Vol.2020-GI-43 No.6
2020/3/13

CFR法とQ学習法の格闘ゲーム人工知能への適用

山岡 勇太[1,a)]　保木 邦仁[1]

概要：格闘ゲームの限定的なゲーム状況に対し CFR 法と Q 学習法を適用し，両手法の性能を比較した．格闘ゲームは二人不完全情報ゲームでありビデオゲームである．CFR 法は二人ゼロ和不完全情報ゲームの ε 均衡点を求める手法である．Q 学習法は強化学習法の一種である．結果として，Q 学習法で開発されたプレイヤは搾取されうる戦略を取るが，均衡戦略を取るプレイヤに対しては良好な性能を発揮することがあることがわかった．また，大きさ 16 の集団を形成し，これの Q 学習を行うことで Q 学習法の性能の改善の余地を見い出した．

1. はじめに

ゲームを上手にプレイする人工知能 (AI) を開発するための研究は古くから行われており，近年では機械学習法を用いた AI の開発事例が数多く報告されている． バックギャモンでは 1994 年に，強化学習法の一種である TD(λ) 法と深層学習法を用いて開発された TD-Gammon が，人間の上級者に匹敵する性能を示した [1]． 強化学習と深層学習を用いた強い AI の開発事例としてはこれが初である．囲碁では 2017 年に，AlphaGo が世界トッププレイヤであるカ・ケツに勝利した [2]．AlphaGo は，強化学習法と深層学習法を用いて開発され，さらに豊富な計算資源を活用することで，通常では 100 年以上かかる学習を数日で行った．また同年には，ポーカーの一種であるヘッズアップノーリミットテキサスホールデム (HUNL) で，Deep Stack が初めてプロのプレイヤに勝利し [3]，2018 年には同ゲームで Libratus がプロの世界ランク上位 4 人と対戦し，統計的有意に勝ち越した [4]．Deep Stack は，2 人不完全情報ゲームの ε 均衡点を求めることができる CFR 法 [5] を改良した CFR+ [6] と深層学習法を用いて開発された．Libratus も，CFR+と CFR 法の改良手法の一つである Monte Carlo CFR (MCCFR) [7] を用いて開発された．バックギャモンは状態数が 10^{20} 程度あるのに対し，囲碁は 10^{170} 程度，HUNL は 10^{160} 程度あると言われている．さらに，HUNL は不完全情報性を持つため，囲碁と同等かそれ以上に複雑なゲームと言われている．このように非常に膨大な状態集合を持つ複雑なゲームに対しても，豊富な計算資源を活用した機械学習法を用いることで人間のプロを超えるような AI が開発可能になった．

先に挙げたボードゲーム・カードゲームの他に，ビデオゲームの研究も行われている．2011 年にはマリオに強化学習法の一種である Q 学習法を適用し，高性能な AI を開発する試みが行われた [8]．2015 年には Atari2600 に含まれる 49 のゲームのうち，29 のゲームで人間の上級者と同等かそれ以上の性能を持つ AI が開発された．この AI は Deep Q-Network という，Q 学習と深層学習法を併用した手法を用いて開発された [9]．Deep Q-Network はゲーム画像のみから学習を行うことが可能という点で革新的であり，現在はこれを改良した手法が数多く提案されている．

2 人不完全情報ゲームであり，かつビデオゲームでもあるゲームの一つに格闘ゲームがある． 本研究の目的は, 不完全情報ゲームで成果をあげた CFR 法と，ビデオゲームで成果をあげた強化学習法を格闘ゲームに適用し，ある限定されたゲーム状況において，両手法の性能を比較し評価することである．本研究で使用する格闘ゲームは，立命館大学の知能エンターテインメント研究室 (Intelligent Computer Entertainment Lab.，ICE) が開発した AI 研究用格闘ゲームの FightingICE である [10]．また同研究室は，モンテカルロ木探索を用いて勝率が最大となるように意思決定を行う MctsAi を公開した [11]． 本研究では，これとの性能比較も行う．

2. 基礎知識

本章では，本研究で必要となる基礎知識を紹介する．

2.1 FightingICE

FightingICE とは，2013 年に ICE が開発した AI 研究用格闘ゲームである (図 1 参照)．FightingICE はゲームを進

[1] 電気通信大学
The University of Electro-Communications
[a)] y1411199.uec@gmail.com

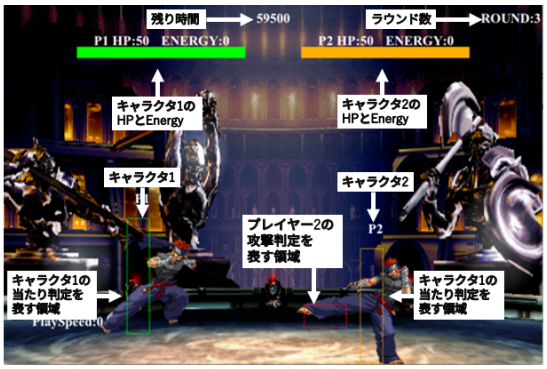

図 1　FightingICE のプレイ中のゲーム画面の一例．2 つの AI それぞれがキャラクタを操り格闘する．

行させる対戦マネージャと AI 2 つで構成される．対戦マネージャは 2 体のキャラクタからなる現在のゲーム状況を AI に送り，AI は送られてきたゲーム状況を元にキャラクタの次の行動を決定し対戦マネージャに送る．対戦マネージャは 2 つの AI から行動を受け取った後にゲーム状況を更新し，再び AI に新たなゲーム状況を送る． これらのようなメッセージの通信を繰り返してゲームが進行する．

初めに，ゲーム 1 回の流れを述べる． 各 AI は使用するキャラクタのタイプを決定する． 使用可能なタイプは ZEN，GARNET，LUD の 3 種である．タイプを決定したのち格闘戦を開始する． 1 戦は 3 ラウンドからなり，1 ラウンドは 60 秒である．3 ラウンド後，勝利したラウンド数が多いプレイヤがゲームの勝者となる． 各ラウンドの勝利条件は，制限時間内に相手の HP が 0 になるか，ラウンド終了時の HP が相手よりも高いかである．ただし，制限時間内に自キャラクタと相手キャラクタの HP が同時に 0 になるか，またはラウンド終了時のお互いのキャラクタの HP が等しい場合は引き分けとなる．

各ラウンド中，プレイヤはフレーム単位でゲーム状況を受け取り，キャラクタの行動を決定する． フレームとはビデオゲームにおける時間の最小単位のことで, FightingICE では 1 秒間は 60 フレームである．

次に，キャラクタの行動について述べる． 行動は相手にダメージを与える攻撃行動と，それ以外の非攻撃行動に分けられる．行動は 1 回のキー入力で実行されるものと，複数のキーを連続で入力することで実行されるものがある．本研究では，複数のキーの連続入力は中断しないものと仮定する．

非攻撃行動には移動行動と防御行動がある．移動行動はキャラクタの位置を変更することができる． 例えば，ジャンプやダッシュやバックステップがある． 移動行動は，相手との距離を調整したり，相手の攻撃を避けるために用いられる．防御行動は相手の攻撃を防御することができる．

攻撃行動は主にパンチやキックなどの通常技と，防御できない投げ技と，ダメージの大きい必殺技の 3 つがある．必殺技は，格闘中に溜まるポイントを消費することで使用することができ，相手が防御している場合でも多少のダメージを与えることができる．また，攻撃行動には当たり判定があり，キャラクタの当たり判定を表す領域と重なると，そのキャラクタにダメージを与えることができる．

最後にキャラクタの体勢について述べる． 体勢には基本体勢と復帰体勢の 2 種がある．基本体勢はキャラクタが行動を選択可能であるときの体勢である． キャラクタが使用可能な行動は体勢に応じて変わる．復帰体勢は，キャラクタが相手の攻撃を受けたときや，地上に着地したときの体勢である． キャラクタが復帰体勢を取っている場合は行動の選択ができない．

2.2　CFR 法と Q 学習法の概要

CFR 法とは，Zinkevich らによって提案された二人ゼロ和不完全情報ゲームの ε 均衡点を求める手法である [5]．ゼロ和ゲームとは，任意の行動戦略の組 $b = (b_1, b_2)$ に対して, プレイヤすべての期待利得の総和 $U_1(b) + U_2(b)$ ($U_i(b)$ は b の下でのプレイヤ i の期待利得）が 0 となるゲームのことである．CFR 法を説明するにあたって, $E_i(x), b_{iv}(e), \pi_i^b(x), \pi_{-i}^b(x), \pi^b(x,y), \pi^b(I), \pi_i^b(I), \pi_{-i}^b(I)$ は以下を表すこととする．

$E_i(x)$： ゲーム木の根から節点 x へのパスに含まれるプレイヤ i の枝すべてからなる集合

$b_{iv}(e)$： プレイヤ i が節点 v で枝 e を選択する確率

$\pi_i^b(x)$： b のもとで節点 x に到達する確率 $\pi^b(x)$ にプレイヤ i が寄与する確率 (すなわち $\pi_i^b(x) = \prod_{(e,v)\in E_i(x)} b_{iv}(e)$)

$\pi_{-i}^b(x)$： プレイヤ i を除くすべてのプレイヤが $\pi^b(x)$ に寄与する確率

$\pi^b(x,y)$： b の下で節点 x から節点 y へ到達する確率

$\pi^b(I)$： 情報集合 I に到達する確率 (すなわち $\pi^b(I) = \sum_{x\in I} \pi^b(x)$)

$\pi_i^b(I)$： $\pi^b(I)$ にプレイヤ i が寄与する確率

$\pi_{-i}^b(I)$： プレイヤ i を除くすべてのプレイヤが $\pi^b(I)$ に寄与する確率

CFR 法は反復計算を行なって average strategy を改善し，近似精度の高い ε 均衡点を求める方法である．反復計算を T 回行ったとする（Algorithm1 参照）． 反復 t 回目におけるプレイヤ i の行動戦略を b_i^t とおく．プレイヤ i の任意の情報集合 $I \in \mathcal{I}_i$ の任意の行動 $a \in A(I)$ に対する average strategy $\bar{b}_i^T(I)(a)$ は次の式で定義される．

$$\bar{b}_i^T(I)(a) = \frac{\sum_{t=1}^T \pi_i^{b^t}(I) b^t(I)(a)}{\sum_{t=1}^T \pi_i^{b^t}(I)} \qquad (1)$$

行動戦略の組 b に従ったときの，手番 x におけるプレイヤ i の期待利得を $u_i^b(x)$ とおく．b に従ったときの，情報集合 $I \in \mathcal{I}_i$ におけるプレイヤ i の counterfactual utility $u_i^b(I)$ は次の式で定義される．

$$u_i^b(I) = \frac{\sum_{x \in I, w \in W} \pi_{-i}^b(x)\pi^b(x,w)u_i(w)}{\pi_{-i}^b(I)} \quad (2)$$

この式は，プレイヤ i が情報集合 I を経由しようとした場合の，I における期待利得を表している．

$R_i^T(I,a), R_i^{T,+}(I,a)$ は次の式で定義される．

$$R_i^T(I,a) = \frac{1}{T}\sum_{t=1}^{T} \pi_{-i}^{b^t}(I)\left(u_i^{b^t|_{I\to a}}(I) - u_i^{b^t}(I)\right) \quad (3)$$

$$R_i^{T,+}(I,a) = \max\left(R_i^{T,+}(I,a), 0\right) \quad (4)$$

反復 $T+1$ 回目の行動戦略 $b_i^{T+1}(I)(a)$ は次の式で定義される．

$$b_i^{T+1}(I)(a) = \frac{R_i^{T,+}(I,a)}{\sum_{a' \in A(I)} R_i^{T,+}(I,a')} \quad (5)$$

ただし，$\sum_{a' \in A(I)} R_i^{T,+}(I,a') = 0$ ならば $b_i^{T+1}(I)(a) = 1/|A(I)|$ とする． また, $R_i^0(I,a) = 0$ とする．

Algorithm 1 CFR アルゴリズム

1: $\forall i \in N, \forall I \in \mathcal{I}_i, \forall a \in A(I), R_i^0(I,a) \leftarrow 0$
2: **for** $t = 1$ to T **do**
3: $\quad \forall i \in N, b_i^t \leftarrow$ (R_i^{t-1} を元にプレイヤすべての行動戦略を式 (5) を用いて計算)
4: $\quad b^t$ の下での各 $R_i^t(I)(a)$ を木の頂点から初期点の方向に順番に計算
5: **end for**
6: $b^1 \sim b^T$ を元に $\bar{b}^T$ を計算
7: **return** $\bar{b}^T$

搾取量とは，戦略の組の性能を評価する指標の一つである [12]．$b = (b_1, b_2)$ をプレイヤすべての戦略の組とする．このとき搾取量 x は以下の式で計算される．

$$x = \frac{\max_{b_1'} U_1(b_1', b_2) + \max_{b_2'} U_2(b_1, b_2')}{2} \quad (6)$$

また，b は $2x$ 均衡点であることが知られている．

Q 学習法は Watkins によって提案された強化学習法の一種である [13] [14]．Q 学習法にはテーブル型の Q 学習法と関数近似を用いた Q 学習法がある． 関数近似を用いた Q 学習法では，行動価値関数をパラメータ $\boldsymbol{\theta}$ で表現された関数 $\hat{Q}(\cdot|\boldsymbol{\theta})$ で近似する．あるマルコフ決定過程があるエピソードを生成して，状態 s で挙動方策 π によって行動 a を選択し，状態 s' に遷移して報酬 r を得たとする． このとき，目標値 $r + \max_{a' \in A(s')} \hat{Q}(s', a' \mid \boldsymbol{\theta})$ に $\hat{Q}(s, a|\boldsymbol{\theta})$ を近づけるように $\boldsymbol{\theta}$ を更新する．

experience replay とは，エピソードに含まれる状態 s，行動 a，次状態 s'，報酬 r' の組を直近 D 個だけデータ集合（リプレイバッファ）に保存しておき，行動価値を更新する際にリプレイバッファの中からランダムに組を取得してミニバッチを構成する手法である．

target network とは，挙動方策 π が ε グリーディの場合において，ニューラルネットワーク（NN）の重み $\boldsymbol{\theta}$ の更新 i 回目のとき，$\max_{a' \in A(s')} \hat{Q}(s', a' \mid \boldsymbol{\theta})$ の計算に $\lfloor i/C \rfloor \times C$（$C$ は正の整数）回目の重み $\boldsymbol{\theta}'$ を用いる手法である [9]．

3. 研究目的

本研究の目的は，性質が異なる二つの手法（CFR 法と Q 学習法，表 1 参照）を格闘ゲームに適用し，両手法の性能を比較することである．著者らの知る限りにおいて，これらの手法の性能を格闘ゲームにおいて比較した研究は報告されていない．

本研究では，FightingICE などの格闘ゲームのゲーム木は非常に大きいため，CFR 法を適用することは困難であることから，限定的なゲーム状況のみを考える．また，FightingICE はリアルタイムでゲームが進行し，ゲーム木の点すべてを訪問することは困難であることから，FightingICE の詳細なルールを調査し，その結果を元にエミュレータを実装する．さらに，Q 学習法の行動価値関数は NN で近似して表す．性能の比較は，CFR 法と Q 学習法に加え，既存 AI の MctsAi も含めて行う．性能の指標は，搾取量と勝率とする． 性能の比較はまた，ディレイ 0 と 15 の場合で行う．ディレイとは，AI が受け取るゲーム状況の 15 フレーム分の遅延のことである[*1]．これは，FightingICE がゲーム状況に人間がもつような制約を課すための仕組みである．

4. ゲームルールの調査とエミュレータの実装および実験

本章では，ゲームルールの調査方法とその結果について述べた後，エミュレータの詳細と動作検証の結果を示す．

ゲームルールの詳細を調査するにあたって，FightingICE の公式サイト[*1]で公開されているルールに関する情報 (PDF の文書並びに CSV 形式の表) や GitHub で公開されている FightingICE のソースコードを参照した[*2]．調査した結果を付録に示す．

調査したゲームルールに基づきエミュレータを実装した． エミュレータの仕様は以下の通りである．

- 3 種のキャラクタのうち ZEN のみ使用可能．
- リアルタイム性を排除し，対戦マネージャは AI の意思決定が終了するまで待つ．
- ディレイやゲームの制限時間，キャラクタのゲーム開始時の HP，Energy，位置などのパラメータを変更可．
- ゲーム画面の描画なし．

実装したエミュレータの動作確認を行う． エミュレー

[*1] Fighting Game AI Competition : http://www.ice.ci.ritsumei.ac.jp/ ftgaic/index-2h.html (last access: Jan 26, 2020)
[*2] TeamFightingICE/FightingICE - GitHub : https://github.com/TeamFightingICE/FightingICE (last access: Jan 26, 2020)

表 1　CFR 法と Q 学習法のメリットとデメリット

	メリット	デメリット
CFR 法	二人ゼロ和不完全情報ゲームの ε 均衡点を求めることができる	ゲーム木が大きいと適用が困難
Q 学習法	行動価値関数を関数近似することで膨大なゲーム木を持つゲームに適用可能	関数近似を行うと行動価値が収束する保証がない 不完全情報ゲームを適切に扱うことができない

タの動作は，FightingICE の既存 AI 3 体をエミュレータで動作するように実装し，各 AI の，FightingICE での勝率とエミュレータでの勝率が同程度になるかどうかで確認する．実験に使用する既存 AI は以下の 3 つである．

- ランダムに行動を決定するランダムプレイヤ
- モンテカルロ木探索で行動を決定する MctsAi
- IF-THEN ルールで行動を決定する Machete

どの AI のペアも 200 ラウンド対戦する (左右それぞれ 100 ラウンド)． ゲームパラメータの設定は，Fighting Game AI Competition の Standard League のルールに従う．また，勝率を計算するにあたり，勝ちを 1，負けを 0，引き分けを 0．5 とする．95%信頼区間は標準誤差から見積もった．

ランダムプレイヤは，FightingICE とエミュレータ両方において，他 AI 2 つに殆ど勝てなかった．また，Machete の MctsAi に対する勝率は 0.86 ± 0.10 (FightingICE) と 0.77 ± 0.11 (エミュレータ) であり，Machete が MctsAi に勝ちやすいという傾向はエミュレータ上でも再現されていた．ただし，FightingICE においては，MctsAi の意思決定はフレーム落ちによりしばしば無視されてしまうため，Machete の勝率が多少高くなるようである．さらに，思考時間が短いランダムプレイヤと Machete の対戦実験の計算は，エミュレータの方が約 16 倍速かった．

5. CFR 法と Q 学習法の実装および実験

本章では，CFR 法を用いた実験と Q 学習を用いた実験の共通設定を述べたあと，両実験の詳細と結果を示す．

5.1 実験の共通設定

本研究では以下のようにゲーム状況を抽象化する．

- 開始時の各キャラクタの HP は 2
- 開始時の各キャラクタの Energy は 150
- 開始時のキャラクタ間の距離が 4px
- 開始時の残り時間 1 秒
- 開始時の体勢は STAND
- 4 フレームごとに行動の決定が可能

この状況は，待機し続ける相手に対して任意の攻撃行動が届く距離であり，任意の攻撃行動を行うことができ，どの攻撃行動が当たっても負けてしまう状況である．これらに加えて，キャラクタの行動集合は以下の 2 通りを考える．

行動集合 1　攻撃行動すべてからなる集合

行動集合 2　投げ技弱（THROW_A），必殺技強の一種（STAND_D_DB_BB），防御の一種（STAND_GUARD）からなる集合

表 2　4 つのゲーム木の大きさ．「D」はディレイ．

	行動集合 1		行動集合 2	
	D0	D15	D0	D15
情報集合	74	47	86	24
枝	1766	1320	348	276
内部節点	106	78	116	92
葉節点	1661	1243	233	185
葉の経路の最大長	4	4	30	24

ただし，投げ技弱，必殺技強，防御については付録の表 A.1, A.2 を参照されたい．先のゲーム状況において行動集合 1 の場合は，純戦略の組 (必殺技強，必殺技強) が均衡点である．行動集合 2 の場合は，純戦略上の均衡点はありそうにない． もしディレイがなければ，行動集合 2 の各行動には以下のような関係がある．

- 投げ技弱は防御に勝てるが，必殺技強には負ける．
- 防御は必殺技強に勝てるが，投げ技弱には負ける．
- 必殺技強は投げ技弱には勝てるが，防御には負ける．

ただし，防御が必殺技強に勝つためには，防御を選択した直後の意思決定において，必殺技強を選択する必要がある．必殺技強を選択しない場合は負けとなる． つまり，行動集合 2 の各行動はジャンケンのグー・チョキ・パーのようなものである．

5.2 CFR 法を用いた実験

表 2 に，2 種の行動集合とディレイの下で，情報集合，木の枝，木の内部節点および木の葉節点数を示す．ここで，内部節点は選択可能な行動の数が 2 つ以上存在する意思決定点とした．

図 2 は，各ゲーム状況に対し CFR 法を適用したときの搾取量の変化を示す．図 2 では，行動集合 1 のディレイ 0 とディレイ 15 の場合の曲線がほぼ重なっている．この図より，戦略の組が徐々に均衡点に近づいていくことがわかる．また，行動集合 1 の場合の方が搾取量の減少が早いことがわかる． これは，純戦略均衡点が存在することに起因すると考えられる．

CFR 法で 10^4 回反復を行うのに，およそ 3，4 日程度の時間を要した． また，CFR 法を適用するためには，各情報集合での行動集合の大きさが全て同じだと仮定した場合，情報集合の数 × 行動集合の大きさ × プレイヤの数 ×4 程度

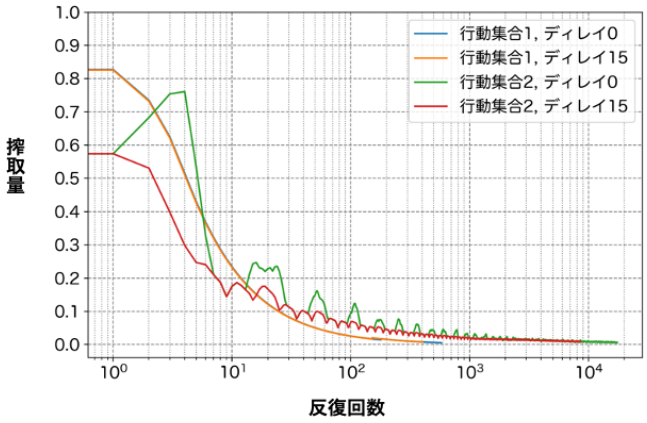

図 2　各ゲーム状況での搾取量の変化

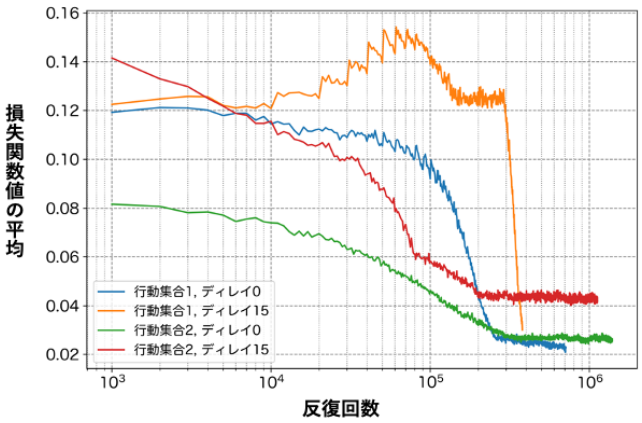

図 3　各ゲーム状況での損失関数値の平均の変化

表 3　NN の構成

	入力の数	出力の数	ReLU	重みの数
入力	-	163	-	0
中間層 1	163	128	✓	20992
中間層 2	128	128	✓	16512
中間層 3	128	128	✓	16512
出力	128	41	-	5289

表 4　キャラクタの特徴．当り判定は box は左右の x 座標および上下の y 座標で表現

ユニット数	特徴
1	HP
1	Energy
1	向いている方向
4	キャラクタの当り判定 box
1	x 方向の移動速度
1	y 方向の移動速度
41	効果発動中の行動
4	拳・蹴りの当り判定 box
15	態勢
12	エネルギー弾の当り判定 box

の浮動小数点数を保持する必要がある．CFR 法の実験は，Intel Xeon(R) CPU E31275 のコア 1 つを使って行った．

5.3　Q 学習法を用いた実験

挙動方策は ε グリーディ方策，推定方策はグリーディ方策とする．学習の際は，ε グリーディ方策のランダム行動の割合を，学習開始から 10^5 フレーム経過するまでの間に 1.0 から 0.1 に単調減少させる．10^5 フレーム経過後は 0.1 で固定する．また，[9] の experience replay と target network を導入する．NN は表 2 のゲーム木の内部節点すべての行動価値関数を表現するのに十分な数の重みを持つ（表 3 参照）．NN の入力には，各キャラクタのパラメータ (162 ユニット，表 4 参照) とラウンド開始からの経過フレーム数 (1 ユニット) を用いる．学習における報酬は，勝てば 1，負ければ −1，それ以外は 0 とする．また，損失関数には二乗誤差を，重みの最適化手法には確率的勾配降下法を用いる．学習に関するパラメータを表 5 に示す．ここで，リプレイサイズは experience replay のパラメータであり，状態，行動，報酬，次状態の組を保持する数を表す．対戦相手にはランダムプレイヤを用い，1 つのラウンドから 1 つのエピソードを採取して，30 万エピソード分の学習を行う．

表 5　学習のパラメータ

パラメータ名	値
学習率	5.0×10^{-6}
モメンタム	0.9
リプレイサイズ	10^5
ミニバッチサイズ	32
ターゲットネットワークの更新頻度 C	10^4

図 3 に，損失関数値の平均の変化を示す．ここで反復とは，ミニバッチを用いた重みの更新処理 1 回を指す．縦軸は反復回数 1000 回にわたる損失関数値の平均を表す．

反復を 10^6 回行うのに，およそ 1 時間を要した．また，Q 学習法を行う場合，ネットワークの重みの数だけ浮動小数点数を保持すれば良い．Q 学習法の実験は Intel Core i7 2.5GHz で行った．Q 学習法と MctsAi は，期待勝率が最大となるように意思決定を行う点で類似している．しかし，MctsAi は 1 回の意思決定に 16 ミリ秒を要するが，Q 学習法は，本研究のネットワークの規模であれば 1 ミリ秒以内で意思決定を行うことができる．

5.4　性能比較

まず，各ゲーム状況に対する CFR 法と Q 学習法の搾取量を計算した (表 6)．ただし，Q 学習プレイヤは ε グリーディ方策に基づいて行動するものとし，$\varepsilon = 0.05$ とする．純戦略の組の均衡点が存在する場合は，両手法とも搾取されにくい（0 に近い）行動戦略を与えていることがわかった．混合戦略が必要な場合は，Q 学習プレイヤは搾取されうることがわかった．また表には書いていないが，MctsAi の搾取量も，ほとんど決定論的な意思決定を行うため，1

表 6　各ゲーム状況におけるプレイヤに対する搾取量．「D」はディレイ．最前で 0，最悪で 1．

	行動集合 1		行動集合 2	
	D0	D15	D0	D15
CFR 法	0.006	0.008	0.007	0.009
Q 学習法	0.044	0.044	0.966	0.950

表 7　Q 学習プレイヤの勝率（%）．「D」はディレイ．

	行動集合 1		行動集合 2	
	D0	D15	D0	D15
対 CFR	48 ± 1	49 ± 1	50 ± 4	51 ± 4
対 MctsAi	64 ± 3	64 ± 3	40 ± 3	48 ± 3

付近になることがわかった．

次に，CFR 法と Q 学習法で得られた行動戦略を勝率によって比較した．これらのプレイヤを用いて左右 1000 ラウンドずつ対戦を行い勝率を計算した．ただし，Q 学習プレイヤは ε グリーディ方策に基づいて行動するものとし，$\varepsilon = 0.05$ とする．勝ちを 1，負けを 0，引き分けを 0.5 としたときの，各ゲーム状況における Q 学習プレイヤの CFR プレイヤと MctsAi に対する勝率を推定した (表 7)．
純戦略の組の均衡点が存在する場合，両手法の勝率は 5 割程度であった．混合戦略が必要な場合，Q 学習プレイヤは搾取されうる戦略を取っているにもかかわらず，均衡戦略をとるプレイヤに対し勝率が 5 割程度であった．また，Q 学習プレイヤは MctsAi とおおよそ同等の性能を示した．

5.5　プレイヤープールを用いた実験

前項では Q 学習プレイヤと MctsAi の搾取量がほとんど 1 であることを確認した．そこで本研究では，Multi-Agent Reinforcement Learning [15] のような方法で Q 学習プレイヤの性能を改善することを試みる．本研究ではまず，とりあえずの感触を掴むために，以下に示す簡潔な手順でプレイヤを開発する．

1　リプレイバッファと重みの履歴の組を N 個用意（リプレイバッファは空であり，重みの履歴はランダムに初期化された重み 1 個とする）
2　以下繰り返し
　2-1　N 個の組から重複を許さずランダムに 2 組選択し，プレイヤ 1 とプレイヤ 2 を構成
　2-2　ラウンドを最新の重みに基づく挙動方策で 1 つ生成
　2-3　各プレイヤのリプレイバッファを更新
　2-4　リプレイバッファが溜まっていれば，各プレイヤの最新の重みをそれぞれ一定回数更新し，重みの履歴を更新

この手順で開発されたプレイヤを複合プレイヤと呼ぶ．複合プレイヤの意思決定方法は以下の通りである．

表 8　各ゲーム状況における複合プレイヤに対する搾取量（最善で 1，最悪で 0）と複合プレイヤの CFR プレイヤに対する勝率（%）

	行動集合 2	
	D0	D15
搾取量	0.352	0.476
対 CFR の勝率	40 ± 4	43 ± 4

1　ランダムに組を 1 つ選択
2　その組の最新の重みのグリーディ方策で意思決定

先の 4 つのゲーム状況のうち，行動集合 2 の場合における複合プレイヤの搾取量と複合プレイヤの CFR プレイヤに対する勝率を計算した（表 8）．勝率の計算方法は前項と同様である．ただし $N = 16$ とし，挙動方策には ε グリーディを用いた．その他の Q 学習法に関するパラメータは第 5.3 項のものと同じである．

結果として，Q 学習プレイヤに対する搾取量は大幅に改善された．その一方で，均衡戦略を取るプレイヤに対する勝率は少し小さくなってしまった．

6.　おわりに

本研究では，不完全情報ゲームでありビデオゲームである格闘ゲームの限定的な状況を考えて，CFR 法と Q 学習法を適用し，それぞれの手法の性能を調査した．[9] で用いられていた experience replay と target network を導入し，Q 学習法の行動価値関数は NN で関数近似した． また，CFR 法を適用するために，ゲームルールの詳細を調査し，エミュレータを実装した．結果として，Q 学習法は決定論的なプレイヤに対して搾取されうることと，均衡戦略を取るプレイヤに対しては，良好な性能を発揮することがあることが確認できた．また，学習の反復を 10^6 回行うのに要した時間はおよそ 1 時間であった． 学習を行うためにはリプレイバッファと NN の重みを 2 セットだけ保持すれば良いため，メモリ消費量も少なかった．これに対し CFR 法は，本研究で扱ったような小さいゲーム状況に対しても，10^4 回の反復で 3,4 日を要した．ゲーム木が大きくなるにつれて，計算時間や必要なメモリが膨大に増えてしまうため，よりオリジナルのルールに近いゲーム状況に対する適用は困難だということがわかった．

さらに，本研究では Q 学習プレイヤの性能の改善を試みた．結果として，Q 学習プレイヤに対する搾取量は大幅に改善された．その一方で，均衡戦略を取るプレイヤに対する勝率はやや悪化した．

参考文献

[1]　Tesauro, G.: Temporal difference learning and TD-Gammon, *Communications of the ACM*, Vol. 38, No. 3, pp. 58–68 (1995).
[2]　Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou,

表 **A.1** ZEN の非攻撃行動

行動名	行動の効果が終了するフレーム数	MotionLevel	他行動が選択可能になるまでのフレーム数
FORWARD_WALK	5	10	0
DASH	12	10	2
JUMP	24	4	4
FOR_JUMP	22	4	4
BACK_JUMP	22	4	4
BACK_STEP	25	2	25
STAND_GUARD	23	10	0
CROUCH_GUARD	23	10	0
AIR_GUARD	23	10	0
NEUTRAL	0	10	0

I., Panneershelvam, V., Lanctot, M. et al.: Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol. 529, No. 7587, pp. 484–489 (2016).
[3] Moravčík, M., Schmid, M., Burch, N., Lisỳ, V., Morrill, D., Bard, N., Davis, T., Waugh, K., Johanson, M. and Bowling, M.: Deepstack: Expert-level artificial intelligence in heads-up no-limit poker, *Science*, Vol. 356, No. 6337, pp. 508–513 (2017).
[4] Brown, N. and Sandholm, T.: Superhuman AI for heads-up no-limit poker: Libratus beats top professionals, *Science*, Vol. 359, No. 6374, pp. 418–424 (2018).
[5] Zinkevich, M., Johanson, M., Bowling, M. and Piccione, C.: Regret minimization in games with incomplete information, *Advances in neural information processing systems*, pp. 1729–1736 (2008).
[6] Tammelin, O.: Solving large imperfect information games using CFR+, *arXiv preprint arXiv:1407.5042* (2014).
[7] Lanctot, M., Waugh, K., Zinkevich, M. and Bowling, M.: Monte Carlo sampling for regret minimization in extensive games, *Advances in neural information processing systems*, pp. 1078–1086 (2009).
[8] Liao, Y., Yi, K. and Yang, Z.: Cs229 final report reinforcement learning to play mario, Technical report, Technical report, Stanford University (2012).
[9] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G. et al.: Human-level control through deep reinforcement learning, *Nature*, Vol. 518, No. 7540, pp. 529–533 (2015).
[10] Lu, F., Yamamoto, K., Nomura, L. H., Mizuno, S., Lee, Y. and Thawonmas, R.: Fighting game artificial intelligence competition platform, *2013 IEEE 2nd Global Conference on Consumer Electronics (GCCE)*, IEEE, pp. 320–323 (2013).
[11] Yoshida, S., Ishihara, M., Miyazaki, T., Nakagawa, Y., Harada, T. and Thawonmas, R.: Application of Monte-Carlo tree search in a fighting game AI, *2016 IEEE 5th Global Conference on Consumer Electronics*, IEEE, pp. 1–2 (2016).
[12] Johanson, M., Waugh, K., Bowling, M. and Zinkevich, M.: Accelerating best response calculation in large extensive games, *Twenty-second international joint conference on artificial intelligence* (2011).
[13] Watkins, C. J. C. H.: Learning from delayed rewards, Ph.D. Thesis, Cambridge (1989).
[14] Sutton, R. S. and Barto, A. G.: 強化学習，森北出版 (2000).
[15] Zhang, K., Yang, Z. and Başar, T.: Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms, *arXiv preprint arXiv:1911.10635* (2019).
[16] 山岡勇太：CFR 法と強化学習法の格闘ゲーム人工知能への適用，修士論文，電気通信大学大学院 (2020).

付　　録

キャラクタ 3 体のうち，ZEN の非攻撃行動と攻撃行動の詳細を表 A.1, A.2 に示す．MotionLevel とは，その行動のキャンセルしやすさを表す指標である．ダメージとは，攻撃が相手に当たった際の，相手の HP の減少量を表す．Startup とは，攻撃を選択してから攻撃判定が発生するまでのフレーム数を表す．Active とは，攻撃判定が発生するフレーム数を表す．Recovery とは，攻撃判定がなくなり，他行動の選択が可能になるまでのフレーム数を表す．IsDOWN とは，攻撃が相手に当たった際に，相手の態勢を DOWN にできるかどうかを表す．詳しくは [16] を参照されたい．

表 **A.2** ZEN の地上 (上部) と空中 (下部) における攻撃行動

行動名	ダメージ	StartUp	Active	Recovery	使用 Energy	取得 Energy	AttackType	IsDOWN	MotionLevel	エネルギー弾の出現フレーム数
THROW_A	10	5	1	24	5	2	throw	×	10	0
THROW_B	20	20	1	9	20	10	throw	×	10	0
STAND_A	5	4	3	11	0	2	high	×	7	0
STAND_B	10	5	3	15	0	5	high	×	7	0
CROUCH_A	5	3	3	12	0	3	low	×	7	0
CROUCH_B	10	11	3	6	0	5	low	×	7	0
STAND_FA	5	5	3	28	0	2	high	×	6	0
STAND_FB	12	12	3	28	0	10	middle	×	6	0
CROUCH_FA	5	4	3	31	0	2	low	×	6	0
CROUCH_FB	10	12	3	45	0	5	low	○	6	0
STAND_D_DF_FA	10	20	0	40	2	3	high	×	3	150
STAND_D_DF_FB	30	15	0	45	30	10	high	×	3	100
STAND_D_DF_FC	120	15	0	33	150	30	high	○	10	100
STAND_F_D_DFA	10	10	20	40	0	5	high	○	3	0
STAND_F_D_DFB	40	10	10	40	50	15	low	○	3	0
STAND_D_DB_BA	10	31	5	21	0	5	middle	×	3	0
STAND_D_DB_BB	25	5	10	45	50	15	middle	○	3	0
AIR_A	8	4	4	6	0	5	middle	×	7	0
AIR_B	10	6	9	8	0	10	middle	×	7	0
AIR_DA	8	5	10	33	5	5	middle	×	6	0
AIR_DB	10	15	10	25	5	10	middle	×	6	0
AIR_FA	8	11	15	16	0	5	middle	×	6	0
AIR_FB	10	7	13	15	0	10	middle	×	6	0
AIR_UA	10	5	10	33	0	5	middle	×	10	0
AIR_UB	20	8	10	32	0	10	middle	×	10	0
AIR_D_DF_FA	10	20	0	40	0	0	middle	×	3	150
AIR_D_DF_FB	30	20	0	40	50	15	middle	×	3	100
AIR_F_D_DFA	10	5	15	40	10	5	middle	×	3	0
AIR_F_D_DFB	20	14	8	38	40	15	middle	×	3	0
AIR_D_DB_BA	10	10	15	35	10	5	middle	×	3	0
AIR_D_DB_BB	40	14	15	41	50	15	middle	○	3	0