行動選択に対するゲーム内変化を利用したゲーム AI の学習
Game AI Using Changes Caused by Selected Actions

佐々木　治人
Haruto Sasaki
法政大学情報科学部ディジタルメディア学科
E-mail: haruto.sasaki.5r@cis.k.hosei.ac.jp

Abstract

This paper addresses the low learning efficiency of Q learning due to its enormous learning time. We propose a method for regressing the optimal actions by predicting the Q parameters of whole actions to improve its learning efficiency by multiple regression analysis. We use a 2D fighting game-type platform called FightingICE as an experiment environment. The experiment of the proposed method shows that our method sometimes surpasses the rule-based AI that the conventional machine learning-based method could not surpass. This may be because the method has been able to predict the action, which has the larger Q parameter, not only repeating the action that Q parameter has recorded. On the other hand, the result shows that although in some cases it surpasses the conventional method, the overall score was lower. This result indicates that our method failed to predict the accurate Q parameters, and is less effective than the conventional method. This may be because the proposed method is excessively affected by the multiple regression analysis, and the result calculated from this could be greatly changed by the recorded Q parameters. Especially, the correlation coefficient largely fluctuates with changes in Q parameters , and the extra data initializes Q parameters with zeros for all options.

1. はじめに

近年，人間や機械の生成するデータは膨大に増加する傾向にあり，それらを基に複雑な判断を行うことは人間の能力を凌駕している．そこで組み合わせや順列といった計算をより効率的に行い最良の判断を下すため，コンピュータに人間の脳を模倣させる人工知能(AI)という技術が開発された．その中でロボティクスやゲーム AI に頻繁に活用される強化学習には sarsa 法，モンテカルロ法等が存在するが，現在の主流は Q 学習である．Q 学習の基本的な手法は，特定の状態において機械が取り得るすべての行動の選択肢の中から一つ選び，実際に行動に移した結果どれだけ状況が良くなった，あるいは悪くなったかを得られた報酬から数値(Q 値)として学習するというものであり，最終的に学習したデータを基に同じ状態に対して得られる報酬が最大になるような選択をとることを目的としており，ゲームソフトウェアではデバッグ作業の自動化に活用されている．しかしこれまで用いられてきた手法では，特定の状態において一度に一つの行動の選択に対する結果の学習しかできないという弱点がある．そのため，十分な量だけ Q 値を記録し，学習を完了するのに膨大な時間が必要になる．

本研究では，対戦型 2D アクションゲームのプラットフォーム FightingICE を用いて，これまでの Q 学習手法の弱点である，膨大な学習量による効率の低さを解消することを目的とし，従来の Q 学習で得られるレベルの最適性を確保した行動を選択できることを目標とする．効率の低さを解消するために，全ての行動の Q 値を予測することで最適だと考えられる行動を優先的に学習する手法を開発する．実験では，2018 年に開発された AI である Thunder と，FightingICE から提供されている AI である MctsAI を比較対象とし，3 ラウンド 100 回戦行ったときの体力差を記録する．実験の結果，対 Thunder ではごくまれに勝利することがあり，対 MctsAI での勝率は 30%程度に留まった．

2. 関連研究

Q 学習の基本的な理念は，Christopher らによって 1992 年に提唱された [1]．機械に特定の状況を再現した環境を与え，その環境の中のある状態において取り得る行動を一つ選択した結果，得られた報酬を具体的な数値として一定の割合で保存する．保存した値は Q 値と呼ばれ，再び同じ状態に直面したときに行動と Q 値のペアから Q 値が最大の行動を実際の行動として出力する．この手法の問題として，Q 値は実際に行動を選択するまで値が 0 であることが挙げられる [1]．ある行動において得られた報酬がどれほど小さくとも，0 より大きければ最大となるため，その行動を繰り返す．つまり，その状態においてより大きな報酬を得られる行動があったとしても値を計測できていないため，実際に値を計測できた行動を優先する．

より大きな報酬を探索する方法として，設定した確率で Q 値が最大でない行動を選択する手法がある．しかし完全な無作為によって選択されるため，偶然最適な行動が選ばれるまで学習させる必要があり，選択できる行動が増えるとその分時間が大きく掛かる．Mnih らの研究では複雑な環境であるテレビゲームの一つ，Atari 内で AI を学習させることに成功している [2]．しかし，この技術は最適な行動の学習自体は従来の技術と同じであり，一部

Supervisor: Prof. Hiroshi Hosobe

のゲームでは入力の複雑さから学習が上手くいっていないことが指摘されている．従来の Q 学習手法に対し，一部のパラメータに対して結果が最適となるパラメータを予測するという手法を開発した [3]．しかし，この手法は一つの行動を選択した後に最適なパラメータを求めるものであり，行動の選択は従来の技術と同様である．

近年，従来の Q 学習に加え，モンテカルロ木探索(MCTS)を応用し，囲碁に用いる AI の開発が行われている [4]．この AI は現在の状態だけでなく，選択を行った後に遷移する先の状態で得られる報酬も計算する．MctsAI は FightingICE 上でモンテカルロ木探索を行う AI である [5]．これは FightingICE に状態の遷移を計算する機能が備わっていることを利用して探索を実現している．

3. 準備

3.1. Q 学習

Q学習では状態sにおいて行動aの行動する価値(Q値)を計算し，最大の価値を持つ行動を分析することで学習を行う．Q値の計算は通常，以下の式で行われる．

$$Q(s,a) \leftarrow Q(s,a) + \alpha\big(R + \gamma \max \mathrm{Q}(s',a') - Q(s,a)\big)$$

ここで，αは学習率，Rは報酬，γは割引率であり，$\max \mathrm{Q}(s',a')$は直後の状態に記録された Q 値の最大を示す．前述のとおり，Q 学習では上記の式によって計算された Q値から，最大の Q値を持つ行動aを選択するのみに留まっており，Q値を予測する機能はない．

3.2. Masson らの手法

Masson らの研究では最適なパラメータを予測する機能を追加するため，図 1 (a)に示すような従来の Q 学習手法に対し，図 1 (b)に示すような行動部分の記録を動作とパラメータに分離して記録する構造になっている．この構造により，次の最適な行動を予測するにあたり，図 1 (b)のような行動についてあらかじめ学習したデータから最適な動作を抽出し，次にその動作について最適なパラメータを予測する．つまり，動作は過去のデータからの抽出にとどまっている．このことより，最適な行動を出力する効率が落ちていると考えられる．

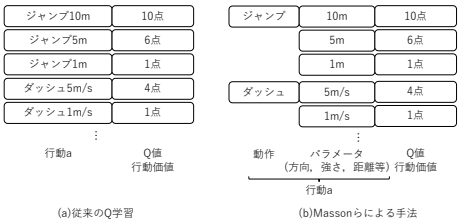

図 1　行動と Q 値の保存方法

Q 値の計算部分は従来の Q 学習と同様であるが，図 1 (b)のように動作とパラメータを分離して記録し，Q 値はパラメータに対応する形式で保存している．パラメータと Q 値の組から Peters らによって提唱された勾配探索法によって，単一のパラメータ変数から Q 値を計算する [6]．この手法によって Q 学習の学習時間の短縮が実現されていると考えられる．

3.3. FightingICE

FightingICE は，対戦型 2D アクションゲームの形式を持つ AI 研究に用いられるプラットフォームである．プレイヤーや AI は画面内のどちらか片方のキャラクターを上下左右と A，B，C の七つのボタンの入力によって操作し，対戦相手の体力をより多く減らした側の勝利となる．このプラットフォームの特徴として，自身や対戦相手の体力や座標の取得が可能なほか，「下，右下，右と A ボタン同時押し」など時間的に連続な入力を必要とする行動を直接指定することができる．また，それぞれの行動を行ったときに発生する速度や攻撃の当たり判定座標を取得することが可能である．

これまでの FightingICE の AI 研究では，特定の状態において行う行動を一意的に設定し，よい対戦結果が得られるかどうか実験するという研究手法が一般的であった．Thunder は，2018 年に開発された FightingICE プラットフォーム用の AI である．これは状態に対する選択を人力で一意に設定したものであり，自身で最適な選択を学習させるものではない．すなわち，チャットボットのような「人工無能」に近い．

FightingICE における研究で学習機構を備えた AI として，MctsAI [5]がある．これはモンテカルロ木探索法を用い，現在の状態から遷移する状態をシミュレーションすることで最適な行動を選択する AI である．しかしこの AI を運用するには，自身や対戦相手の状態，取り得る行動等を手動で設定する必要がある．

4. 条件設定

提案手法を FightingICE に適用するため，以下の条件を設定する．

4.1. 状態の区分

FightingICE では自分や対戦相手の X，Y 座標，体力や現在行っている行動を取得することができ，本研究で開発する AI を運用するに当たってこれらのデータを用いて現在の状態を区分する．状態の区分には自身の X 座標，X 軸速度，Y 軸速度，対戦相手の X 軸距離，Y 軸距離，X 軸速度，Y 軸速度の 7 項目を用いる．

4.2. 報酬の設定

時間tで獲得した報酬r_tは以下の式で表わされる．

$$r_t = r_t^{\mathrm{dis}} + 10r_t^{\mathrm{dmg}} - 10r_t^{\mathrm{selfdmg}} - 10$$

r_t^{dis}は時間tにおいて対戦相手に近づいた距離，r_t^{dmg}は時間tにおいて対戦相手に与えたダメージ，r_t^{selfdmg}は時間tにおいて自身が受けたダメージの値を表わす．FightingICE では対戦相手に攻撃を当て，体力を減らすことで勝利に近づくほか，対戦相手に近づくことで攻撃を当てることができる．そのため対戦相手の体力を減らしたことへの報酬に加え，対戦相手に近づいたことへの報酬を設定する．同様に，自身の体力を減らすことで勝利から遠ざかるため，自身の体力を減らすことへ負の報酬を与える．また得られた報酬が小さかった場合，未学習の行動を優先させるため−10 の報酬を与える．

4.3. Q 値の更新

時間tで状態sの時に行動aをとったときのQ値は以下の式で更新される．

$$Q(s,a) \leftarrow Q(s,a) + \alpha\big(R_t - Q(s,a)\big)$$

$$R_t = \sum_{i=t+1}^{t+30} r_i$$

tは単位をフレームとする時間，sは現在の状態，aは AI が行う行動，R_tは一定時間で得られた報酬の合計，r_tは時間tに得られた報酬の値，αは近似の補正値(定数)である．通常の Q 学習では，行動を行った直後に得られた報酬値を用いるが，FightingICE のように即時的に報酬を得ることができない環境では有効ではない．そのため本研究では代わりに一定時間内に得られた報酬の合計を用いることで行動に対する報酬を計算する．

5. 提案手法

本研究では，従来の Q 学習に対し少ない学習時間で最適な行動を出力できる手法の開発を行う．前述の Masson らの研究を基に，本研究は行動についても予測可能な手法を構築し，最適な行動を出力する効率をさらに向上させる．

5.1. 重回帰分析

Masson らの手法では，行動の中の動作については従来の Q 学習と同様，過去の学習の結果を引用するにとどまっているため，さらに効率を上げられる余地が残っている．具体的には，これまでの Q 値の記録からまだ学習できていない行動に対しても実際に選択する前に Q 値を予測することで最適な行動を出力する効率の向上を図る．

本手法に用いる式は以下のとおりである．

$$Q(s,a)_{\mathrm{reg}} = \beta_{s,0} + \sum_{i=1}^{n} x_{a,i}\beta_{s,i}$$

$$\begin{bmatrix}\beta_{s,0}\\ \beta_{s,1}\\ \beta_{s,2}\\ \vdots\\ \beta_{s,n}\end{bmatrix} = (X^TX)^{-1}X^T\begin{bmatrix}Q(s,a_1)\\ Q(s,a_2)\\ \vdots\\ Q(s,a_m)\end{bmatrix}$$

$$X = \begin{bmatrix}1 & x_{a_1,1} & x_{a_1,2} & \cdots & x_{a_1,n}\\ 1 & x_{a_2,1} & x_{a_2,2} & \cdots & x_{a_2,n}\\ \vdots & \vdots & \vdots & & \vdots\\ 1 & x_{a_m,1} & x_{a_m,2} & \cdots & x_{a_m,n}\end{bmatrix}$$

ここで，sは現在の状態，aは選択する行動，$Q(s,a)_{\mathrm{reg}}$は状態sにおいてaを行ったときに予測される Q 値，iは行動に記録されている複数のパラメータに便宜上つけた番号，$\beta_{s,i}$は状態sにおけるi番目の重回帰係数，$\beta_{s,0}$は 0 番目の重回帰係数(Y 切片)，$x_{a,1}$は図 2 中のマトリクス値，nはパラメータの総数，mは行動の数(動作の数と一致)である．本式の中で，使用されている行列計算部分は図 2 のマトリックス表の数値を使用した行列計算式となっている．以上の計算を状態sにおいて選択できる行動すべてに対して行い，予測される Q 値を計算する．

5.2. 多次元の重回帰分析

重回帰分析において，説明変数のパラメータを二乗した値を用いると，線形でない相関の予測が可能になる．本研究ではパラメータが大きければ大きいほど Q 値が大きくなると限らないと考えられるものに対して，そのパラメータを二乗したパラメータを追加することで，得られる Q 値のより正確な推定が可能になり，さらに効率よく最適な行動を学習できるようになると考えられる．

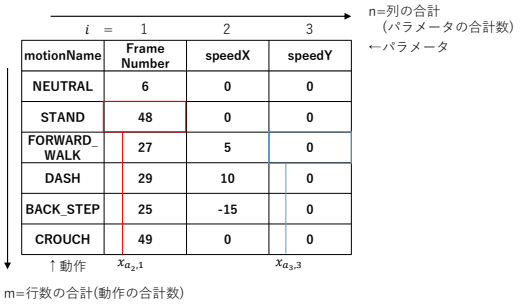

図 2　$x_{a_m,n}$マトリックス表

6. 実装

本研究では Java 言語を用いて開発を行った．学習機構では，Q 学習を計算するクラスを，AI が選択できる行動のリスト，状態の区分のリスト，αの値を与えることで初期化する．初期化されたデータと同クラスに用意されている関数によって Q 値を更新する．

重回帰分析式の相関係数は使用するパラメータの数，選択できる行動の数，Q 値のデータを入力することで求められる．求められた相関係数は Q 学習機構クラスの持つ変数に格納され，最大の Q 値を持つ行動を推定する．選択できる行動それぞれに対して，パラメータと相関係数を掛け合わせ，その合計の値を Q 値として推定する．

7. 実験

本研究の提案手法を Thunder と対戦させることで学習させる．その後，学習させた AI を Thunder や MctsAI と，3 セット 100 回戦の計 300 回直接対戦させ，お互いの体力の差をスコアとして結果を出す．

7.1. 対 Thunder

提案手法と Thunder を対戦させた結果を図 3 に示す．結果から，大部分で獲得したスコアがマイナスであり，Thunder に対して全体として敗北していることが見られる．しかし，円で囲まれている部分では Thunder よりも高いスコアを獲得する場合も見られる．

7.2. 対 MctsAI

提案手法と Thunder を対戦させた結果を図 4 に，また，状態の区分について，攻撃を受け付けない状態などの対戦相手の状態による区分を増やし，対戦させた結果を図 5 に示す．結果を見ると，どちらも MctsAI に対してより高いスコアを獲得することが複数回あり，図 5 の円で囲まれた部分では連続でスコアが上回る部分が見られるが，全体としてスコアが下回っている．

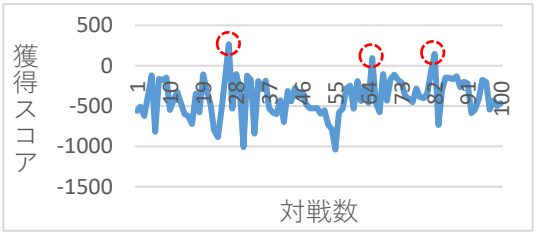

図 3　Thunder との対戦スコア

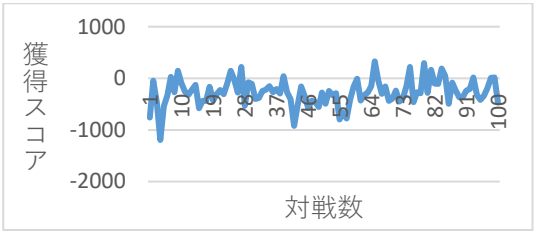

図 4　MctsAI との対戦スコア

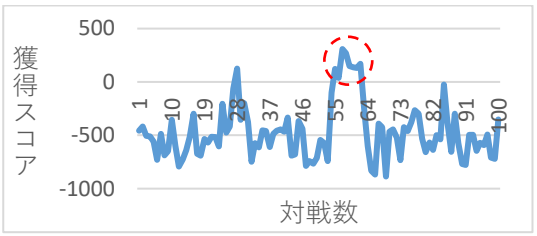

図 5　状態区分を増やしたときの MctsAI との対戦スコア

8. 議論

対 Thunder では稀にスコアが上回ることがあったが，対 MctsAI では全体的にスコアが大きく下回り，学習が上手くいっていない状態となった．対 Thunder に注目すると，大部分でスコアが下回ったが，一部ではスコアが上回ることもあった．MctsAI 対 Thunder の対戦結果と比較すると，MctsAI は Thunder に対してスコアが上回ることが一度もなかったため，対 Thunder においては MctsAI を上回っていると考えられる．しかし，対 MctsAI ではスコアが多くの部分で下回っており，直接的な比較では MctsAI よりも優れているとは言えない．

これらの原因として，提案手法が重回帰分析式に大きく影響されるためだと考えられる．実際に記録された Q 値は重回帰分析の計算に使用され，AI が行動を選択するに際しては予測された Q 値のみを使っている．Q 値の予測をするための相関係数は重回帰分析によって算出されるが，計算結果は目的変数によって，つまり保存されている Q 値のデータによって大きくぶれが生じ，正確な分析ができなくなる．

また，提案手法では Q 値のデータの初期値を全て 0 にしていたことが予測に影響していたことが考えられる．本来 Q 値が大きくなるはずの行動も Q 値の実データが 0 として分析され，予測される Q 値は低くなる．さらに，算出される相関係数に影響し，同様なパラメータを持つ行動も予測される Q 値が小さくなると考えられる．

以上より，提案手法は重回帰分析に大きく影響されるため，安定して高いスコアを獲得するのは難しく，従来の手法に対して明確に有効な手法にするためには，より正確な重回帰分析を行う必要がある．具体的には，算出される相関係数を保存されている Q 値の変化に対して大きく変化しないようにする他，Q 値の初期値を 0 にするのではなく，実際に得られる Q 値のデータをある程度取得した後に重回帰分析を利用した学習を始めることなどが挙げられる．

9. おわりに

本研究では従来の Q 学習の学習効率の低さに着目し，FightingICE を研究環境として，膨大な学習量による効率の低さを解消し，なおかつ従来の Q 学習と同等の最適性を確保した行動を選択することを目的とした．学習効率の低さを解消するため，行動の価値を予測し価値が最大の行動を予測することで学習時間を短縮する手法を提案した．提案した手法の実験では，一部では従来手法よりもスコアで上回る結果が得られたが，全体的にはスコアが下回る結果となった．安定して優位性を発揮するためには，記録されている Q 値の変化によって重回帰分析の結果を大きく変動しないようにする等，重回帰分析の精度を高める必要があると考えられる．

文　献

[1] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3-4, pp. 279-292, 1992.

[2] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg and D. Hassabis, "Human-Level Control through Deep Reinforcement Learning," *Nature*, vol. 518, no. 7540, pp. 529-533, 2015.

[3] W. Masson, P. Ranchod and G. Konidaris, "Reinforcement Learning with Parameterized Actions," *Proceedings of the 13th AAAI Conference on Artificial Intelligence*, pp. 1934-1940, 2016.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. v. d. Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel and D. Hassabis, "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, no. 529, pp. 484-489, 2016.

[5] S. Yoshida, M. Ishihara, T. Miyazaki, Y. Nakagawa, T. Harada and R. Thawonmas, "Applying and Improving Monte-Carlo Tree Search in a Fighting Game AI," *Proceedings of the 13th International Conference on Advances in Computer Entertainment Technology*, no. 27, pp. 1-6, 2016.

[6] J. Peters and S. Schaal, "Natural Actor-Critic," *Neurocomputing*, vol. 71, no. 7-9, pp. 1180-1190, 2008.