Bulletin of Networking, Computing, Systems, and Software – www.bncss.org, ISSN 2186-5140
Volume 8, Number 1, pages 54–59, January 2019

Acquisition of Best Bet for 10,000 Yen Game Using Machine Learning

Ryota Komori, Yasuaki Ito and Koji Nakano
Department of Information Engineering, Hiroshima University
Kagamiyama 1-4-1, Higashi-Hiroshima, Hiroshima, 739-8527 Japan
{komori, yasuaki, nakano}@cs.hiroshima-u.ac.jp

***Abstract*—The 10,000 yen game is a two-player game that was originally designed for a programming competition. The game has a simple rule and is classified as a two person zero sum finite definite incomplete information game in game theory. In this paper, we propose the best strategy for the 10,000 yen game. The strategy was found from results acquired by machine learning approach for 10,000 game AI. The machine learning approach uses Deep Q-Network and the learning has been performed by playing games against itself. If a player use the strategy as perfect play, the player can win the game with a probability of at least 50% against any strategies.**

***Index Terms*—reinforcement learning, self learning, incomplete information game, game theory, backward induction**

I. はじめに

機械学習は主に教師あり学習と教師なし学習，強化学習の 3 種類に分けられる．近年，その一つの分野をなす強化学習の手法が目覚ましい精度を出し，大きく発展を遂げている．強化学習とは，図 1 のように環境中に置かれたエージェントが，環境との相互作用を通して，最適な行動を決定する問題を扱う機械学習の一種である．例えば，囲碁や将棋といったゲームは，本質的には最適な行動を選択していくことが目的なので，強化学習とは相性が良い問題である．強化学習のアルゴリズムには，Q 値と呼ばれるある状態における行動の価値を扱う Q-Learning [1] や Sarsa [2] などがある．しかし，これらのアルゴリズムには状態行動空間の爆発と呼ばれる大きな課題がある．そこで，Deep Q-Network [3] のアルゴリズムでは，Q 値を直接表現するのではなく，ニューラルネットワークで近似することによってこの課題を対処している．そして，環境の状態がエージェントの観測として直接渡される場合，行動を出力とする決定的な関数で表すことが可能な最適方策が少なくとも 1 つは存在することが理論的に知られている．

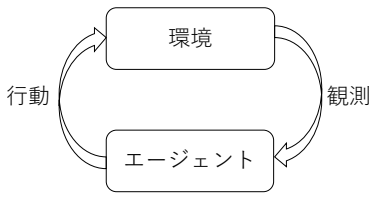

Fig. 1. 強化学習

1 万円ゲーム [4] とは，ゲーム理論における二人零和有限確定不完全情報ゲームに分類される対戦ゲームである．本論文では，機械学習を用いた 1 万円ゲームにおける最強戦略の獲得について提案する．ゲームの終盤における意思決定点では，後退帰納法 [5] によって導き出される最適な戦略を利用する．それ以外の意思決定点では，最適な行動を決定する問題を扱う強化学習アルゴリズムを用いて獲得することができる AI 戦略を利用する．1 万円ゲームは状態や行動が連続値で定義され，その次元は非常に大きい．さらに，本質的には最適な行動を選択していくことが目的である．したがって，Deep Q-Network のアルゴリズムを 1 万円ゲームに適用することで，強力な戦略を獲得する．ここで，自己対戦による学習の例として，囲碁における自己対局を繰り返した学習によってプロ棋士に勝利した AlphaGo Zero [6] がある．本研究では，囲碁の自己対局と同様に，1 万円ゲームにおけるルールのみを与えた自己対戦によってニューラルネットワークは戦略を学習する．そして，データセットなしの自己学習によって得られた戦略の傾向から 1 万円ゲームにおける最適な戦略を発見した．これら 2 つの最適な戦略を合わせることで，全ての意思決定点で選択すべき行動が決定された．結果として，1 万円ゲームにおける最強戦略はあらゆる戦略に対して 50%以上の勝率となった．

II. 1 万円ゲーム

1 万円ゲームはプログラミングコンテスト用に考案された対戦ゲームである．プレイヤーは相手がどの選択をするのかわからない状態で同時に行動を選択する同時手番ゲームである．また，ゲーム理論において二人零和有限確定不完全情報ゲームに分類され，非協力・戦略型ゲームとして表現される．

A. ルール

1 万円ゲームは二人のプレイヤー A と B が対戦するゲームである．各プレイヤーは最初に 10000 円ずつ持っている．この 10000 円を 10 回のラウンドに分けて，場に賭ける．そのときに，少ない金額を出したプレイヤーの方が総取りする．同じ金額なら，それぞれが自分が賭けた金額を取る．つまり，各ラウンド $i(1 \leq i \leq 10)$ における A と B が賭ける金額を，それぞれ a_i と b_i とすると，次のようになる．

- $\sum_{i=1}^{10} a_i = \sum_{i=1}^{10} b_i = 10000$
- $a_i \geq 0$ かつ $b_i \geq 0$
- $a_i < b_i$ のとき，A が $a_i + b_i$ 円を得る．$a_i > b_i$ のとき，B が $a_i + b_i$ 円を得る．$a_i = b_i$ のとき，A と B はそれぞれ $a_i (= b_i)$ 円を得る．

また，各プレイヤーは賭ける金額 a_i と b_i を決定するときに，過去に賭けた金額 $a_1, ..., a_{i-1}$ と $b_1, ..., b_{i-1}$ を知っていて，この情報を利用することができる．10 ラウンド目が終

了したとき，二人のプレイヤーは合計で 20000 円持っていることになる．このときに，獲得した金額が 1 円でも多いプレイヤーの方が，この対戦の勝者となる．そして，この対戦を 1000 回連続で行う，この連続対戦中に過去の対戦履歴を利用することができ，相手が賭ける金額の傾向や戦略を分析して賭ける金額を決定することもできる．1000 回の対戦を行った結果，勝ち数の多いプレイヤーの方が，1 対 1 対戦の勝者となる．対戦の例を図 2 に示す．

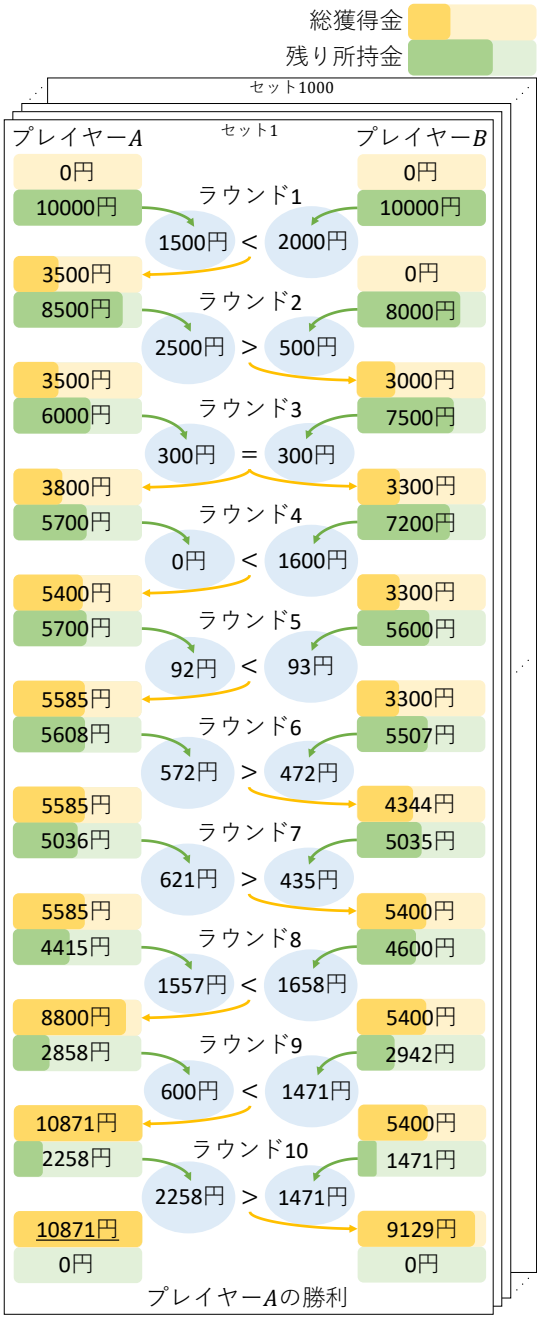

Fig. 2. 対戦例

B. 基本戦略

賭ける金額の合計は 10000 円なので，片方のプレイヤーが全てのラウンドで勝つことはできず，勝ったり負けたりする．基本的な考え方として，相手が賭ける金額の-1 円を賭けると，総取りの金額を最大にすることができるため，勝利に繋がる．しかし，相手が賭ける金額が小さいときの-1 円の場合，それ以外のラウンドで相手が獲得する金額を大きくすることになるため，将来的に勝ちにくくなる．また，相手が賭ける金額よりもかなり大きい金額を賭けると，それ以外のラウンドで自分が獲得する金額を獲得しやすくなるため，将来的に勝ちに繋がる．まとめると，自分の残り所持金 A_i と相手の残り所持金 B_i，自分の総獲得金 P_i，相手の総獲得金 Q_i を用いて図 3 のように，対戦状況を優勢・拮抗・劣勢の領域に分類することができる．

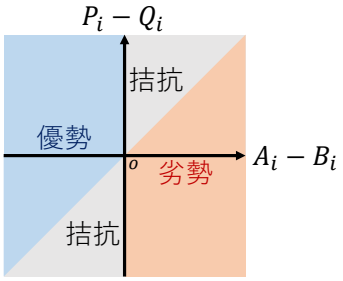

Fig. 3. 優劣の分類

優勢な状態にするためには，次のような戦略に従えばよい．

- 自分の所持金を減らす
  相手が小さな金額を賭けたときに，自分はそれよりかなり大きい金額を賭ける．
- 自分の獲得金を増やす
  相手が大きな金額を賭けたときに，自分はそれよりわずかに小さい金額を賭ける．

これらの戦略に従うためには，対戦中に得た過去の対戦履歴を利用し，相手が賭ける金額の傾向や戦略を分析し対応することが重要である．

C. 9・10 ラウンド最適戦略

1 万円ゲームは有限繰り返しゲームである．したがって，後退帰納法を適用すると，9・10 ラウンド目に自分が賭ける最適な金額 α, β が導き出される．8 ラウンド目が終了したときの自分の残り所持金 A_8 と相手の残り所持金 B_8，自分の総獲得金 P_8，相手の総獲得金 Q_8 を用いて，α と β はそれぞれ次の式 1，2 と求まる．

$$\alpha = \begin{cases} \frac{10000-Q_8}{2} & \text{if} A_8 \leq B_8 \\ \frac{10000-P_8}{2} & \text{else} \end{cases} \tag{1}$$

$$\beta = A_8 - \alpha \tag{2}$$

また，9・10 ラウンドのどちらで大きな金額を賭けるかを決定するために，α と β を 1/2 の確率で入れ替える．つまり，9 ラウンド目に α(もしくは β)，10 ラウンド目に β(もしくは α) を賭ける 2 通りから 1/2 の確率で選択する．

ただし，相手も式 1，2 に基づいて 9・10 ラウンド目に賭ける金額を決定しているとすると，自分は半分ずつの金額を賭けることで，有利な状態にすることができる場合がある．

このとき，9・10 ラウンド目に自分が賭ける金額 a_9，a_{10} は次の式 3 と求まる．

$$a_9 = a_{10} = \frac{10000 - A_8}{2} \tag{3}$$

さらに，$A_8 > B_8$ のとき，10 ラウンド目に相手が賭ける金額 b_{10} は $b_{10} = B_8 - \alpha$ であり，$a_{10} < b_{10}$ ならば自分は勝利する．よって，自分が有利な状態になる条件は次の式 4 と求めることができる．

$$3(A_8 - B_8) < P_8 - Q_8 \tag{4}$$

この条件を満たすとき，相手が式 1，2 に基づいて賭ける金額を決定しているならば自分は勝利する．しかし，相手が式 3 に基づいて金額を決定していると自分は敗北する．したがって，自分が有利な状態にあるとき，2/3 の確率で勝利する．逆に，自分が不利な状態にあるとき．2/3 の確率で敗北する．以上のように，9・10 ラウンド目に最適な金額を賭けることを 9・10 ラウンド最適戦略と呼ぶ．

D. 勝敗の分類

二人のプレイヤーが 9・10 ラウンド最適戦略を実行することで，8 ラウンド目が終了したときの対戦状況を図 4 のように必勝・有利・互角・不利・必敗の領域に分類することができる．

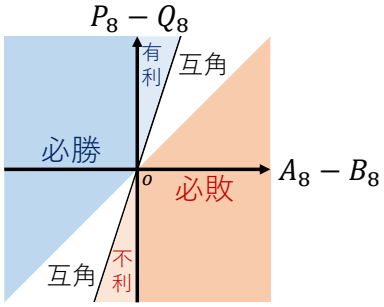

Fig. 4. 勝敗の分類

必勝とは，9・10 ラウンド目において相手がどんな金額を賭けても自分が勝利する状態を表す．有利とは，2/3 の確率で自分が勝利する状態を表す．互角とは，9・10 ラウンドのどちらで大きな金額を賭けるかによって勝敗が決定する状態を表す．ただし，相手が 9・10 ラウンド最適戦略に従っていないならば，自分が必勝の状態になる可能性がある．例えば，8 ラウンド目が終了したときに $A_8 = 4000, B_8 = 7000, P_8 = 2000, Q_8 = 7000$ とすると互角の状態である．ここで，自分は $a_9 = 2500, a_{10} = 1500$，相手は $b_9 = 0, b_{10} = 7000$ と賭けた場合は $P_{10} = 10500$ となり，自分の勝利となる．不利とは，1/3 の確率で自分が勝利する状態を表す．必敗とは，自分がどんな金額を賭けても敗北する状態を表す．ただし，相手が 9・10 ラウンド最適戦略に従っていないならば，自分が勝利する可能性がある．例えば，8 ラウンド目が終了したときに $A_8 = 7000, B_8 = 4000, P_8 = 4000, Q_8 = 5000$ とすると必敗の状態である．ここで，自分は $a_9 = 3000, a_{10} = 4000$，相手は $b_9 = 4000, b_{10} = 0$ と賭けた場合は $P_{10} = 11000$ となり，自分の勝利となる．

III. Deep Q-Network

Deep Q-Network とは，最適な行動を決定する問題を扱う強化学習アルゴリズムの一つである．Q-Learning と呼ばれる強化学習手法の Q 関数部分をニューラルネットワークで近似することによって，高次元な知覚情報から直接 Q 値を推定することを可能にしている．Deep Q-Network のアルゴリズムを Algorithm 1 に示す．

Algorithm 1 Deep Q-Network

Initialize a replay memory D to capacity N
Initialize action-value function Q with random weights θ
for episode=1, M **do**
  Initialize sequence s_1 and prepeocessed sequenced $\phi_1 = \phi(s_1)$
  while not T **do**
    With probability ϵ select a random action a_t, otherwise select $a_t = \max_a Q^*(\phi(s_t), a)$
    Execute action a_t in emulator and observe reward r_t and terminal T
    Set $s_{t+1} = s_t, a_t$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in D
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from D
    $y_j = \begin{cases} r_j & \text{if } T = \text{True} \\ r_j + \gamma \max_{a'} Q(\phi_{j+1}, a') & \text{otherwise} \end{cases}$
    Perform a gradient descent step on $(y_j - Q(s_j, a_j; \theta))^2$ with respect to the network parameters θ
  end while
end for

状態遷移を保存するためのメモリ D の最大保存数を N，ランダムな重みパラメータ θ を持つニューラルネットワークを Q，最大学習回数を M，終了判定を T とする．時刻 t に対する報酬 r の重み付けのために割引率 γ を導入することで，累積報酬 R_t は $R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + ... + \gamma^{n-t-1} r_n = r_{t+1} + \gamma R_{t+1}$ となる．強化学習における目標は，この累積報酬を最大化するような戦略 Q^* を発見することである．

Deep Q-Network のアルゴリズムにおけるプロセスを繰り返すことで，状態 s と行動 a，報酬 r における遷移のセット $s_0, a_0, r_1, s_1, a_1, r_2, s_2, ..., s_{n-1}, a_{n-1}, r_n, s_n$ が得られる．これがマルコフ決定過程での強化学習の設定になる．つまり，状態を観測し行動を実行すると，環境の中で状態が確率的に遷移し報酬が得られる．また，マルコフ決定過程はマルコフ性を持っているため，次の状態 $s' = s_{t+1}$ は一時刻前の状態 s_t と行動 a_t だけによって決まる．つまり，現在の状態と行動から，次の時刻の状態と報酬を予測することができる．さらに，繰り返し計算することにより，すべての将来の状態と報酬を予測することが可能になる．

また，戦略 Q を実行し続け，状態を観測した後に得られる利得の期待値 $\mathbb{E}$ として，行動価値関数 $Q^*(s, a)$ は $Q^*(s, a) = max_\pi \mathbb{E}[R_t | s_t = s, a_t = a, \pi]$ と表せる．これは，ベルマン方程式 $\mathbb{E}[r + \gamma \max_{a'} Q^*(s', a') | s, a]$ に変形できる．したがって，選択する戦略に依存せず，ある状態における利得を計算できるようになる．つまり，1 つの環境下での最適な行動を選択できるようになる．学習する時に利用する ϵ-greedy 法及び Experience Replay についてそれぞれ説明する．

A. ϵ-greedy 法

ϵ-greedy 法とは，局所探索法と並んで近似アルゴリズムの最も基本的な考え方の一つである．具体的には，確率 ϵ でランダムに行動を選択し，確率 $1-\epsilon$ で最適な行動を選択する．ただし，行動を選択するたび ϵ を減少させる．これにより，Q 値の初期値に依存することなく，様々な行動に対する適切な Q 値の学習が可能となる．エージェントの行動はこの ϵ-greedy 法に基づいて決定される．

B. Experience Replay

Experience Replay とは，Replay Memory D に観測した遷移 (s, a, r, s') を保存し，D からランダムサンプリングした遷移を学習に利用する手法である．強化学習において与えられるデータは，一般的に時系列的に連続したものである．データ間に相関があると学習が収束しなくなってしまうため，データ間に生じる依存関係を除去する必要がある．そのため，Experience Replay によるデータ間の無相関化を行う．

IV. 学習手法

Deep Q-Network の強化学習アルゴリズムを 1 万円ゲームに適用することで，強力な AI 戦略を獲得する．強化学習が対象とする問題を図 5 のようにモデル化 $P(s'|s, a)$ する．エージェント，行動，環境についてそれぞれ説明する．

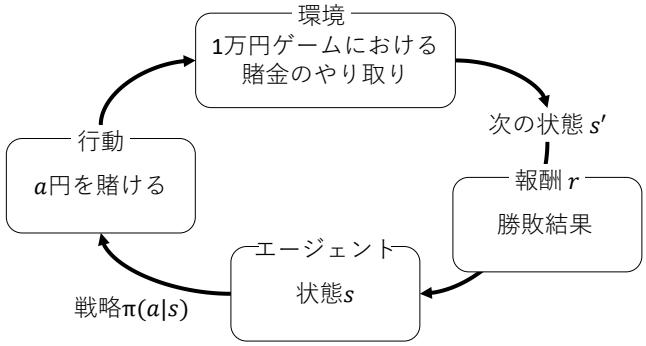

Fig. 5. モデル $(s'|s, a)$

A. エージェント

エージェントは 1 万円ゲームにおけるプレイヤーを表す．プレイヤーはある状態 s にあるとき，賭ける金額を選択することで報酬を得る．プレイヤーが利用可能な情報は連続対戦中に得た過去の対戦履歴である．これは過去と現在の対戦データに分けることができる．過去の対戦データから，直近 5 回の対戦で 1 から 8 ラウンド目に各プレイヤーが賭けた金額とその時の勝敗結果を過去の対戦状況として作成する．現在の対戦データから，現在のラウンド i と自分の残り所持金 A_i，相手の残り所持金 B_i，自分の総獲得金 P_i，相手の総獲得金 Q_i を用いて現在の対戦状況として作成する．この過去と現在の対戦状況を連結し，各要素ごとに正規化することで状態 s を作成する．

B. 行動

連続な行動空間を扱うために，行動空間を適当に離散化する必要がある．よって，1 万円ゲームにおける行動空間とは $K(0 \leq K \leq 10000)$ 円を賭けるということである．ただし，10000 円を賭けるという選択肢は敗北または引き分けの結果にしかならないため，選択する必要性がない．その選択肢を除くと，選択可能な行動群は $\{0$ 円を賭ける, 1 円を賭ける, ..., 9999 円を賭ける $\}$ となる．この行動群から 1 つ選択し，その選択肢を行動 $a(0 \leq a \leq 9999)$ とする．

行動を選択するためにはニューラルネットワークを利用する．ニューラルネットワークは状態 s を入力として，各行動 a に対する行動価値 $Q(s, a)$ を出力する．つまり，入力は対戦中に得られる対戦履歴であり，出力は行動を選択したときの見込み報酬である．また，出力層における 10000 ユニットの中で最適な行動を 1 つ選択する際に，行動は自分の残り所持金以下に限定される．そして，行動価値 $Q(s, a)$ が最大値を出力しているユニットを一つ選択することで，最適な行動を決定する．

C. 環境

環境は 1 万円ゲームのルールに基づいて行われる対戦を表す．各ラウンド i では，勝敗が確定しているかを判定する．勝敗の基準は総獲得金であり，二人が賭ける金額の合計は 20000 円である．したがって，自分の総獲得金 P_i または相手の総獲得金 Q_i が 10000 円を上回ったとき勝敗は確定する．また，9・10 ラウンド最適戦略に従うと，8 ラウンド目が終了した時点で勝敗が必勝・有利・互角・不利・必敗に分類される．それぞれの勝率は $1, 2/3, 1/2, 1/3, 0$ と決定できる．以上より，終了判定 T は次式 5 のように表せる．

$$T = \begin{cases} \text{True} & \text{if } P_i > 10000 \vee Q_i > 10000 \\ \text{True} & \text{elif } i = 8 \\ \text{False} & \text{else} \end{cases} \tag{5}$$

D. 報酬

報酬は選択した行動に基づき，エージェントが獲得する評価を表す．勝敗が決定したとき，環境はエージェントに報酬を与える．その報酬は，勝利ならば正，敗北ならば負，引き分けならば 0 である．また，報酬のクリッピングによって学習が進みやすくなる効果を持つため，与える報酬を正ならば $+1$，負ならば -1 と固定する．つまり，8 ラウンド目が終了したときに得られる報酬は必勝ならば $+1$，必敗ならば -1，それ以外ならば 0 となる．さらに，自分の総獲得金が 10000 円を上回ったとき，自分は勝利となるので報酬は $+1$ となる．逆に，相手の総獲得金が 10000 円を上回ったときの報酬は -1 となる．以上より，終了判定 $T =$True であるときに，報酬 r は次式 6 で求めることができる．

$$r = \begin{cases} +1 & \text{if } P_i > 10000 \\ +1 & \text{elif } A_8 \leq B_8 \wedge A_8 + Q_8 \leq 10000 \\ -1 & \text{elif } Q_i > 10000 \\ -1 & \text{elif } A_8 \geq B_8 \wedge A_8 + Q_8 \geq 10000 \\ 0 & \text{else} \end{cases} \tag{6}$$

ただし，$A_8 = B_8$ かつ $A_8 + Q_8 = 10000$ のとき報酬は 0 となる．

E. ニューラルネットワーク

ニューラルネットワークの目的は，損失関数の値をできる限り小さくするような重みパラメータを見つけることである．得られた報酬からすぐにフィードバックをかけて学習させるとうまくいかないことが多い．したがって，重みの更新が行われているニューラルネットワークのコピーをとっておき，そこから算出する．このコピーを Target-Network [7] と呼ぶ．行動を選択するたび，Target-Network $\hat{Q}$ の重みパラメータ θ^- に Q-Network Q の重みパラメータ θ をコピーすることで更新する．この Target-Network を利用することで，学習を効率的に行うことができる．

Dropout [8] とは，ニューロンをランダムに消去して学習する手法である．これを利用することによって過学習を防ぐ効果がある．本研究で使用するニューラルネットワークでは，入力直後では 80%のニューロンを活性化し，それ以降の層では 50%を活性化させる．

ニューラルネットワークへの入力は過去と現在の対戦状況からなる状態 s である．したがって，図 6 のように入力は過去 (input_1) と現在 (input_2) に分岐させる．それぞれの入力はアフィン変換の後に連結される．

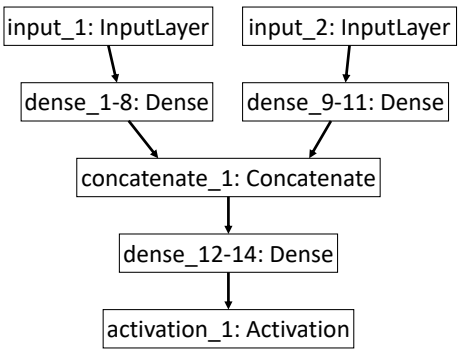

Fig. 6. 分岐ニューラルネットワーク

F. 自己対戦

データセットなしでルールのみを与えた自己対戦を繰り返し行うことによって，ニューラルネットワークは AI 戦略を獲得する．プレイヤー A は ϵ-greedy 法に従って行動を選択し，戦略を学習する．もう一方のプレイヤー B はプレイヤー A が学習した戦略に基づいて最適な行動のみ選択する．したがって，学習初期では，どちらのプレイヤーもランダムな行動を選択することになる．学習中期以降では，プレイヤー A はランダムもしくは最適な行動を確率 ϵ に基づいて決定する．プレイヤー B は現時点における学習した戦略に基づいて最適な行動を選択するようになる．

V. 実験

A. ハイパーパラメータ

Experience Replay において，学習が開始した時点では状態遷移が貯まっていないので，最初にランダムな行動を 20000 回行い，遷移を蓄積する．また，遷移の最大保存数 N =160 万を超えた分は古い遷移から消えていくようにする．そして，行動を選択するたびにランダムな 64 個の遷移をサンプリングして学習に利用する．ϵ-greedy 法において，初期値 $\epsilon = 1.0$ とし，そこから 300 万回の行動を選択していく間に $\epsilon = 0.05$ まで線形に減少させ，それ以降は $\epsilon = 0.05$ に固定する．Target-Network $\hat{Q}$ において，10000 回の行動をするたびに戦略を更新する．また，最適な重みパラメータを探索するための最適化手法には，適応的学習率の方式である RMSprop を利用する．学習率において，図 7 のように損失関数が最も減少傾向にある $\eta = 10^{-4}$ を採用する．そして，ニューラルネットワークが識別を行う際の処理としての活性化関数には，ReLU(Rectified Linear Unit) を利用する．

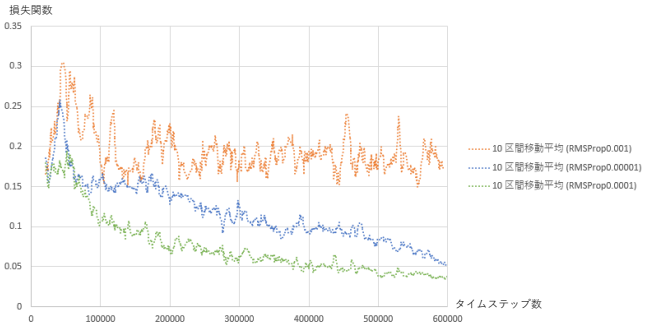

Fig. 7. RMSProp と学習率

B. 学習結果

TensorFlow の可視化を補完するツールである TensorBoard を利用し，トレーニングにおける損失関数と報酬の推移をそれぞれ図 8，9 及び 10 に示す．

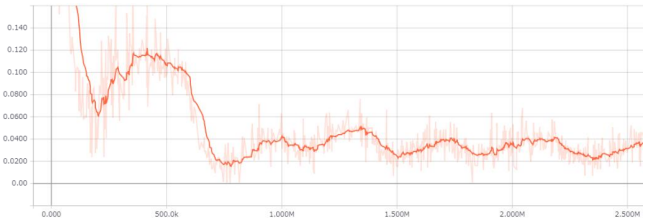

Fig. 8. 損失関数

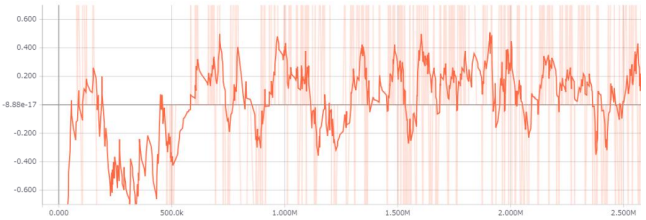

Fig. 9. 報酬

損失関数や報酬は学習が進んでいるかどうかを表す指標としてみなすことができる．損失関数は減少傾向にあることから，相手が賭ける金額の傾向や戦略に対応していることがわかる．報酬が増加傾向にあることから，自分の最適な戦略に勝利する確率が上がっていることがわかる．また，自己対戦によって得られる報酬は期待通りの 0 前後となっている．

続いて，対戦が終了するまでのラウンド数の推移を図 10 に示す．これは終了条件 T が満たされたときのゲームのラウンド数を表している．8 ラウンド目が終了すると勝敗は決定されるため，1 回の対戦における最大ラウンド数は 8 となる．最初，自分はランダムな行動を選択する．よって，相手は最適な行動として小さな金額を賭けることになる．すなわち，学習序盤における対戦は一時的に長期化する傾向がある．その後すぐに，自分も最適な戦略を実行するようになるため，ゲームの序盤で対戦が決するように変化している．それ以降では，対戦が長期的に継続していることがわかる．

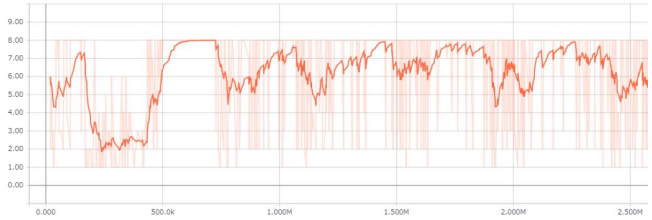

Fig. 10. 対戦が終了するまでのラウンド数

まとめると，自己学習の過程では，選択される金額に次のような傾向が見られた．

・学習初期
　ランダムに金額を選択する．
・学習中期
　5000 円以上のような大きい金額がほとんど選択されなくなり，次第に低額になる．
・学習後期
　10 円未満のようなかなり小さい金額を常に選択するようになる．

全体を通して，学習が進むにつれて勝敗の確定がゲームの終盤になる傾向がある．学習の結果，1〜8 ラウンド目で常に 10 円未満の金額を選択する戦略が強力であることがわかった．

C. *ALL 1* 戦略

学習によって得られた傾向から 1 万円ゲームにおける最適な戦略を発見した．その戦略とは，1 から 8 ラウンド目で常に 1 円，9・10 ラウンド目で 9・10 ラウンド最適戦略に基づいて賭けることである．この最適な戦略を ALL 1 戦略と呼ぶ．ALL 1 戦略を実行すると，次のどちらかの条件を満たすことで必勝の状態になる．

- 8 ラウンド目が終了するまでに自分の総獲得金 P_i が 10000 円を上回る．
- 8 ラウンド目が終了したときに相手の残り所持金 B_8 が 9992 円以上である．

条件を満たさない場合，有利または互角の状態になる．つまり，ALL 1 戦略はあらゆる戦略に対して 50%以上の勝率となる．

D. 比較検証

ALL 1 戦略の有効性を示すために，ALL 0 戦略及び ALL 2 戦略と比較を行う．各戦略は，1 から 8 ラウンド目で常に 0 円，2 円を賭ける．ALL 0 戦略の場合，相手が 1 円以上賭けた場合に必敗になる．例えば，自分は ALL 0 戦略を実行し，相手は 1 から 8 ラウンド目で常に 1 円を賭ける戦略を実行すると自分は敗北する．ALL 2 戦略の場合，8 ラウンド目が終了したときに，相手の残り所持金が自分の残り所持金よりも少ないと必敗になる可能性がある．例えば，自分は ALL 2 戦略を実行し，相手は 1 ラウンド目で 10 円，2 から 8 ラウンド目で常に 1 円を賭ける戦略を実行すると自分は敗北する．

まとめると，ALL 1 戦略を実行することで，8 ラウンド目が終了したときに図 11 のように必ず必勝・有利・互角の領域内のどこかになる．したがって，ALL 1 戦略に対する必勝は存在せず，ALL 1 戦略はあらゆる戦略に対して互角以上になる．また，ALL 1 戦略同士の対戦は必ず引き分けになる．

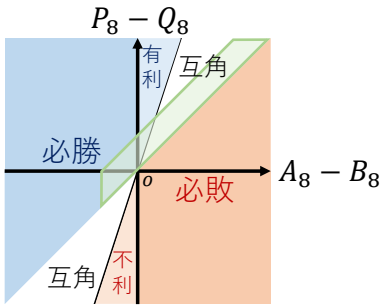

Fig. 11. ALL 1 戦略の勝敗

VI. まとめ

1 万円ゲームとは，ゲーム理論において二人零和有限確定不完全情報ゲームに分類される対戦ゲームである．1 万円ゲームにおける 1 から 8 ラウンド目で AI 戦略に従い，9・10 ラウンド目で 9・10 ラウンド最適戦略に従う．AI 戦略では，深層強化学習のアルゴリズムである Deep Q-Network を用いて獲得することができる戦略を利用する．データセットなしでルールのみを与えた自己対戦を行い，強力な戦略を学習した．そして，その自己学習によって得られた傾向から最適な戦略を発見した．9・10 ラウンド最適戦略では，後退帰納法によって導き出される最適解を利用する．結果として，1 万円ゲームにおける最強戦略はあらゆる戦略に対して 50%以上の勝率となった．

REFERENCES

[1] Christopher J. C. H. WatkinsPeter Dayan, ”Q-learning” Machine Learning, 8, 279-292 (1992), 1989.
[2] V. Mnih, K. et al., “ Playing Atari with Deep Reinforcement Learning ” NIPS, arXiv:1312.5602, 2013.
[3] Marc G Bellemare et al., ”The arcade learning environment: An evaluation platform for general agents”, Journal of Artificial Intelligence Research, 47:253–279, 2013.
[4] Koji Nakano, ”On the 10000 Yen Game”, The Bulletin of the LA Symposium, in Japanese, vol.52 pp.23-27, 2009.
[5] Jules Hedges, ”Backward Induction for Repeated Games” Department of Information Engineering, EPTCS 275, 2018, pages 35-52.
[6] David Silver et al., ”Mastering the game of Go without human knowledge”, Nature volume 550, pages 354–359, 2017.
[7] V. Mnih et al., “ Human-level control through deep reinforcement learning ” nature, 2015.
[8] Nitish Srivastava1 et al., ”Dropout: A Simple Way to Prevent Neural Networks from Overfitting”, Journal of Machine Learning Research 15 2014, pages 1929-1958, 2014.