# 非ブートストラップ手法を利用した深層強化学習アルゴリズムの提案

○小玉直樹　原田拓（東京理科大学）　宮崎和光（独立行政法人大学改革支援・学位授与機構）

# A Proposal of a Deep Reinforcement Learning Algorithm using Non-bootstrap Method

∗N. Kodama, T. Harada (Tokyo University of Science) , and K. Miyazaki (National Institution for Academic Degrees and Quality Enhancement of Higher Education)

**Abstract**− In recent years, Deep reinforcement learning has attracted attention by a Deep Q-Network. Although the Deep Q-Network shows performance beyond human experts in Atari2600 video game simulation, Q-learning, which is a reinforcement learning used by Deep Q-Network, requires the Markov property on the environment for optimum policy acquisition. On the other hand, although Profit Sharing is known as a robust reinforcement learning for Partial Observation Markov Decision Processes, it is difficult to approximate the divergent evaluation values of Profit sharing by a deep neural network. Therefore, in this study, we propose Deep P-Network which approximates action selection probability instead of the evaluation values of Profit Sharing by the deep neural network. The method is a robust method for Partial Observation Markov Decision Processes using non-bootstrap, and trains the deep neural network using reinforcement values of Profit Sharing. The proposed method is compared with DQN in Pong of Atari2600 video game under Partially Observable Markov Decision Processes.

**Key Words:** deep reinforcement learning, deep q-network, reinforcement learning, q-learning, profit sharing, neural network

## 1　緒言

近年，Deep Q-Network (DQN)[1] を利用した深層強化学習が研究者の注目を集めている．DQN は伝統的な強化学習手法である Q-learning [2] と深層ニューラルネットワークを組み合わせた手法であり，Atari2600 ビデオゲームシミュレーション [3] の多くのゲームにおいて，人間のエキスパートを超える結果を得ている．また，DQN は様々な拡張機能も提案されており，その性能は向上し続けている [4, 5, 6, 7, 8]

Q-learning のような動的計画法に基づく手法は，ブートストラップと呼ばれる後続状態の行動価値の推定量に基づいて現在の行動価値の推定値を更新する手法を利用し，環境がマルコフ決定過程 (Markov Decision Processes; MDPs) であれば期待報酬を最大化するような最適解が得られることが知られている．しかし，実環境では不完全知覚問題 [9] によって部分観測マルコフ決定過程 (Partially Observable Markov Decision Processes; POMDPs) の下での学習が求められる場合がある．そのような環境では，Q-learning による学習は最適解への収束の保証を失ってしまう．

一方，Q-learning とは異なる強化学習手法として Profit Sharing [10] が提案されている．Profit Sharing は非ブートストラップ手法であり，合理性定理 [11] に従うことで一部の POMDPs の下でも報酬獲得までの政策の獲得が保証される．そのため，Profit Sharing は不完全知覚問題やマルチエージェント環境において，Q-learning よりも有効であることが示されている [12]．また，Profit Sharing は DQN との組み合わせとしても利用されている．DQNwithPS [13] や Learning-accelerated DQN [14] では，DQN に Profit Sharing の強化関数を目標値とした学習を追加することで，学習速度の向上に貢献している．しかし，Profit Sharing 単一と深層ニューラルネットワークの組み合わせはまだ実現されていない．それは，Profit Sharing は各状態-行動の評価値に強化値を加算していくことで学習を行うため，その評価値は発散し，DQN と同じようにニューラルネットワークでの評価値の近似が行えないためである．そこで本研究では，評価値ではなく行動選択確率を近似するニューラルネットワークを使って学習することで，抽象的な入力が扱える非ブートストラップ手法の Deep P-Network を提案する．

提案手法は Atari2600 ビデオゲームシミュレーションの Pong を利用し，DQN と学習性能の比較を行う．その際，本提案手法の特徴である非ブートストラップ手法の有効性を検証するため，入力のゲーム画面の画像数を減らすことで速度情報を無くした POMDPs 環境での比較も行い，不完全知覚環境での検証も行う．

## 2　背景

### 2.1　強化学習

各離散化された時間ステップ $t = 0, 1, 2, \cdots$ において，状態 $s_t$ が環境からエージェントに与えられ，エージェントは行動 $a_t$ を選択し環境に応答する．そして，報酬 $r_t$ と次の状態 $s_{t+1}$ が環境からエージェントに与えられる．この相互作用は *Markov Decision Process* の状態遷移 $(s_t, a_t, r_t, s_{t+1})$ の連鎖として形式化される．行動は，各状態での行動の確率分布として定義される政策 $\pi$ によって与えられる．

### 2.2　Q-learning

Q-learning におけるエージェントの目的は，将来の報酬の割引総和として定義される割引収益 $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$ の最大化である．Q-learning では，行動価値関数 $Q^\pi(s, a) = \mathbb{E}_\pi[G_t | s_t = s, a_t = a]$ を利用することによって，期待割引収益の推定値の学習を行う．

その行動価値関数は Eq. (1) のベルマン方程式に従う．

$$Q^*(s,a) = \mathbb{E}[r_t + \gamma \max_{a'} Q^*(s_{t+1}, a') | s_t = s, a_t = a] \tag{1}$$

ただし，実際には Eq. (2) のように価値反復として利用される．

$$Q_{k+1}(s,a) = \mathbb{E}_\pi[r_t + \gamma \max_{a'} Q_k(s_{t+1}, a') | s_t = s, a_t = a] \tag{2}$$

反復回数 $k$ が無限大に近づくにつれ，価値反復アルゴリズムは最適行動価値関数 $Q_k \to Q^*$ に収束することが保証されている．

Q-learning のような動的計画法に基づく手法は後続状態の行動価値の推定量に基づいて，現在の行動価値の推定値を更新する．このような更新法はブートストラップと呼ばれ，環境が MDPs であれば期待報酬を最大化するような最適政策 $\pi^*$ が得られる．

### 2.3 Deep Q-Network

Deep Q-Network (DQN) は Q-learning における行動価値関数を深層ニューラルネットワークで近似することによって，画像などの抽象的な入力を扱うことができる深層強化学習手法である．DQN は，複数枚の前処理済みのゲーム画面のみを入力として扱うだけで，Atari2600 ビデオゲームシミュレーションの多くにおいて人間のエキスパートを超える結果を得ている．

DQN では，各ステップにおいて，エージェントは，環境との相互作用を通じて得られた経験 $(s_t, a_t, r_t, s_{t+1})$ をリプレイメモリ [15] に保存する．DQN は Eq. (3) による誤差の最小化として，確率的勾配降下法によるニューラルネットワークの学習パラメータ $\theta$ の最適化を行う:

$$(r_j + \gamma \max_{a'} Q_{\bar{\theta}}(s_{j+1}, a') - Q_\theta(s_j, a_j))^2 \tag{3}$$

ここで，$j$ はリプレイメモリからランダムにサンプルされた時間ステップ，$\bar{\theta}$ はターゲットネットワークの学習パラメータである．誤差勾配は $\theta$ にのみ逆伝播され，ターゲットネットワークは直接最適化されず，定期的に $\theta$ からのコピーが取られる．

### 2.4 Profit Sharing

経験強化型学習（Exploitation-oriented Learning;XoL) [16] として知られる Profit Sharing では，探索と利用のトレードオフにおいて，利用を重視する強化学習手法である．Profit Sharing は非ブートストラップ手法であるため，POMDPs の下でも比較的頑健である特徴を持つ [12]．特に，Profit Sharing は合理性定理 [11] に従うことで一部の POMDPs の下でも報酬獲得までの政策の獲得が保証される．ただし，Profit Sharing におけるエージェントの目的は，最適政策の獲得ではなく，継続した報酬獲得のための政策の獲得である．

Profit Sharing では，正の報酬を観測した時，エピソード内のルールの評価値 $\omega(s_i, a_i)$ は強化関数 $f_i$ によって強化される．

$$\begin{aligned} \omega(s_i, a_i) &\leftarrow \omega(s_i, a_i) + f_i \\ i &= T, T-1, \ldots, T-H+2, T-H+1 \end{aligned} \tag{4}$$

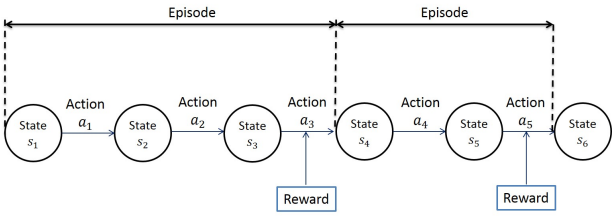


Fig. 1: Example of episodes

ここで，$T$ はエピソード終端のステップ，$i$ はエピソード内の強化されるルールのステップである．エピソードとは，報酬間または初期状態から報酬までのルール系列を指し (Fig. 1)，そのルールの数は $H$ で表される．一般に Profit Sharing で利用される $f_i$ は Eq.(5) の合理性定理を満たす Eq.(6) のような等比減少関数が利用される．

$$L \sum_{j=T-H+1}^{i} f_j < f_{i+1} \tag{5}$$

$$f_i = r_T \lambda^{T-i} \tag{6}$$

ここで，$\lambda$ は割引率であり，L は有効ルールの数である．有効ルールは，いかなるエピソードにおいても迂回経路上に存在するルール（無効ルール）以外のルールとして定義され，迂回経路は，エピソード内に同一状態が存在する場合に，その同一状態間のルール系列として定義される．一般的に有効ルールの数は知ることはできないが，実用上は「行動選択肢の数 −1」とすれば十分である [11]．

Profit Sharing では，$\omega(s_t, a_t)$ を利用して確率的に行動を決定する場合が多い．例えば，ルーレット選択を利用する場合，状態 $s_t$ における任意の行動 $a_k$ の選択確率 $P(a_k|s_t)$ は以下のように計算される：

$$P(a_k|s_t) = \frac{\omega(s_t, a_k)}{\sum_{j=1}^{A} \omega(s_t, a_j)} \tag{7}$$

ここで，$A$ は行動選択肢の数である．

## 3 Deep P-Network

Profit Sharing は離散入力を扱う手法であるため，DQN と同じように画像のような抽象的な入力を扱うことはできない．そこで，DQN のように，Profit Sharing と深層ニューラルネットワークの組み合わせ手法が求められるが，Profit Sharing の評価値を強化し続けることによって発散するという特性を持っている．そのため，評価値をニューラルネットワークで近似する場合，ネットワークの学習パラメータの発散が容易に引き起こされ，学習が上手く行えない問題が発生する．そこで本研究では，評価値 $\omega(s_t, a_t)$ の近似ではなく，行動選択確率 $P(a_k|s_t)$ の近似を行う Deep P-Network (DPN) を提案する．行動選択確率の出力のためには，出力層の活性化関数にソフトマックス関数を利用し，出力値の合計値を 1 とする．

### 3.1 目標値の設定

Profit Sharing では，正の報酬を観測したとき，エピソード内のルール $(s_i, a_i)$ の $\omega(s_i, a_i)$ が $f_i$ によって強化され，Eq. (7) にしたがって，$(s_i, a_i)$ の $P(a_k|s_t)$ は 1 に近づける学習を行う．さらに，この時の $P(a_k|s_t)$

の 1 への収束速度は $f_i$ の大きさに依存して変化するという学習特性を持つ．DPN では，これらを踏まえて目標値を設計する．

DPN では正の報酬を観測したとき，エピソード内のルール $(s_i, a_i)$ の選択確率 $P_\theta(a_k|s_t)$ が $f_i$ に依存した収束速度で 1 に近づくような目標値を設定して，ニューラルネットワークの訓練のための誤差を計算する．具体的には，目標行動選択確率 $\bar{P}_\theta(a_k|s_t)$ は以下のように設定される．

$$\bar{P}_\theta(a_k|s_t) = \begin{cases} (1-f_i)P_\theta(a_k|s_t) + f_i & (a_k = a_i) \\ (1-f_i)P_\theta(a_k|s_t) & (otherwise) \end{cases} \quad (8)$$

ただし，$0 \leq \bar{P}_\theta(a_k|s_t) \leq 1$ であるため，$0 \leq f_i \leq 1$ である必要がある．したがって，報酬は $0 \leq r_T \leq 1$ の範囲で設定しなければならない．

DPN は行動選択確率分布 $P_\theta(a_k|s_t), k = 1, 2, \ldots, A$ を目標行動確率分布 $\bar{P}_\theta(a_k|s_t), k = 1, 2, \ldots, A$ に近づける学習を行う．ニューラルネットワークの訓練は誤差の最小化問題として確率的勾配降下法が用いられるため，DPN ではそれぞれの確率分布間の誤差を計算する必要がある．したがって，DPN では確率分布間の尺度として定義されている交差エントロピーを利用する．このとき，$\bar{P}_\theta(a_k|s_t)$ と $P_\theta(a_k|s_t)$ の交差エントロピーは Eq. (9) のようになる．

$$-\sum_{k=1}^{A} \bar{P}_\theta(a_k|s_t) \log P_\theta(a_k|s_t) \quad (9)$$

したがって，ニューラルネットワークの訓練では，Eq. (9) を最小化するために確率的勾配降下法が利用される．

### 3.2 アルゴリズム

DPN の学習フローチャートを Fig. 2 に示す．DPN はエピソードのルール系列保持のために，毎ステップにおいて，エージェントと環境の相互作用の後，経験 $(s_t, a_t, t)$ を一時的に保存しておく．そして正の報酬を観測したとき，エピソード内のルールと強化値が replay memory に保存される．ネットワークの訓練は，報酬観測に関わらず毎ステップ行われる．ここでは，DQN と同じように replay memory からいくつかのサンプルをランダムサンプリングし，それぞれのサンプルに対して，Eq. (8) にしたがって目標分布を計算する．その後，それぞれのサンプルに対して，Eq. (9) によって交差エントロピーを計算し，確率的勾配降下法を用いてミニバッチ学習が行われる．

DPN のアルゴリズムは Algorithm 1 に示す．

## 4 Atari2600 シミュレーション

### 4.1 評価方法

本論文では，Atari2600 のゲームの Pong におけるスコアの比較から手法の性能の評価を行う．Pong は，DQN が最高スコアで収束できることから，提案手法も同様に最高スコアで収束できるかどうかという点と収束までの速さという点で比較が行いやすいため，手法の性能評価が行いやすいという点から選択した．エージェントのスコアは，100,000 ステップ毎に学習を止め 10 ゲーム行い，それぞれのスコアの平均値を取ることで得られる．ここで，各ゲームからスコアが出力される度に 1 ゲームとカウントされる．なお，スコアは $-21$ から 21 の間の整数値を取り，より高いスコアを獲得するほど，性能が高いと評価する．

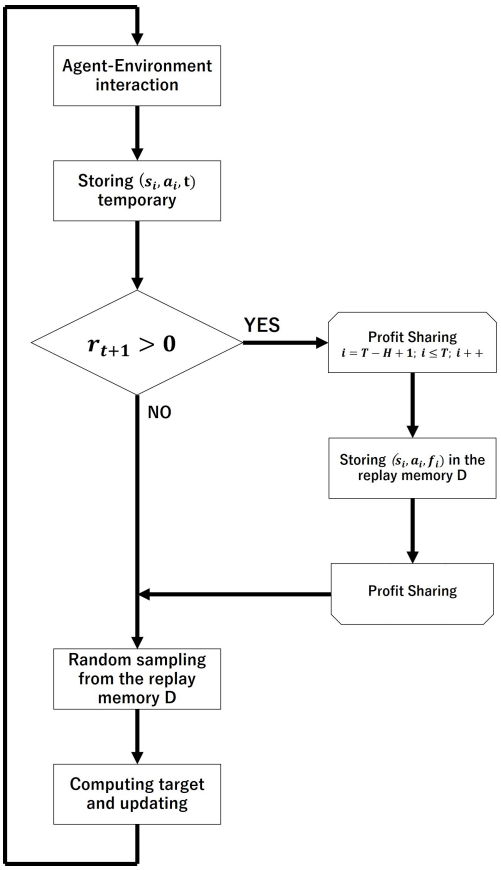

Fig. 2: The flowchart of Deep P-Network

**Algorithm 1** Deep P-Network

**Input:** budget $T$, minibatch $k$
Initialize replay memory $D$ to capacity $N$
Initialize episode start step $\tau = 0$
Randomly initialize weight $\theta$
Observe $s_0$
**for** $t = 0$ to $T$ **do**
  Choose $a_t$ and observe $r_t$, $s_{t+1}$
  Store experience$(s_t, a_t, t)$ temporary
  **if** $r_t > 0$ **then**
    **for** $i = \tau$ to $t$ **do**
      Compute $f_i = r_t \lambda^{t-i}$
      Store rules and reinforcement values $(s_i, a_i, f_i)$ in $D$
    **end for**
    **for** $j = 1$ to $k$ **do**
      Sample random minibatch of rules and reinforcement values $(s_j, a_j, f_j)$ from $D$
      Set

$$\bar{P}_\theta(a_k|s_t) = \begin{cases} (1-f_i)P_\theta(a_k|s_t) + f_i & (a_k = a_i) \\ (1-f_i)P_\theta(a_k|s_t) & (otherwise) \end{cases}$$

    **end for**
  **end if**
  Update $\theta$ using $-\sum_{k=0}^{A} \bar{P}_\theta(a_k|s_t) \log P_\theta(a_k|s_t)$
  **if** $r_t \neq 0$ **then**
    Update $\tau \leftarrow t + 1$
  **end if**
**end for**

各手法で扱うニューラルネットワークでは，グレースケール化と $84 \times 84$ pixel にリサイズした直近の複数フレームのゲーム画面の画像が入力される．複数フレームの画像入力は，ゲームのオブジェクトの速度情報を得るために必要であり，入力画像が 1 フレームの場合，エージェントはオブジェクトの速度情報を知ることは

出来ない．本実験では，入力画像の枚数を 1, 2, 4 枚のそれぞれ実験し，速度情報に関する POMPDs 環境で提案手法の有効性の検証を行う．

ゲーム冒頭には，ランダムな初期状態からスタートさせるために，最大 30 ステップの何も行動をしない期間を挿入する．

### 4.2 ハイパーパラメータの設定

DQN と DPN は行動戦略として $\varepsilon-$greedy 戦略を利用する．Table 1 には，本実験で利用する $\varepsilon-$greedy 戦略のハイパーパラメータを示す．ここで，initial exploration は $\varepsilon-$greedy 戦略における $\varepsilon$ の初期値，final exploration は $\varepsilon-$greedy 戦略における $\varepsilon$ の最終値，final exploration frame は $\varepsilon-$greedy 戦略において，$\varepsilon$ を線形に減少させた時に final exploration に到達するまでに必要なフレーム数である．

本実験では，DQN と DPN で共に文献 [1] による 3 層の畳み込み層と 2 層の全結合層を持つ DQN アーキテクチャを利用する．ただし，DPN では出力層の活性化関数として出力を 0.01 から 0.99 にクリッピングしたソフトマックス関数を利用する．

本実験では入力の画像のフレーム数を 1,2,4 枚の 3 種類で行われるが，文献 [1] の DQN アーキテクチャは，4 枚の画像を入力として扱っているため，1 枚と 2 枚の画像を扱う本実験ではそれぞれ異なるニューラルネットワークのハイパーパラメータを利用する．文献 [1] から変更するそれぞれのハイパーパラメータは Table 2 に示す．

DPN では，確率的勾配降下法として Adam [17] を利用する．また，DPN では正の報酬を観測したエピソードの経験のみリプレイメモリに保存するため，全経験をリプレイメモリに保存する DQN よりも扱う経験数が少ない．したがって，リプレイメモリの容量とネットワークの訓練を始めるためのリプレイメモリの最低保存量は DQN より小さい値を利用する．

それらのハイパーパラメータは Table 3 に示す．

### 4.3 実験結果と考察

本実験では，まず，割引率 $\lambda$ を変化させた場合の DPN の学習を確認するために，画像 4 枚を入力に利用した Pong で実験を行った．その $\lambda$ による DPN の性能比較の図を Fig. 3 に示す．また，画像 1, 2, 4 枚をそれぞれ入力に利用した Pong における，$\lambda = 0.99$ の DPN と DQN の比較の図をそれぞれ Fig. 4, Fig. 5, 及び Fig. 6 に示す．これらの全ての結果は横軸にトレーニングステップ数，縦軸に 10 万試行毎の評価で得られたスコアを 15 実験で平均化したスコアを示す．

Fig. 3 より，割引率は小さくするほど学習の立ち上がりが遅くなっていることが確認できる．これは，割引率が小さいほど報酬から遠いルールへの強化値も大きく減衰するため，Eq. (8) によって設定される $\bar{P}_\theta(a_k|s_t)$ と $P_\theta(a_k|s_t)$ との交差エントロピーも小さくなり，政策の改善がゆっくり進むからであると考えられる．この特性は，Profit Sharing でも同様に引き起こされ，割引率が小さいほど報酬から遠いルールの評価値の強化量は小さくなるために，行動確率の変化量も小さくなる．したがって，DPN は Profit Sharing の特性を上手く再現できていると考えられる．また，DPN と合理性定理の関係はまだ未知であるが，Pong では合理性定理を満たさない大きな割引率を利用しても十分に学習が出来ていることが確認できる．これは，本実験で用いた Pong は自らの行動に関わらず常に環境が変化するため，迂回経路が発生しにくい問題であったからであると考えられる．以上より，本実験では割引率は最も大きな値の 0.99 に設定した場合に学習が早くかつ最高得点を獲得できていたと考えられる．ただし，DPN に関しても迂回経路が強く強化される場合は十分に考えられる。更に，$\lambda = 0.99$ は立ち上がりは一番早いものの，最高スコア付近までの収束は $\lambda = 0.95$ のほうが早い結果が得られた．このことから，学習速度と学習精度の関係から，適切な割引率を設定する必要がある．また，合理性定理との関係性も踏まえた割引率に関する更なる追求が求められる．

Fig. 4 による入力画像 1 枚での速度情報無しでの学習では，DPN は DQN よりも遥かに上手く学習できていることが確認できる．これは，DPN が Profit Sharing と同様に非ブートストラップ手法であるからだと考えられる．ブートストラップ手法は POMDPs の下では上手く行動価値の推定が出来ないことに対して，非ブート

Table 1: Hyperparameter of $\varepsilon-$greedy

| Hyperparameter | Value |
|---|---|
| initial exploration | 1 |
| final exploration | 0.01 |
| final exploration frame | 250000 |
| evaluation exploration | 0.001 |

Table 2: Hyperparameter of the neural network architectures for the Atari experiments

| Hyperparameter | 1 frame image | 2 frame image |
|---|---|---|
| Convolution layer: channels | 8, 16, 16 | 16, 32, 32 |
| Hidden layer: units | 784, 128 | 1568, 256 |

Table 3: Hyperparameter for the DPN

| Hyperparameter | Value |
|---|---|
| learning rate for Adam | 0.0000625 |
| Adam's $\varepsilon-$greedy parameter | 0.00015 |
| replay memory capacity for DPN | 100000 |
| replay start size | 1000 |

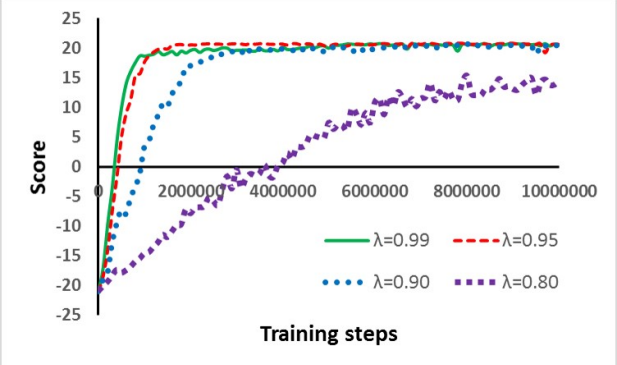


Fig. 3: Comparison of average Pong scores by the DPN with different discount factors

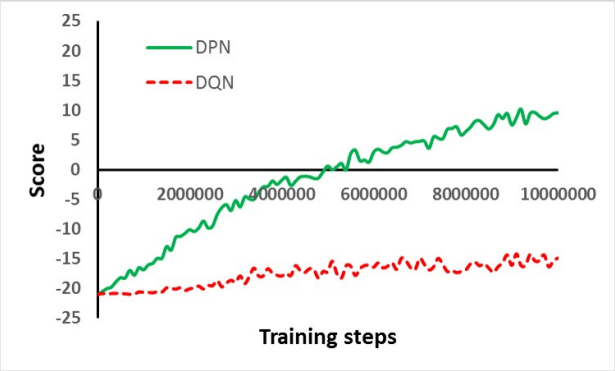


Fig. 4: Comparison of average Pong scores by the DQN and DPN using 1 frame image as input

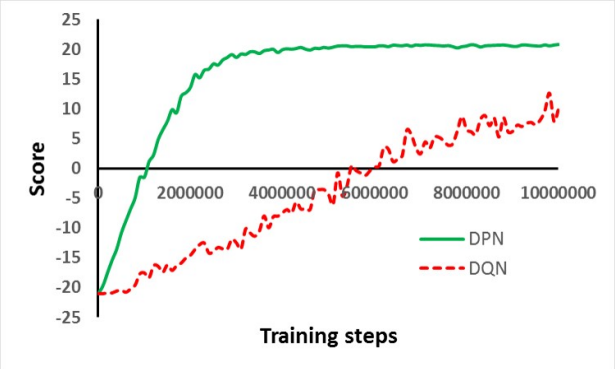


Fig. 5: Comparison of average Pong scores by the DQN and DPN using 2 frame image as input

ストラップ手法は POMDPs に頑強であるため，DPN は速度情報がない不完全知覚な Pong においても上手く学習できていると考えられる．

また，Fig. 5 による入力画像 2 枚での学習でも DPN は DQN よりも上手く学習できた．この実験では速度情報は完全に失ってはいないものの，画像 4 枚の場合よりは不完全な知覚となっているため，DQN の学習はなかなか進まなかったのではないかと考えられる．DPN はこの実験においては，300 万ステップほどで最高得点の +21 付近まで到達し収束している．これは，Profit Sharing の特性の学習の速さと POMDPs 環境に頑強である点を DPN が再現できているからではないかと考えられる．

最後に，Fig. 6 による入力画像 4 枚 (文献 [1] と同様の環境) での Pong の実験結果でも，DPN は DQN よりも素早く最高得点付近で収束できていることが確認できる．これらの結果から Profit Sharing の特性と同じように DPN による POMDPs に対する頑健さと学習の速さが確認され，有効性が示された．しかし，前述したとおり DPN の割引率の適切な設定は未だ未知である．したがって，より多くのタスクでの検証を行うことで，DPN が有効な問題の性質を確認する必要があると考えられる．

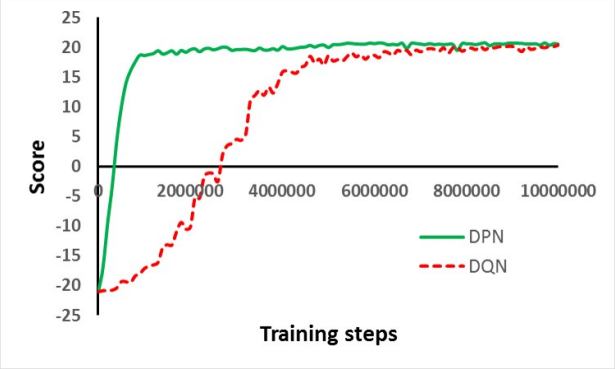


Fig. 6: Comparison of average Pong scores by the DQN and DPN using 4 frame image as input

## 5 結言

Q-learning は MDPs の下でのみ最適解の獲得が保証されている強化学習手法であるため，DQN も同様に POMDPs などの非 MDPs の下では学習保証が得られなかった．そこで，非ブートストラップ手法であることから POMDPs の下での学習が頑健であり，合理性定理によって迂回経路を抑制した学習が保証されている Profit Sharing が求められていた．しかし，Profit Sharing で扱う評価値は発散するため，DQN と同様な深層ニューラルネットワークを用いた評価値の近似は学習パラメータの発散を引き起こす問題があった．そこで本研究では，評価値ではなく行動確率を深層ニューラルネットワークで近似し，Profit Sharing アルゴリズムを参考にアルゴリズムを設計した Deep P-Network を提案した．本提案手法は非ブートストラップであることから POMDPs に頑健であり，その有効性はニューラルネットワークへの入力フレーム数を変化させることで POMDPs 環境とした Atari2600 の Pong によって DQN と比較、検証を行った．結果，提案手法は POMDPs と MDPs な Pong 環境で DQN よりも上手くそして素早く学習でき，新しい非ブートストラップ深層強化学習手法として有効性が示された．

しかし，本論文では提案手法は Pong でしか実験されていない．したがって，提案手法が有効な環境を知るためにはより多くのタスクでの実験が求められる．また，本提案手法と合理性定理の関係性はまだ未知であり，これらの関係性の追求が求められる．

## 参考文献

1) V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Ried-Miller, A.K. Fidjeland, G. Ostrovski, S. Pe-tersen, C. Beatie, A. Sadik, I. Antonoglou, H. King, D. Ku-maran, D. Wierstra, S. Legg, and D. Hassabis, “Human-Level Control through Deep Reinforcement Learning,” Nature, vol. 518, pp. 529–533, 2015.

2) C.J.H. Watkins and P. Dayan, “Technical Note: Q-Learning,” Machine Learning, vol. 8, pp. 55–68, 1992.

3) M.G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling, “The Arcade Learning Environment: An Evaluation Platform for General Agents,” arXiv preprint arXiv:1207.4708, 2012.

4) Hado Van Hasselt, Arthur Guez and David Silver, “Deep Reinforcement Learning with Double Q-learning,” Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, vol. 16, 2016.

5) Tom Schaul, John Quan, loannis Antonoglou and David Silver, “Prioritized Experience Replay,” arXiv preprint arXiv:1511.05952, 2015.

6) Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Haselt, Marc Lanctot and Nando de Freitas , “Dueling Network Architectures for Deep Reinforcement Learning,” arXiv preprint arXiv:1511.06581, 2015.

7) Marc G. Bellemare, Will Dabney and Remi Munos, “A Distributional Perspective on Reinforcement Learning,” arXiv preprint arXiv:1707.06887, 2017.

8) Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar and David Silver, “Rainbow: Combining Improvements in Deep Reinforcement Learning,” arXiv preprint arXiv:1710.02298, 2017.

9) S.D. Whitehead and D.H.Ballard, “Learning to perceive and act by trial and error,” Machine Learning, Vol.7, pp. 45—83, 1991.

10) J.J. Grefenstette, “Credit Assignment in Rule Discovery Systems Based on Genetic Algorithms,” Machine Learning, vol. 3, pp. 225–245, 1988.

11) 宮崎　和光・山村　雅幸・小林　重信,「強化学習における報酬割り当ての理論的考察」，人工知能学会誌，vol. 9, no. 4, pp.580–587, 1994.

12) 荒井　幸代・宮崎　和光・小林　重信,「マルチエージェント強化学習の方法論–Q-learning と Profit Sharing による接近–」，人工知能学会誌，vol. 13, pp. 609–618, 1998.

13) K. Miyazaki, “Exploitation-Oriented Learning with Deep Learning – Introducing Profit Sharing to a Deep Q-Network –,” Journal of Advanced Computational Intelligence and Intelligent Informatics, vol. 21, no. 5, pp. 849–855, 2017.

14) N. Kodama, K. Miyazaki, and T. Harada, “A Proposal for Reducing the Number of Trial-and-Error Searches for Deep Q-Networks Combined with Exploitation-Oriented Learning,” 2018 17th IEEE International Conference on Machine Learning and Applications, pp. 983–988, 2018.

15) Lin, Long-Ji., “Self-improving reactive agents based on reinforcement learning, planning and teaching,” Machine Learning, vol. 8, pp. 293–321, 1992.

16) K. Miyazaki and S. Kobayashi, “Exploitation-Oriented Learning PS-r#,” J. of Advanced Computational Intelligence and Intelligent Informatics, vol. 13, no. 6, pp. 624–630, 2009.

17) Diederik P. Kingma and Jimmy Ba “Adam: A Method for Stochastic Optimization,” arXiv preprint arXiv:1412.6980, 2014.