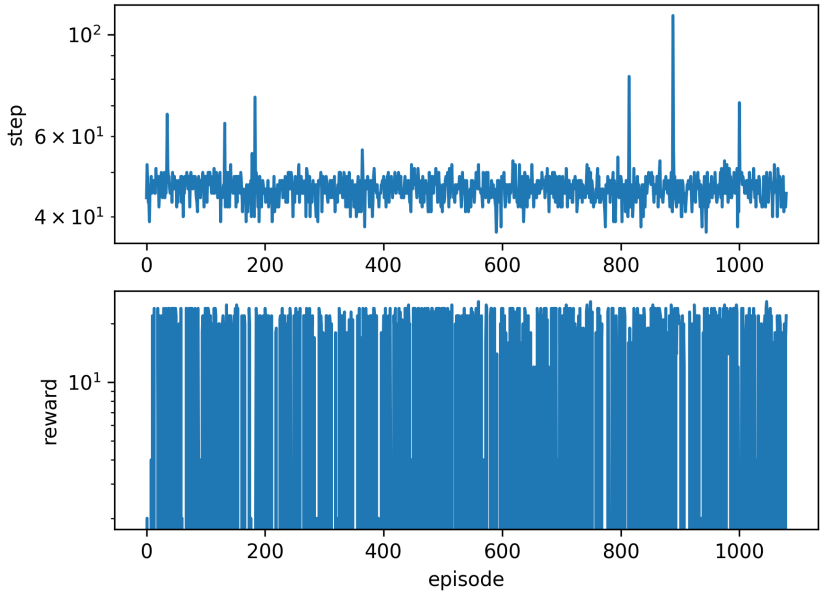

図 4: ステップ数 50000 の際の推移

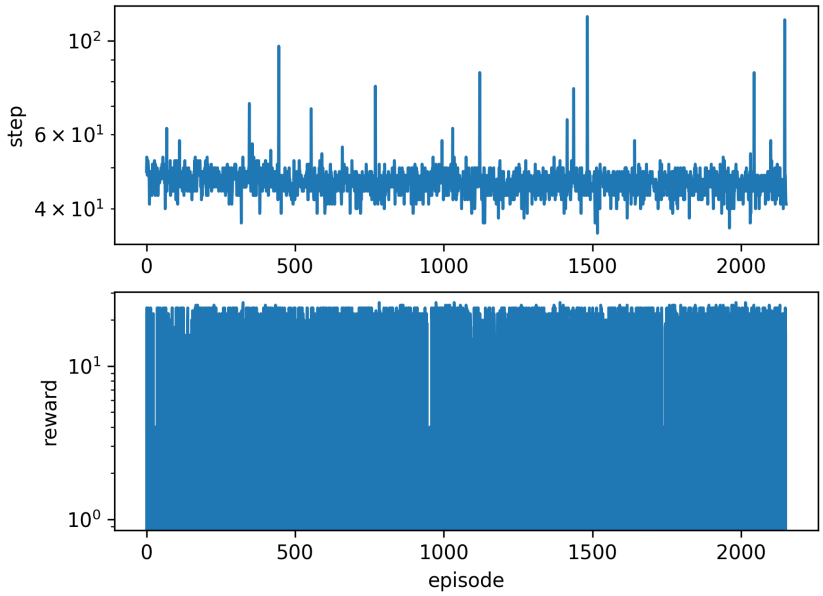

図 5: ステップ数 100000 の際の推移

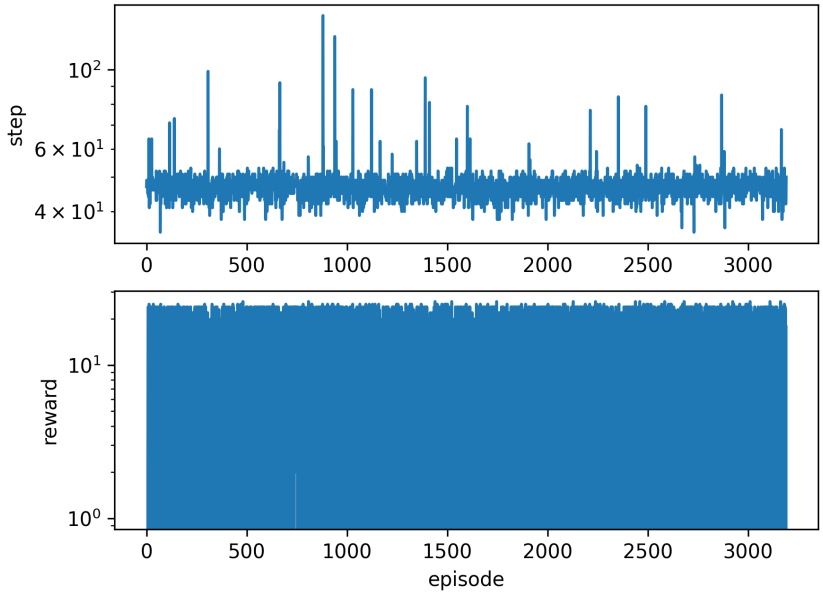

図 6: ステップ数 150000 の際の推移

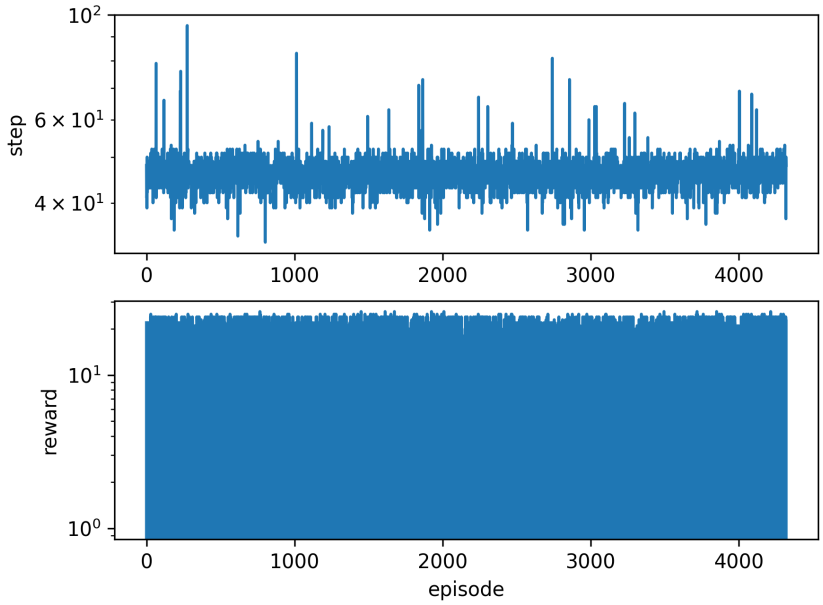

図 7: ステップ数 200000 の際の推移

また, 本資料を書いている途中に気づいたが行動次元数が 35 であるにもかかわらずモデルの全結合層の次元が 16 であったり, デッキにおいてカードのパラメータは先週の実験の数値を使いまわしており HP が低く盤面に残りづらいという問題があるため修正し再実験する必要があると言える.

3.8 今後やること

- 再実験
  今回作成した環境で上手く学習できればやっと環境構築から開放されるので早めに確認・修正する.
- モンテカルロ法の実装
  環境の改良に時間がかかってしまったため先延ばしになってしまった.keras の Agent クラスをオーバーライドしても上手く行かなさそうなので 1 から作ってみる. また,keras-rl を用いると DDQN や DDPG といった深層強化学習手法も簡単に試せそうであったため必要があれば試してみたい [4].
- 自動バランス調整の方法検討知識不足で見当がつかないので調べる.

参考文献

[1] Seth Kitchen. hearthstone-gym, 2018. https://github.com/SethKitchen/hearthstone-gym.

[2] goodclues. Python の強化学習ライブラリ keras-rl のパラメータ設定, 2019.08.19. https://qiita.com/goodclues/items/9b2b618ac5ba4c3be1c5.

[3] Rakus Developer Blog. 「keras」とパラメータ最適化フレームワーク「optuna」を使った 2 値分類モデルの作成, 2022.05.27. https://tech-blog.rakus.co.jp/entry/20220527/keras.

[4] Keras-RL Documention. Overview. https://keras-rl.readthedocs.io/en/latest/agents/overview/.