内部報酬を自動生成する強化学習による一人用RPGの自動攻略

加納 由希夫[1,a)]　鶴岡 慶雅[2]

概要：AIが内部報酬を自動生成することによって，外部報酬を利用しない自律的な強化学習を実現することは，報酬設計が困難であるような現実世界の問題に人工知能を応用させる上で非常に重要な課題の一つである．内部報酬を自動で生成する手法の一つにICM (Pathak, 2017) があり，A3C (Mnih, 2016) の報酬にICMの内部報酬を用いた強化学習は，VizDoomやSuper Mario Brosなどのゲームにおいて高い学習成果を示している．本研究では，ゲームの初期状態が毎回変化するという特徴を持つローグライクゲームに対して，ICMの手法を適用して効率的な強化学習を行えるようにすることを目指す．

Automatic capture of one person RPG by reinforcement learning to automatically generate internal compensation

Yukio Kano[1,a)]　Yoshimasa Tsuruoka[2]

Abstract: Realization of autonomous reinforcement learning that does not use external compensation by automatically generating internal compensation from AI is extremely important in applying artificial intelligence to real world problems where compensation design is difficult. ICM (Pathak, 2017) is one method to automatically generate internal compensation, reinforcement learning using internal compensation of ICM for remuneration of A3C (Mnih, 2016) is used in games such as VizDoom and Super Mario Bros. It shows high learning outcome. In this research, we aim to enable efficient reinforcement learning by applying ICM method to roguelike games, which features the initial state of the game changing every time.

1. はじめに

人工知能の研究分野の一つにゲームAIがある．ゲームは報酬や行動の制約などのルールが厳密に定められており，その評価も容易である．こうした特徴を持つゲームにおいて高い性能を発揮するゲームAIを開発することができれば，そのAIを構築する際に用いた様々な手法を応用することで，現実世界のより複雑な問題を解決する人工知能の開発に大きく貢献できると考えられる．

伝統的な二人零和有限確定完全情報ゲームである将棋やチェス，オセロなどのAIには，局面の良し悪しを判断する評価関数を利用してゲーム木を展開・探索していくようなアルゴリズムを用いているものがよく見られる．この評価関数の作成には，プロの対局データを用いた教師あり学習や，自己対戦を繰り返すような学習などを用いているものがある．例えば，1997年にオセロの世界チャンピオンに勝利したことで有名なBuroらによるLOGISTELLOでは，オセロの石の配置パターンの価値などについて，過去の様々な対局データや数万回の自己対戦データを用いて教師あり学習を行った [1].

より一般的なゲームAIには，強化学習の手法を用いて作られているものも多く存在する．強化学習では，エージェントが環境の状態に応じて選択した行動に対して報酬が与えられ，エージェントはこの報酬が最大となるような行動

[1] 東京大学工学部電子情報工学科
Department of Information and Communication Engineering, Graduate School of Information Science and Technology, The University of Tokyo
[2] 東京大学大学院情報理工学系研究科電子情報学専攻
Department of Information and Communication Engineering, The University of Tokyo
[a)] kano@logos.t.u-tokyo.ac.jp

選択戦略を学習していく．強化学習の代表的な手法としては，Q 学習 [2] や，Q 学習とニューラルネットワークを用いた深層学習を組み合わせた Deep Q Network (DQN) と呼ばれる手法などが挙げられる．Mnih らは DQN の手法を用いて，Atari 2600 系のゲームをクリアできる AI を作り上げている [3].

このようにゲーム AI には，教師あり学習や強化学習などといった機械学習の手法が用いられていることが多い．しかし，教師あり学習には，教師データを収集するコストや，十分な学習を行えるほどの教師データを用意できるとは限らないといった問題がある．一方，強化学習は教師データを必要としないが，学習の報酬設計が難しいという問題がある．

教師あり学習や強化学習における上記のような問題を回避するために，学習データや外部からの報酬を用いずに，ゲームのルールだけから自動的に学習を行えるようにしたい．Pathak らは，外部の環境から報酬が与えられない場合でも人は好奇心によって内発的に動機づけられて行動できることに注目し，その好奇心を内部報酬として定式化した．そして，その内部報酬を自動で生成する手法として Intrinsic Curiosity Module (ICM) を提案した [4]．ICM では，2 つの状態からその間で選択した行動を予想する逆モデルと，状態と選択した行動から次状態を予測する順モデルの 2 つのニューラルネットワークのモデルを同時に学習しながら，順モデルの予測が外れるほど大きくなるような内部報酬を生成する．Pathak らは，VizDoom と Super Mario Bros というゲームに対して，Mnih らによる A3C [5] の報酬に ICM の生成する内部報酬を用いた強化学習を行い，高い学習結果を示した．ICM の内部報酬を用いた強化学習では，AI にとって未知である場所を訪れるほど受け取る報酬が多くなるため，より奥へ進むことや迷路を探索することを目的としたゲームと相性が良いと考えられる．

Pathak らは，ICM の学習において，数ステップ行動したらゲームの最初の状態に戻って学習を繰り返すという手法を用いていた．本研究では，ゲームの最初の状態が毎回ランダムな場合においても，ICM による学習が行えるような手法を提案することが目的である．そこで，そのような特徴をもつゲームであるローグライクゲームに ICM の手法を適用することを目指す．

2. 関連研究

2.1 強化学習

機械学習の手法の一つに強化学習 [6] がある．標準的な強化学習のモデルでは，ある環境におかれたエージェントが，現在の状態を観測し，与えられた選択肢から行動を一つ選択し，それに応じた報酬を受取る．そして，一連の行動によって受け取る報酬の総和が最も大きくなるような方策を試行錯誤しながら学習する．

同じ機械学習の手法の一つである教師あり学習と比較すると，強化学習は教師データを用意する必要がないという利点があるが，明確な指示がない状態で自ら試行錯誤しなければならないため学習にかかる時間は大きくなる．また強化学習には，人間が最適だと判断していた解よりも優れた解を発見する可能性があるという利点もある一方で，人間にとって合理的だとは言えない解に辿り着いてしまう可能性も考えなければならない．そのため，扱う問題に応じて適切な学習方法を選択する必要があると考えられる．

2.2 Q 学習

代表的な強化学習の手法としては，Q 学習 [2] が挙げられる．Q 学習では，まず環境を有限マルコフ決定過程 (finite Markov decision process, finite MDP) でモデル化しなければならない．有限マルコフ決定過程とは，以下の 4 つの要素で表されるモデルである．

- 状態の有限集合：$S = \{s_1, s_2, ..., s_m\}$
- 行動の有限集合：$A = \{a_1, a_2, ..., a_n\}$
- 遷移関数：状態 s で行動 a を選択したときに状態 s' に遷移する確率 $P(s, a, s')$
- 報酬関数：状態 s で行動 a をとったときの報酬 $R(s, a)$

Q 学習の目的は，現在の状態が与えられたときに，取るべき行動を決定する方策を求めることである．ここでは，各状態 s に対して行動 a を与える $\pi(s) = a$ を方策として表現するものとする．この方策は，受け取る報酬の総和を最大化するようなものであってほしいが，一連の行動にかかる時間が非常に長くなってしまうのも避けたい．そこで目的関数として次のようなものを定義する．

$$Q^{\pi}(s, a) = E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) | s_0 = s\right]$$

ただし，γ $(0 < \gamma < 1)$ は割引率と呼ばれる定数である．これは，状態 s から方策 π を行った場合の報酬の総和を示している．ここで，最適化された方策を π^* とすれば，この式は次式のように書き直せる．

$$Q^{\pi^*}(s, a) = R(s, \pi^*(s)) + \gamma \max \sum_{s'} P(s, a, s') Q^{\pi^*}(s', \pi^*(s'))$$

このような，最適化された方策をとったときの行動価値関数 $Q^{\pi^*}(s, a)$ を，関数 $Q(s, a)$ で近似することによって学習を行うのが Q 学習であり，以下のような式で更新していく．

$$Q(s_t, a_t) \longleftarrow Q(s_t, a_t) + \alpha_t \left(R(s_t, a_t) + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right)$$

ここで，α_t は学習率とよばれ，この学習率 α_t が以下の 2 つの式を満たすとき，関数 $Q(s, a)$ は必ず $Q^{\pi^*}(s, a)$ に収束することが証明されている．

$$\sum_{t=0}^{\infty} \alpha_t = \infty \ , \quad \sum_{t=0}^{\infty} {\alpha_t}^2 < \infty$$

2.3 ニューラルネットワークと深層学習

ニューラルネットワーク (neural network, NN) とは，人間の脳を模倣して計算機上で表現したものである．人間の脳内ではニューロンという神経細胞が多数存在し，各ニューロンがシナプスという接合部位によってがっており，電気信号を伝達している．以下の図 1 に，このニューロンをモデル化したものの例を示す．

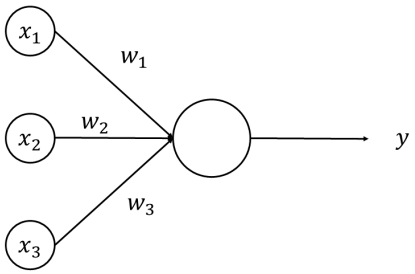

図 1 ニューロンのモデルの例

図 1 のニューロンでは，各入力を x_i，重みを w_i としており，出力が y である．この出力 y は活性化関数 φ を用いて，次のような式で計算される．b はバイアス値である．

$$y = \varphi\left(\sum_i x_i w_i + b\right)$$

活性化関数には様々なものがあるが，例を挙げればシグモイド関数や tanh 関数，ReLU 関数などがある．これらの関数は非線形関数であり，各ニューロンの出力に非線形性を持たせる役割がある．

- シグモイド関数：$\varphi(x) = \dfrac{1}{1+e^{-x}}$
- tanh 関数：$\varphi(x) = \tanh(x)$
- ReLU 関数：$\varphi(x) = \max(0, x)$
- softmax 関数：$\varphi(x_i) = \dfrac{e^{x_i}}{\sum_k e^{x_k}}$

このようなニューロンを接続することにより，ニューラルネットワークを構築することができる．図 2 にニューラルネットワークのモデルを示す．ニューラルネットワークは，まずはじめに入力層を持ち，その後ろに何層かの隠れ層があり，最後に出力層を持つという構造をしているものが多い．この図の場合，出力層を除く全てのノードが，次の層のすべてのノードに繋がっている．このような層のことを全結合層と言う．

ニューラルネットワークを用いることで，任意の連続関数を任意の精度で近似できることが分かっている [7]．一部の不連続関数も，部分的には連続関数と見なせるならば，ニューラルネットワークで近似することができるだろう．次に，ニューラルネットワークがどのように学習をし，目的の関数に近づくのかを説明する．ここでは，教師あり学習を例にとって説明する．教師あり学習は，様々な入力 $\boldsymbol{x_t}$ とそれに対する正しい出力 $\boldsymbol{y_t}$ が記されたトレーニングデータセットを用いて行われる．ある $\boldsymbol{x} \in \boldsymbol{x_t}$ をニューラルネットワークに入力して得られた出力 $\boldsymbol{y}$ とデータセットの出力データ $\boldsymbol{y_t}$ から，最小二乗法やクロスエントロピーなどを用いて目的関数を定め，その目的関数の最適化を行う．最適化には，誤差逆伝播法 [8] によってニューラルネットワーク中の重み w やバイアス値 b を更新していく手法が広く用いられている．

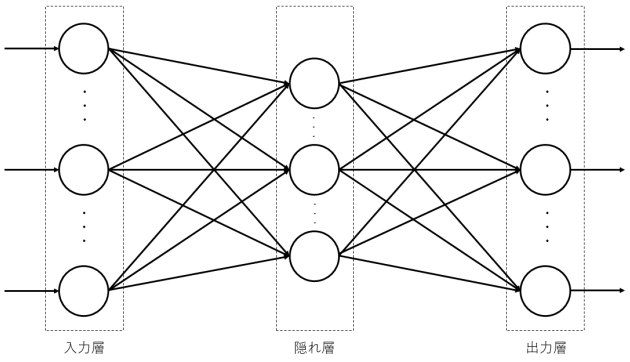

図 2 ニューラルネットワークの例

2.4 Deep Q-Network (DQN)

Deep Q-Network (DQN) [9] は，Q 学習の行動価値関数 $Q(s,a)$ をニューラルネットワークを用いて学習・近似する手法である．DQN では，ニューラルネットワークの重みを θ_i とし行動価値関数を $Q_{\theta_i}(s,a)$ と表現する．誤差関数は次のようにかける．

$$L_i(\theta_i) = E\left[\left(R(s,a) + \gamma \max_{a'} Q_{\theta_{i-1}}(s',a') - Q_{\theta_i}(s,a)\right)^2\right]$$

ただし，$Q_{\theta_{i-1}}(s',a')$ の θ_{i-1} とは，一つ前の重み θ を使って計算することを意味する．この誤差関数を微分して得られる次の式を用いて誤差逆伝播法を行うことで，ニューラルネットワークを学習していく．

$$\nabla_{\theta_i} L_i(\theta_i) =$$
$$E\left[\left(R(s,a) + \gamma \max_{a'} Q_{\theta_{i-1}}(s',a') - Q_{\theta_i}(s,a)\right) \nabla_{\theta_i} Q_{\theta_i}(s,a)\right]$$

2.5 Intrinsic Curiosity Module (ICM)

Intrinsic Curiosity Module (ICM) [4] とは，強化学習の際に用いる報酬をエージェント内部で自動生成するための手法である．ICM は主に 2 つのニューラルネットワークのモデルからなる．

一つは，逆モデル (inverse dynamics model) である．エージェントは環境から状態 s_t, s_{t+1} を観測する．次に，これらの状態をニューラルネットワークの畳み込み層など

を利用して，特徴ベクトル $\phi(s_t), \phi(s_{t+1})$ へ変換する．状態 s_t から s_{t+1} には，行動 a_t によって遷移するが，この行動 a_t を予想するのがこの逆モデルの役割である．このモデルは次のようにかける．

$$\hat{a}_t = g\left(s_t, s_{t+1}; \theta_I\right)$$

ただし，g はニューラルネットワークの学習機能 (learning function) であり，θ_I はニューラルネットワークのパラメータ，$\hat{a}_t$ が予想された行動である．このニューラルネットワークは，次のような最適化を行うように学習する．

$$\min_{\theta_I} L_I\left(\hat{a}_t, a_t\right)$$

L_I は，実際の行動と予想された行動の不一致度を示す損失関数である．

もう一つは，順モデル (forward dynamics model) である．これは，エージェントが観測した状態 $\phi(s_t)$ と選択した行動 a_t から，次状態 $\phi(s_{t+1})$ を予測するモデルである．このニューラルネットワークのモデルは次のようにかける．

$$\hat{\phi}(s_{t+1}) = f\left(\phi(s_t), a_t; \theta_F\right)$$

逆モデルと同様に，f はニューラルネットワークの学習機能 (learning function) であり，θ_F はニューラルネットワークのパラメータ，$\hat{\phi}(s_{t+1})$ が予測された次状態の特徴ベクトルである．このニューラルネットワークは，次のような最適化を行うように学習する．

$$\min_{\theta_F} L_F\left(\phi(s_t), \hat{\phi}(s_{t+1})\right) = \min_{\theta_F} \frac{1}{2}\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|^2$$

この時，この順モデルは内部報酬の信号として，次式のように計算される r_t^i を発する．

$$r_t^i = \frac{\eta}{2}\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|^2$$

ただし，$\eta > 0$ である．これは，次状態が予測できないほどエージェントが受け取る内部報酬の量が多くなることを意味している．この内部報酬によって動くエージェントは，人間が好奇心から行動するように，より予測のつかない未知のエリアに進むような行動を選択するように学習する．

ICM では，これら 2 つのモデルを同時に学習させる．図 3 に，ICM の構造の簡略図を示す．逆モデルを学習することによって，環境から観測された状態 s_t を特徴ベクトル $\phi(s_t)$ に変換する際の特徴変換法として最適だと思われる方法を習得し，それにより順モデルの学習効率を向上させている．Pathak らによれば，順モデルのみを利用した場合よりも，逆モデルと順モデルを同時に利用した場合のほうが，VizDoom と Super Mario Bros. のゲームにおいて，かなり高い学習効率を示すことが分かっている [4].

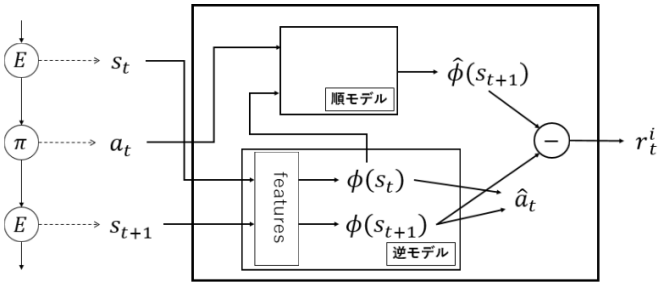

図 3 ICM の構造

2.6 Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) [10] とは，全結合層だけでなく，畳み込み層 (convolution layer) とプーリング層 (pooling layer) と呼ばれる 2 つの層をセットで用いるニューラルネットワークである．例えば画像を入力する場合を考えれば，畳み込みとプーリングのセットの繰り返しにより画像の特徴を次々に抽出していくことができる．このような特徴から，CNN は画像の多クラス分類などによく使われる．

CNN では，まず入力された特徴マップ (画像を考えると分かりやすい) の周辺を数回 0 で埋めるという，ゼロパディング (zero padding) と呼ばれる操作を行うことが多い．その例を図 4 に示す．この場合では，元の特徴マップの周辺 1 周分を 0 でパディングしている．ゼロパディングの利点としては，端のデータに対する畳み込み回数が増えるため端の特徴もしっかり考慮されるようになる点，カーネルのサイズや層の数を調整できる点などが挙げられる．カーネル (kernel) というのは，畳込み演算で用いる窓のことである．フィルタ (filter) とも呼ばれる．畳み込み層では，特徴マップ上をカーネルという窓をずらしながら畳み込み演算を行っていく．この時，カーネルをずらす大きさをストライド (stride) という．CNN では，このカーネルのパラメーターを自動的に学習していき，各畳み込み層で様々な特徴を抽出することができるようになっていく．図 5 と図 6 に，この畳み込み演算操作の例を示す．

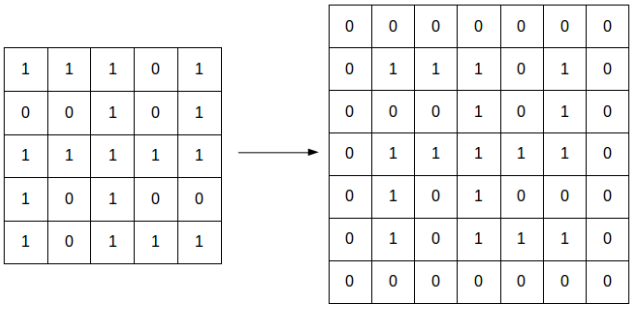

図 4 ゼロパディングの例

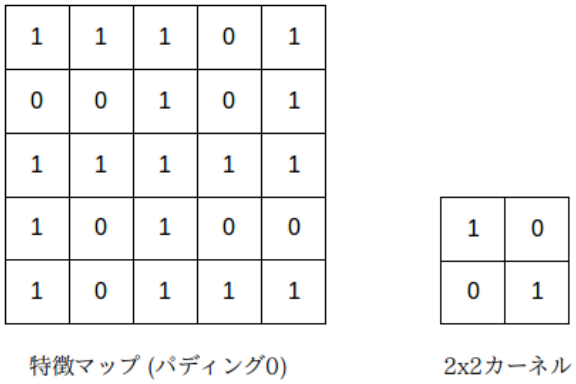

図 5 カーネルの例

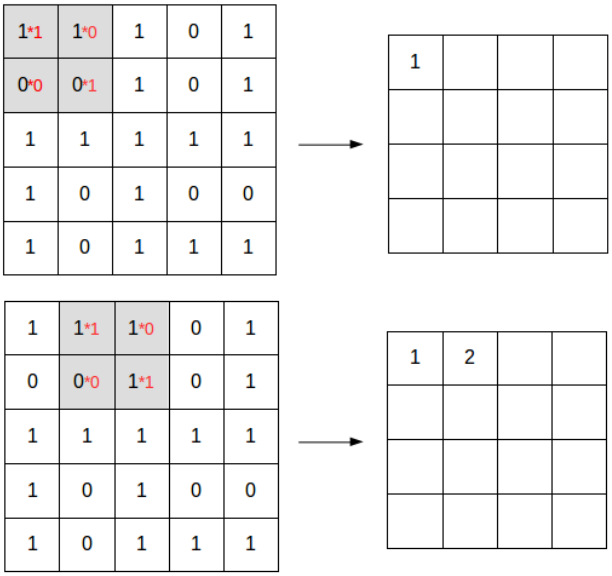

図 6 ストライド 1 の時の操作と出力の例

入力特徴マップの高さと幅を I_h, I_w，ゼロパディング幅を P，ストライドの大きさを S，カーネルの高さと幅を K_h, K_w とするとき，畳み込み層で出力される特徴マップの高さと幅 O_h, O_w は次のようになる．

$$O_h = \frac{I_h + 2P - K_h}{S} + 1$$

$$O_w = \frac{I_w + 2P - K_w}{S} + 1$$

畳み込み層の出力は次にプーリング層に入る．プーリング層で行われるのは，主に情報の圧縮 (down sampling) である．この操作の利点には，計算コストを下げる，過学習をある程度抑制する，様々な大きさの入力に対して同じ大きさの出力を返すなどが挙げられる．図 7 に，図 6 の操作の出力を，プーリングによって圧縮した結果を示す．プーリングに用いたカーネルサイズは 2x2，ストライドを 2 とする．それぞれのカーネルは，その領域内でもっとも大きい数値を出力する．今回例に上げた特徴マップ (図 5, 左) は点対称であったため，プーリングの出力も点対称になっている．

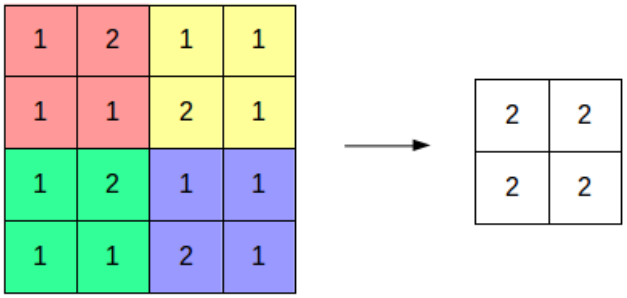

図 7 プーリングの例

3. 提案手法

Pathak らは VizDoom と Super Mario Bros のゲームにおいて ICM の学習を行ったが，それらのゲームではゲームの初期状態 (マップの形) が常に一定であり，数ステップ学習したら一度初期状態に戻ってから学習を繰り返すという方法をとっていた．しかし，同じ初期状態に何度も戻るような方法は，現実の問題に応用しにくいと考えられる．そこで本研究では，ゲームの初期状態が毎回異なる場合でも，外部報酬をほぼ与えずに内部報酬を自動生成する強化学習で効率的にゲームを攻略できるように，ICM の手法を拡張して適応することを提案する．

ゲームの初期状態が毎回異なるゲームとして，本研究では一人用 RPG ゲームの一種であるローグライクゲームを提案する．ローグライクゲームとは，“Rogue” という 1980 年に公表されたゲームと同様の特徴を持っているコンピュータ RPG の総称である．ローグライクゲームでは，ダンジョンと呼ばれる迷路の地形が自動生成され，敵モンスターやアイテムなどがランダムに配置される．プレイヤーが操作する主人公は，これらのモンスターをうまく対処しながら，アイテムを集めたりダンジョンを進んだりしながらゲームを攻略していく．

本研究では，行動やモンスター，アイテム，ルールなどがより単純化されたローグライクゲームを用意し，内部報酬による強化学習を行うものとする．

4. 実験

4.1 実験概要

GPW では，ローグライクゲームへ ICM を拡張させる前提として，まずは ICM のモデルを実装し，正しく学習が進むかどうかを確認することを目的とした．

ICM の順モデルと逆モデルは，機械学習ライブラリの一種である TensorFlow を用いて実装した．逆モデルの特

徴抽出の部分には CNN を用い，それぞれのモデルの出力層には全結合のニューラルネットワークを用いた．

ICM における順モデルと逆モデルの同時学習は，次のような最適化を行うことで実現する．

$$\min_{\theta_I,\theta_F}[(1-\beta)L_I+\beta L_F]$$

L_I や L_F は 2.5 項で説明した関数であり，これらを定数 β で重み付けしたものに対して最適化を行う．

4.2 実験方法

本実験では，実装した ICM モデルを用いて，以下のような実験環境において実際に学習を行った．

- 100×100 行列 A を用意し，その第 (i,j) 成分を，$i+j$ または $|i-j|$ が 5 で割り切れる時は 1，それ以外は 0 と定義する．
- 100×100 の大きさのマップ M を用意し，M 上の点 (x,y) を行列 A の第 (x,y) 成分と対応させる．$(1 \leq x,y \leq 100)$
- 点 (a,b) に存在するエージェントは $a-k \leq x \leq a+k, b-k \leq y \leq b+k$ の $(2k+1)\times(2k+1)$ の範囲を観測する．
- エージェントは，観測結果を $(2k+1)^2$ のサイズのベクトル $v[(2k+1)^2]$ に保管する．
- エージェントは，1 ステップごとに $(x,y) \rightarrow (x+d_x, y+d_y)$ のように移動する．d_x, d_y は，それぞれ $[-1,0,1]$ の中から等確率に選択する．つまり，合計 9 つの選択肢から 1 つを選んで移動する．(どちらも 0 の場合は移動しない) 9 つの行動の選択肢は，9 次元の特徴ベクトルで表現する．
- エージェントが移動したら，ICM には移動前の周辺データ $v_i[(2k+1)^2]$ と移動後の周辺データ $v_o[(2k+1)^2]$ と移動した方向の特徴ベクトル $a[9]$ が入力され，学習が行われる．
- 初め，エージェントは M 上の点 $(50,50)$ に存在し，1 ステップごとに上記の操作を繰り返しながら移動する．30 ステップ目が終了したら，また点 $(50,50)$ に戻り繰り返し学習を続ける．

図 8 は，この実験環境モデルを簡単に説明したものである．0 と 1 が規則的に並んでいるマップ M 上にエージェントが存在する．中央の点がエージェントの位置であり，赤い太枠で囲われている部分がエージェントの観測する範囲である (図は $k=2$ の場合)．周りの赤い 8 本の矢印が，エージェントが移動する際の方向である．

次に，ICM の学習に用いたニューラルネットワークのモデルの詳細を説明する．

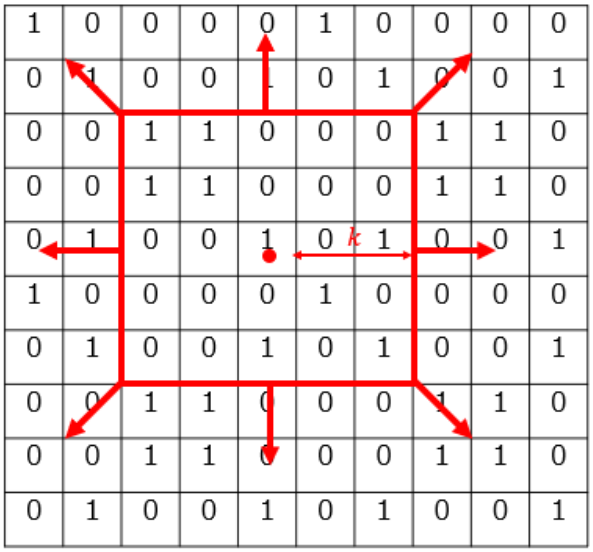

図 8 実験環境モデル

- 逆モデルに用いた CNN は，2 つのレイヤーからなる．第 1 レイヤーでは，5 × 5 のサイズのカーネルを用いて 32 の特徴を計算し出力する．第 2 レイヤーでは，これら 32 の特徴を受け取り，それらそれぞれについて 5 × 5 のサイズのカーネルを用いて 64 の特徴を計算し出力する．
- CNN の全出力は，まず ReLU 関数を活性化関数とする第 1 の全結合層を通り，第 2 の全結合層へ出力される．第 2 の全結合層は逆モデルと順モデルそれぞれで分かれており，逆モデルでは softmax 関数を活性化関数とする全結合層を，順モデルでは sigmoid 関数を活性化関数とする全結合層を用いる．
- L_I には逆モデルの出力と正解の出力とのクロスエントロピーを用いる．L_F は，順モデルの出力と正解の出力の各成分の差の 2 乗の平均を用い，$[(1-\beta)L_I+\beta L_F]$ の最適化を誤差逆伝搬法を用いて行う．誤差逆伝搬法における最急降下法の学習率は 0.0001 とした．

4.3 実験結果・考察

今回の実験では，エージェントの観測範囲を $k=8$ とし，重み付けの定数は $\beta=0.2$ とした．学習結果を図 9 に示す．

グラフの横軸は 200 ステップごとに計算した L_I と L_F，そして $L=(1-\beta)L_I+\beta L_F$ の大きさを対数で表しており，横軸は学習ステップ回数を表している．このグラフからは，学習ステップを積み重ねるほど L_F と L_I がともに振動的ではあるが全体として 0 に近づいていき，L の最適化が正しく行われていることが分かる．L_I については，ほぼ 0 と計算されているところがあり，これが L_I のグラフの一部が負の無限大に発散している理由である．

以上のことから，ICM のモデルの学習が順モデル，逆モデルともに，正しく行われていると判断できる．

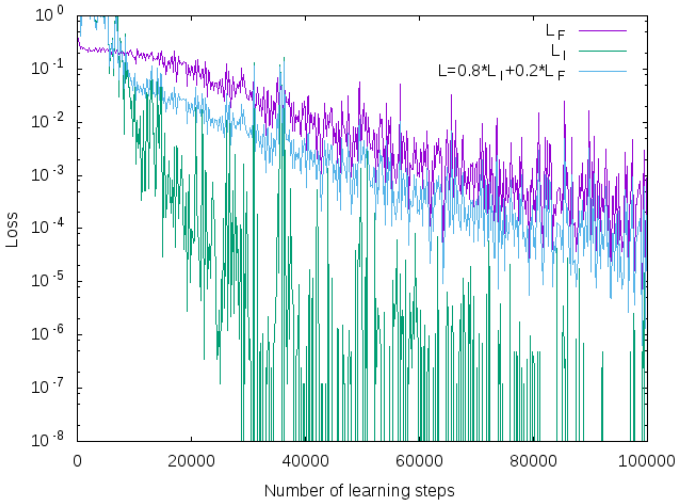

図 9 実験結果

5. おわりに

本稿では，実装した ICM のモデルについて，正しく学習が行えていることを示した．本研究の目的は，この ICM をローグライクゲームの学習に適用することであるため，今後の課題としては，まずは ICM の生成する内部報酬を用いた Q 学習や DQN による学習について簡単な実験を行い動作を確認すること，そして実際にローグライクゲームに対して ICM を用いた強化学習の学習効率の向上を目指すことが挙げられる．

参考文献

[1] Buro, M.: Logistello: A strong learning othello program, *19th Annual Conference Gesellschaft für Klassifikation eV (GfKl)*, Vol. 2, Citeseer (1995).

[2] Watkins, C. J. and Dayan, P.: Q-learning, *Machine learning*, Vol. 8, No. 3-4, pp. 279–292 (1992).

[3] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D. and Riedmiller, M. A.: Playing Atari with Deep Reinforcement Learning, *NIPS Deep Learning Workshop* (2013).

[4] Pathak, D., Agrawal, P., Efros, A. A. and Darrell, T.: Curiosity-driven Exploration by Self-supervised Prediction, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017* (Precup, D. and Teh, Y. W., eds.), Proceedings of Machine Learning Research, Vol. 70, PMLR, pp. 2778–2787 (online), available from ⟨http://proceedings.mlr.press/v70/pathak17a.html⟩ (2017).

[5] Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D. and Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning, *International Conference on Machine Learning*, pp. 1928–1937 (2016).

[6] Kaelbling, L. P., Littman, M. L. and Moore, A. W.: Reinforcement learning: A survey, *Journal of artificial intelligence research*, Vol. 4, pp. 237–285 (1996).

[7] Cybenko, G.: Approximation by superpositions of a sigmoidal function, *Mathematics of Control, Signals, and Systems (MCSS)*, Vol. 2, No. 4, pp. 303–314 (1989).

[8] Rumelhart, D. E., Hinton, G. E., Williams, R. J. et al.: Learning representations by back-propagating errors, *Cognitive modeling*, Vol. 5, No. 3, p. 1 (1988).

[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] LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P.: Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278–2324 (1998).