NICO2AI #12
強化学習II

妹尾 卓磨

目次

- **方策勾配**

方策を直接求めるには

- **深層強化学習**

DNNによる強化学習によって何ができるようになったか

- **内発的動機**

生物は環境からの報酬のみから学習しているのか

方策勾配

目次

- **方策勾配**

方策を直接求めるには

- **深層強化学習**

DNNによる強化学習によって何ができるようになったか

- **内発的動機**

生物は環境からの報酬のみから学習しているのか

方策ベースと価値ベース

- **方策ベース (policy based) ← 今日はこの話**

今の方策で収益を計算して収益を上げる方策を学習する

方策自体を求める

- **価値ベース (value based)**

状態や行動に価値を設定して価値を基に方策を決定する

方策自体は求めない

価値ベース強化学習の課題

- **連続行動空間の扱い (continuous action space)**

Q学習ではQ関数は状態と行動を引数とした関数であった

→**行動は離散的な値 (discrete action space)**

行動がモーターの制御値など連続的場合だと…

→Q(s, 0.1), Q(s, 0.01), Q(s, 0.001), …

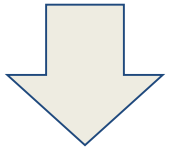
無限通りの行動について価値を評価する必要がある

方策勾配 (policy gradient)

方策を確率的なモデルとして直接推定する

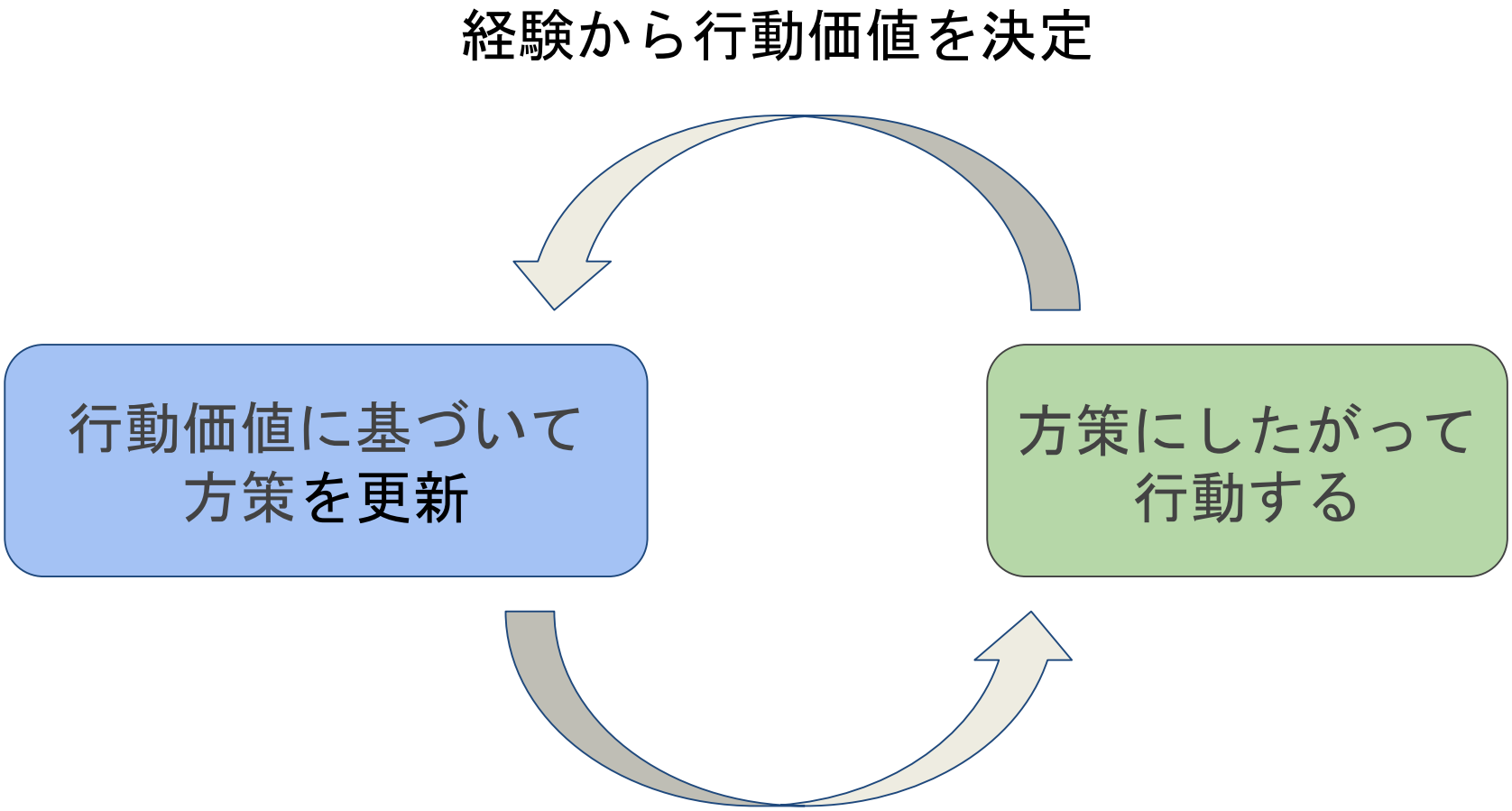
方策πをθでパラメタライズして**勾配法**で最適解を求める

$$\theta \leftarrow \theta + \delta\theta$$



方策πのパラメータを直接求めて**連続的な行動を扱う**

方策ベースの学習フロー



方策勾配の学習ステップ

- 方策πにしたがって行動

方策は**確率的**に表される。例えばsoftmax関数など。

$$\pi_\theta(a|s) = \frac{exp(\theta_{sa})}{\sum_{b \in A} exp(\theta_{sb})}$$

- 方策πの評価

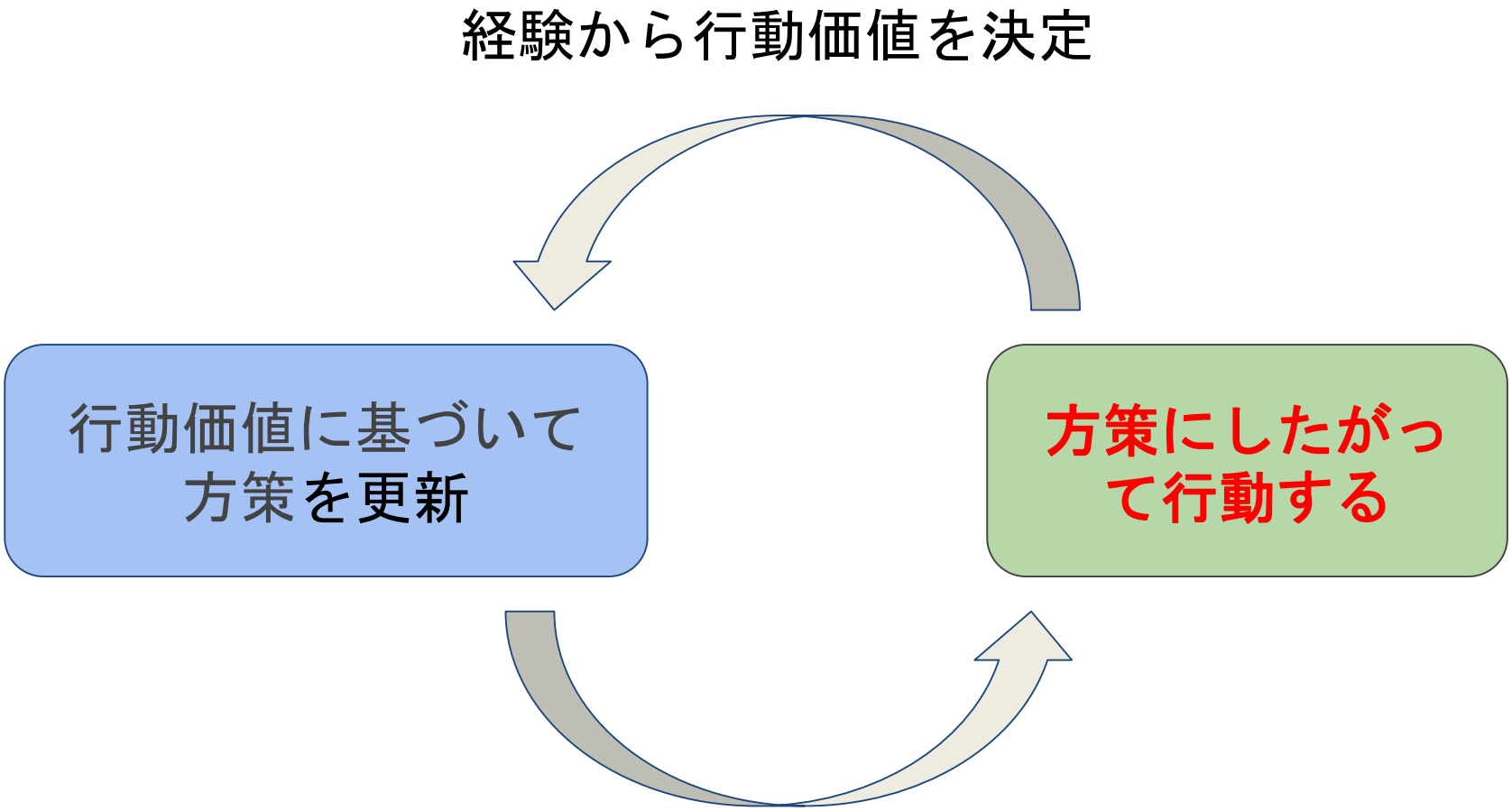
得られた**収益（または平均報酬）**で目的関数を定義

$$J(\theta) = \mathbb{E}\{\sum_{t=1}^{\infty} \gamma^{t-1} r_t | s_0, \theta\}$$

- 方策πの更新

目的関数を最大化するように**勾配法**で方策を更新

方策ベースの学習フロー



方策の決定

方策πは各行動の選択確率を表す確率関数である

- **行動が離散的な場合**

ソフトマックス関数などで確率分布を表す

$$\pi_\theta(a|s) = \frac{exp(\theta_{sa})}{\sum_{b \in A} exp(\theta_{sb})}$$

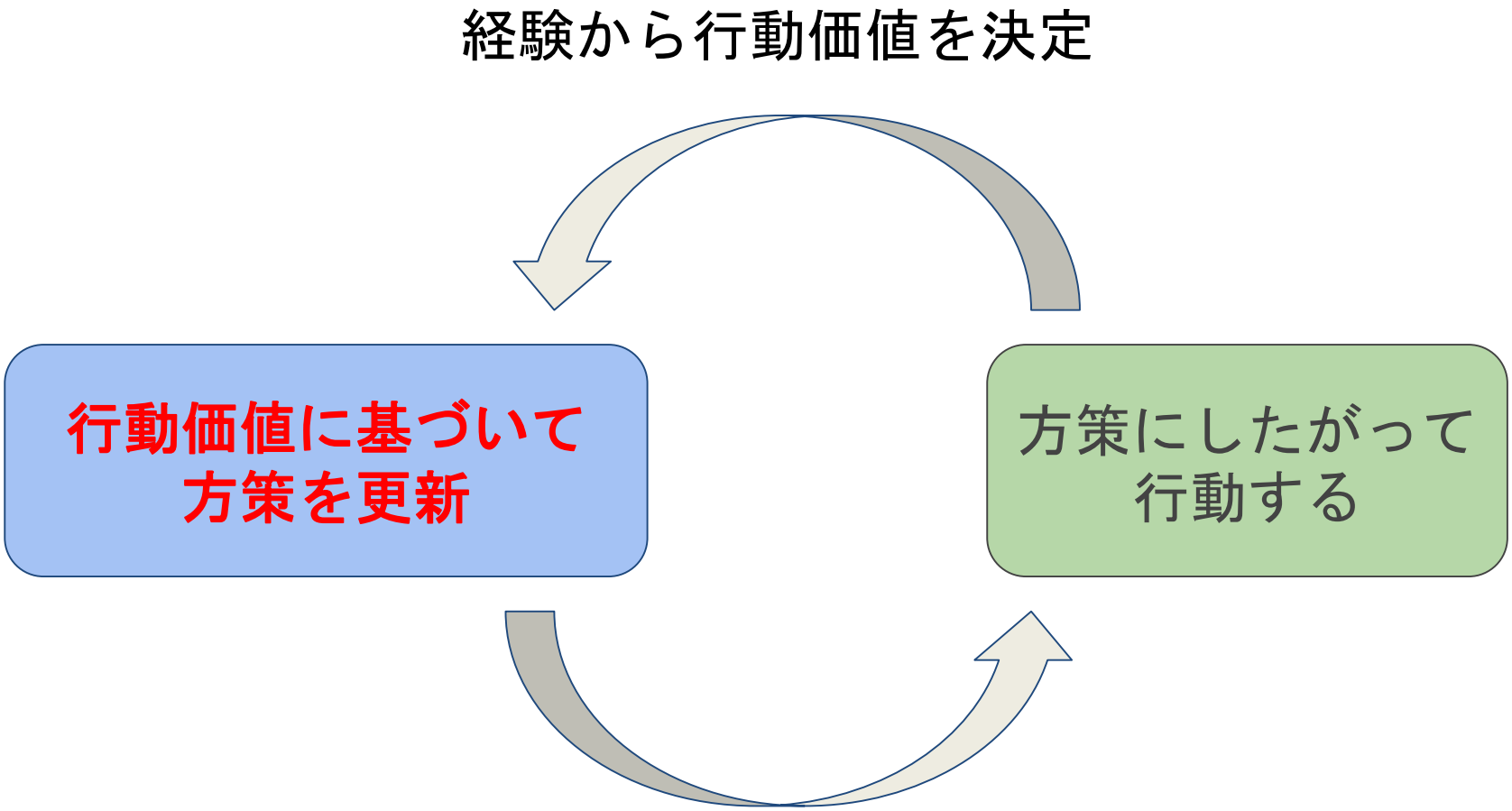
- **行動が連続的な場合**

正規分布等の確率分布で表す。以下ではW,Cをパラメタライズ

$$\pi_\theta(a|s) = \frac{1}{(2\pi)^{d_a/2}|C|^{1/2}} \exp(-\frac{1}{2}(a - Ws)^T C^{-1}(a - Ws))$$

行動の次元数　共分散行列　行動の次元数×状態の次元数の行列

方策ベースの学習フロー



方策勾配定理

勾配を行動価値関数Q^{π}を用いて以下のように表せる

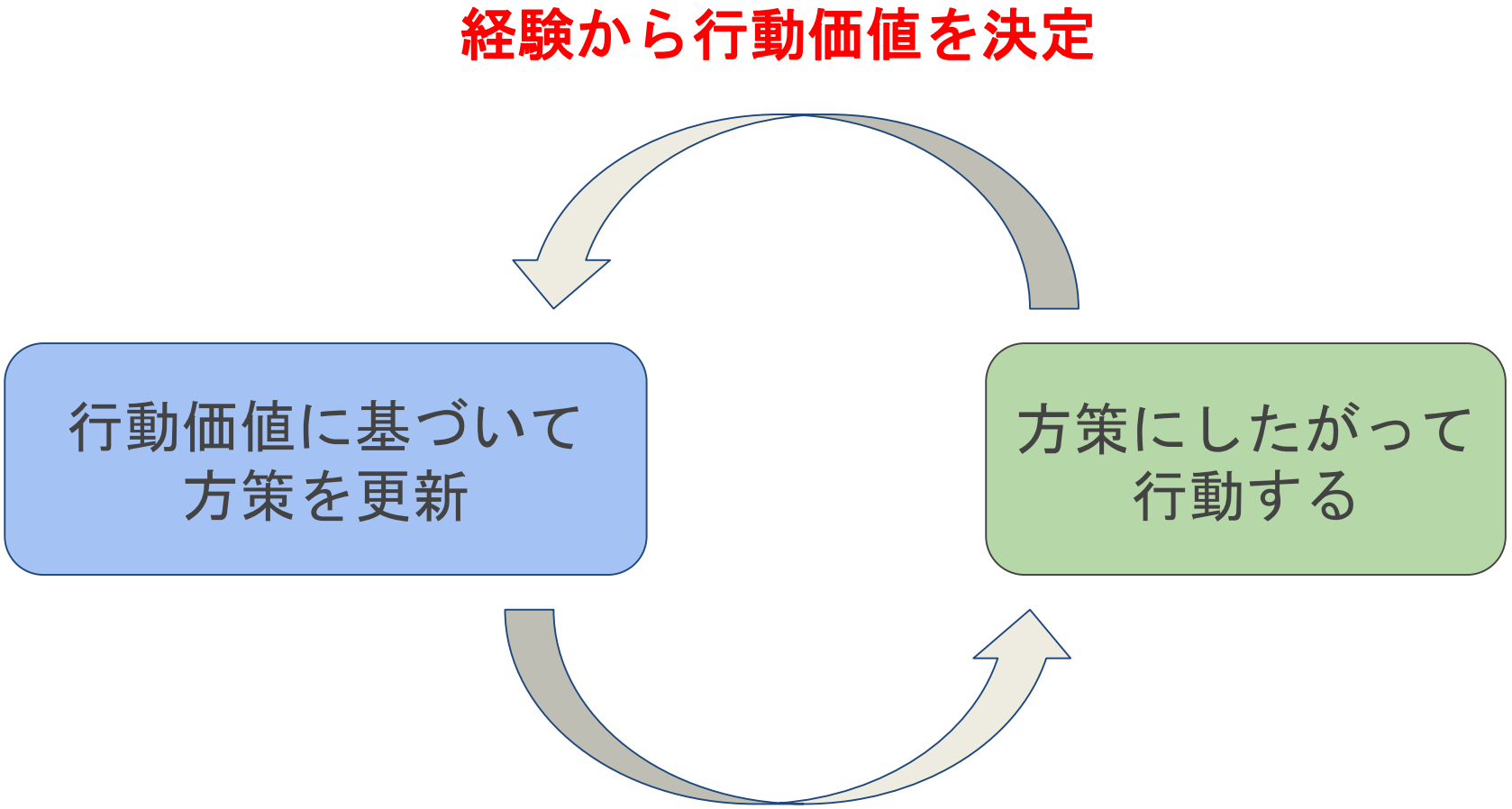
$$Q^{\pi}(s,a) = \mathbb{E}[\sum_{k=1}^{\infty} \gamma^{k-1} r_{t+k} | s_t, a_t, \theta]$$

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi\theta}[\frac{\partial \pi_{\theta}(a|s)}{\partial \theta} \frac{1}{\pi_{\theta}(a|s)} Q^{\pi}(s,a)]$$

$$= \mathbb{E}_{\pi\theta}[\nabla_{\theta} \log \pi_{\theta}(a|s) Q^{\pi}(s,a)]$$

これが方策勾配定理である

方策ベースの学習フロー



行動価値の決定方法

実際には行動価値関数は未知である

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi\theta}[\nabla_\theta \log \pi_\theta(a|s) Q^\pi(s,a)]$$

行動価値関数を推定して学習を行う2つの手法を紹介する

- **REINFORCE**
- **Actor-Critic**

REINFORCE

行動価値関数を**即時報酬**で近似する

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi\theta}[\nabla_\theta \log \pi_\theta(a|s)(r_t - b)]$$

分散を抑えるための定数

方策が確率的なためベースラインで**分散を抑える**

主にベースラインには平均報酬が使われる

$$b = \frac{1}{MT} \sum_{m=1}^{M} \sum_{t=1}^{T} r_{t,m}$$

エピソード数

ベースラインを含めてもいい理由

$$\mathbb{E}[b\nabla_\theta \log \pi(a|s)] = b\sum_a \frac{\partial \pi(a|s)}{\partial \theta}$$

$$= b\frac{\partial}{\partial \theta}\sum_a \pi(a|s)$$

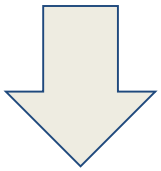
期待値には影響を与えない

$$= b\frac{\partial}{\partial \theta}1 = 0$$

REINFORCEの弱点

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi\theta}[\nabla_\theta \log \pi_\theta(a|s)(r_t - b)]$$

即時報酬で近似しているので簡単に実装が可能



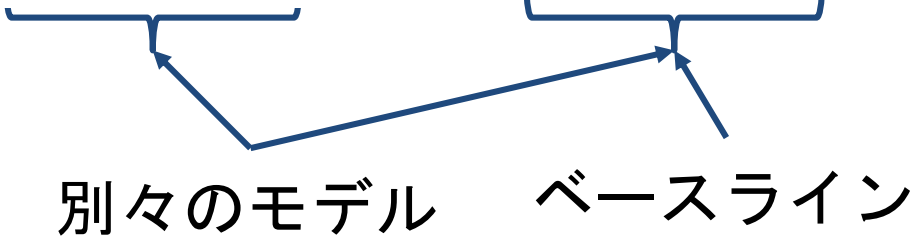
しかし即時報酬だけだと**遅れて発生する報酬**を扱えない

Actor-Critic

方策関数と価値関数を**別々**に求める

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi\theta}[\nabla_\theta \log \pi_\theta(a|s)(Q^\pi(s,a) - V^\pi(s))]$$

$$= \mathbb{E}_{\pi\theta}[\nabla_\theta \log \pi_\theta(a|s)(R_t - V^\pi(s))]$$



行動選択を行う**Actor**と価値を推定する**Critic**で構成

Actor: Criticの推定する価値より高くなる行動を学習
Critic: Actorの経験の価値を正確に推定するように学習

Actor-Criticの優れているところ

方策ベースと価値ベースのいいとこ取り

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi\theta}[\nabla_\theta \log \pi_\theta(a|s)(R_t - V^\pi(s))]$$

- **方策ベースのメリット**

ブートストラップ（推定値を使って更新）なしで学習ができる

- **価値ベースのメリット**

収益の推定を行うことで最適な方策を学習できるようにした

方策勾配

- **方策ベースの手法**

方策自体を確率分布として求めることで連続的な行動も扱える

- **方策勾配定理**

現在の方策で得られる**行動価値を最大化**するように勾配を計算して、方策を更新する

- **行動価値の計算**
  - **REINFORCE**
    即時報酬で近似して、ベースラインで分散を下げる
  - **Actor-Critic**
    価値関数と方策を**別々のモデル**で学習して
    より最適な方策を学習できる

深層強化学習

目次

- **方策勾配法**

方策を直接求めるには

- **深層強化学習**

DNNによる強化学習によって何ができるようになったか

- **内発的動機**

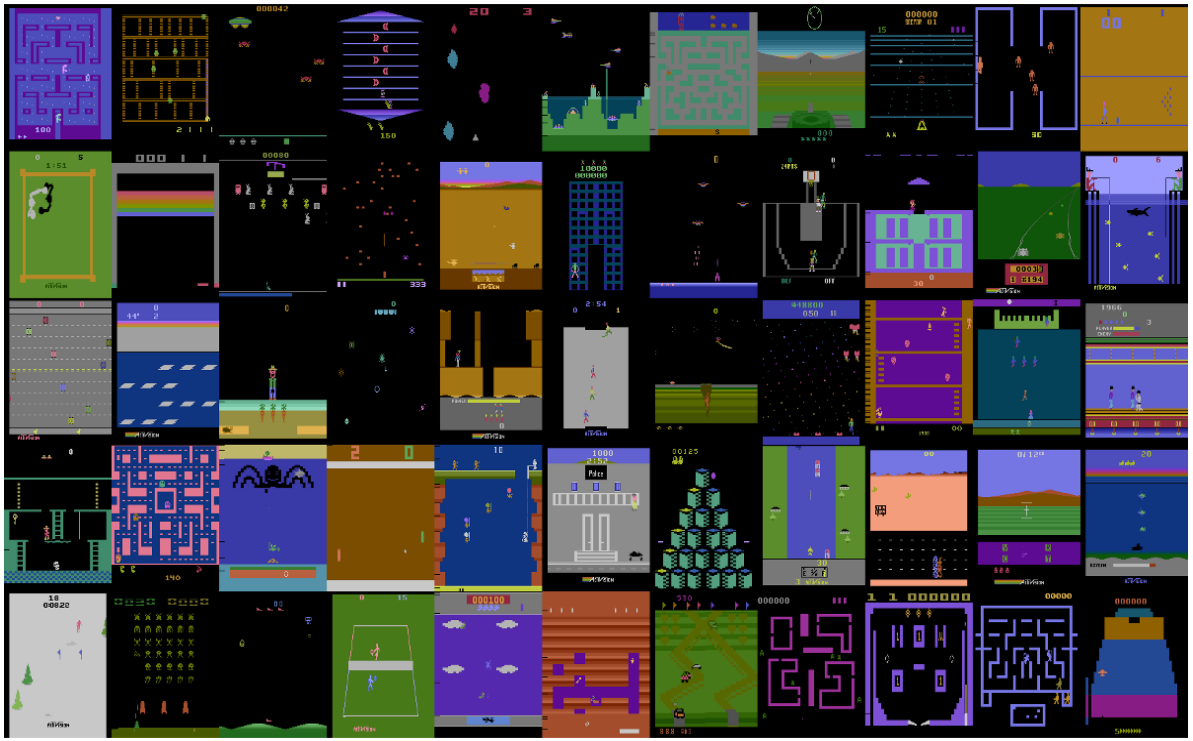
生物は環境からの報酬のみから学習しているのか

Arcade Learning Environment (ALE)

- **The Arcade Learning Environment: An Evaluation Platform for General Agents [Bellemare+ 2013]**

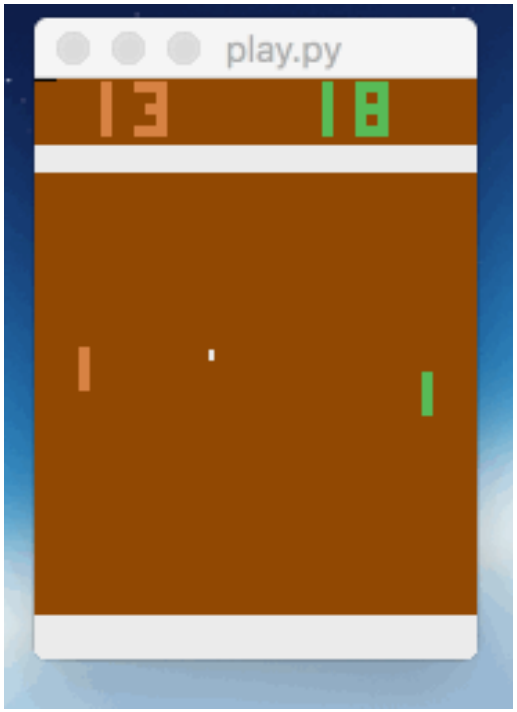
Atari 2600 という米アタリ社のゲーム機のエミュレータ

AI研究のためのフレームワーク



Atariのゲームの例

Pong

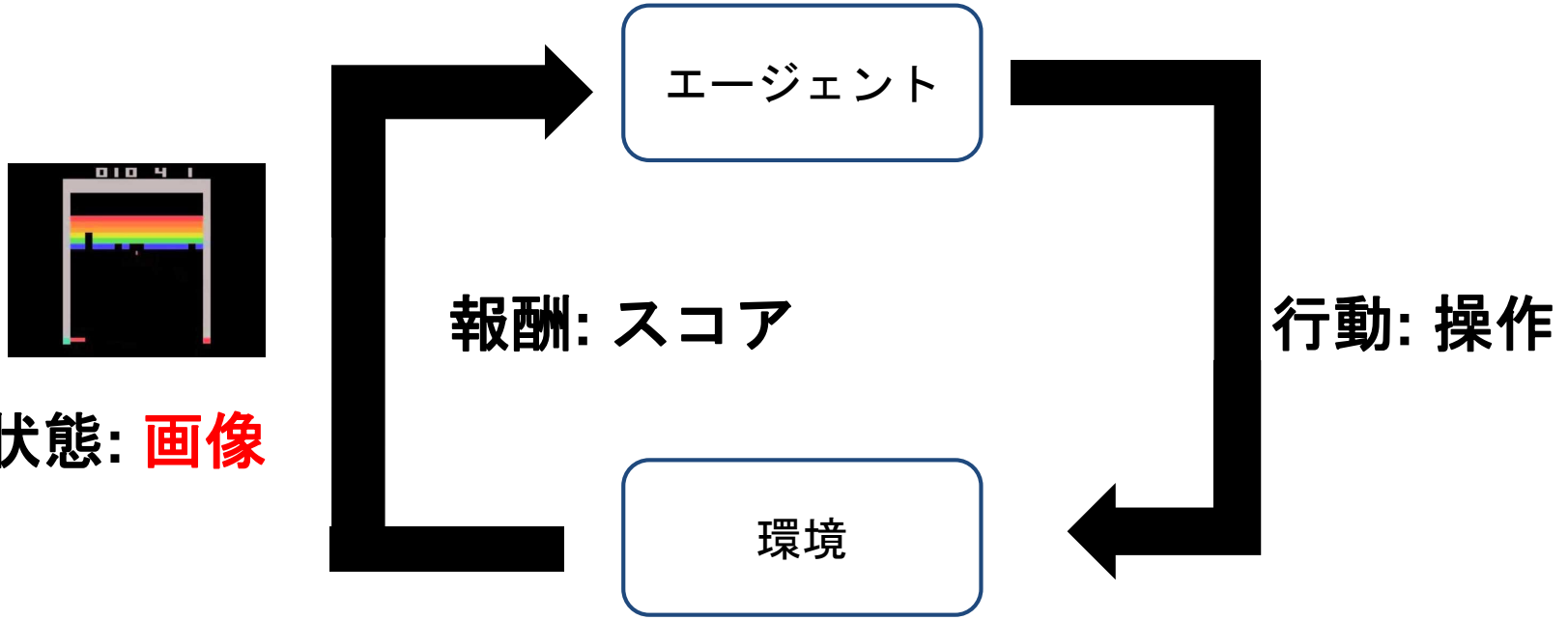


Breakout

Space Invaders

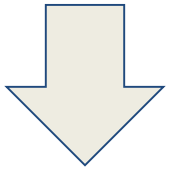
ALEにおける強化学習の設定

エージェントはエミュレータから**ゲーム画像**とスコアを受け取り、エミュレータにゲームの操作を返す



ALEが掲げていた課題

提供している様々なゲームに適応できるような手法の発明

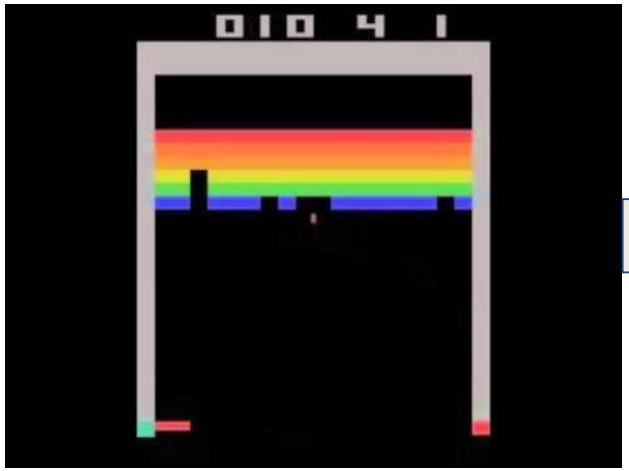


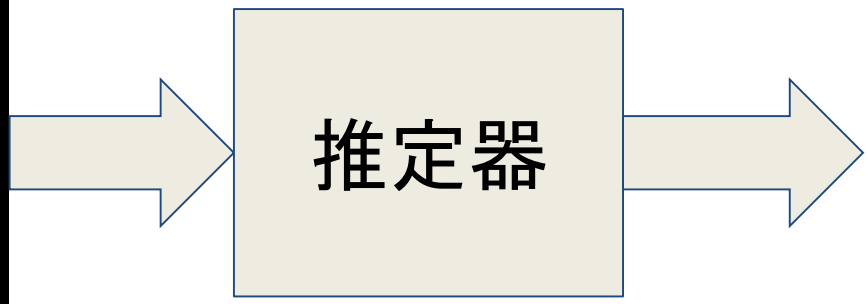
敵の位置や自分の位置の情報など

ゲームに特有の特徴量を使わない

理想的な方法

直接ゲーム画像から価値（Q値）を推定する





価値

テーブルでQ関数を表せないとき

入力状態は84x84の白黒画像

テーブルでQ関数を表すには256^{84x84}の状態の行が必要

Q関数をテーブルではなく関数で近似してみる

$$Q(s,a) \approx \theta_s^T \phi_s(s) + \theta_a^T \phi_a(a)$$

ベクトルを前処理する関数

線形関数で近似した結果

人間のスコアを以下のモデルと比較

- **Linear:** （前処理なし）画像から直接線形近似したモデル
- **HNeat:** （前処理あり）物体の位置が与えられているモデル

人間に勝てない...

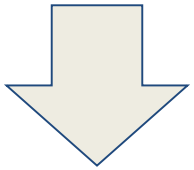
	Breakout	B. Rider	Enduro	Sequest	S. Invaders
Human	31	**7456**	**368**	**28010**	**3690**
Linear*	3	2347	62	657	301
HNeat	**52**	3616	106	920	1720

*後述の工夫を行なって全結合層1つで近似したモデル

どうすれば勝てるのか

ゲームに勝つために画像から直接

有用な特徴量を見つけることができれば勝てるはず



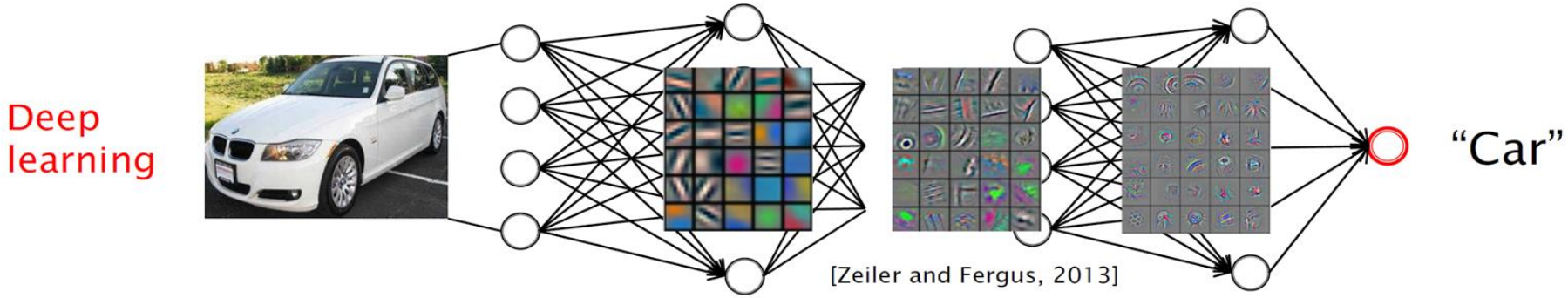
Deep Neural Network

DNNによる特徴抽出

層を重ねて、入力から推測までをend-to-endで学習すると
タスクに有用な特徴抽出ができる

人間が特徴量を設計する必要がなくなった

（画像ではCNNを使うのが一般的）



学習によって得られた特徴表現

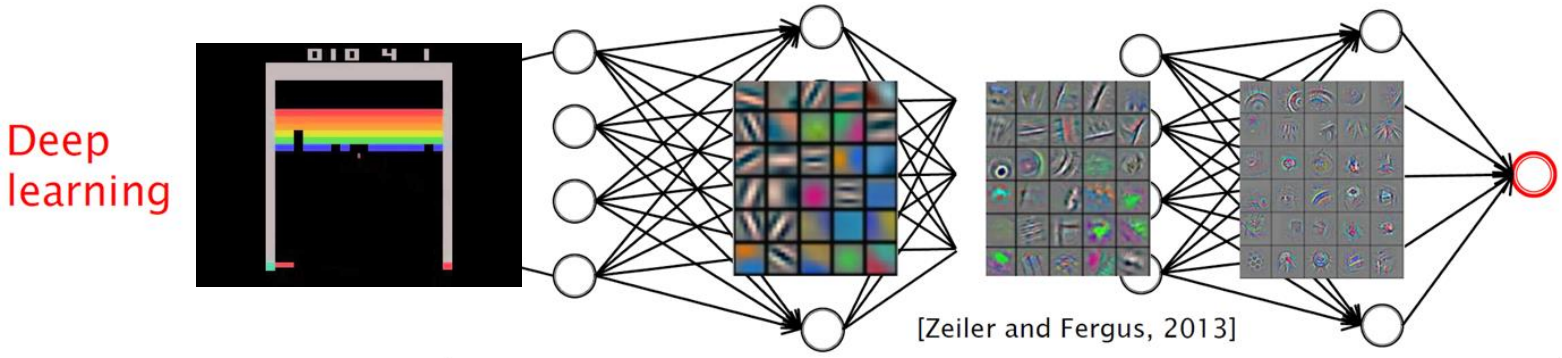
https://www.slideshare.net/nlab_utokyo/deep-learning-49182466

DQN (Deep Q-Network)とは

- Playing Atari with Deep Reinforcement Learning [Mnih+ 2013]
- Human-level control through deep reinforcement learning [Mnih+ 2015]

Q関数をDNNで近似した

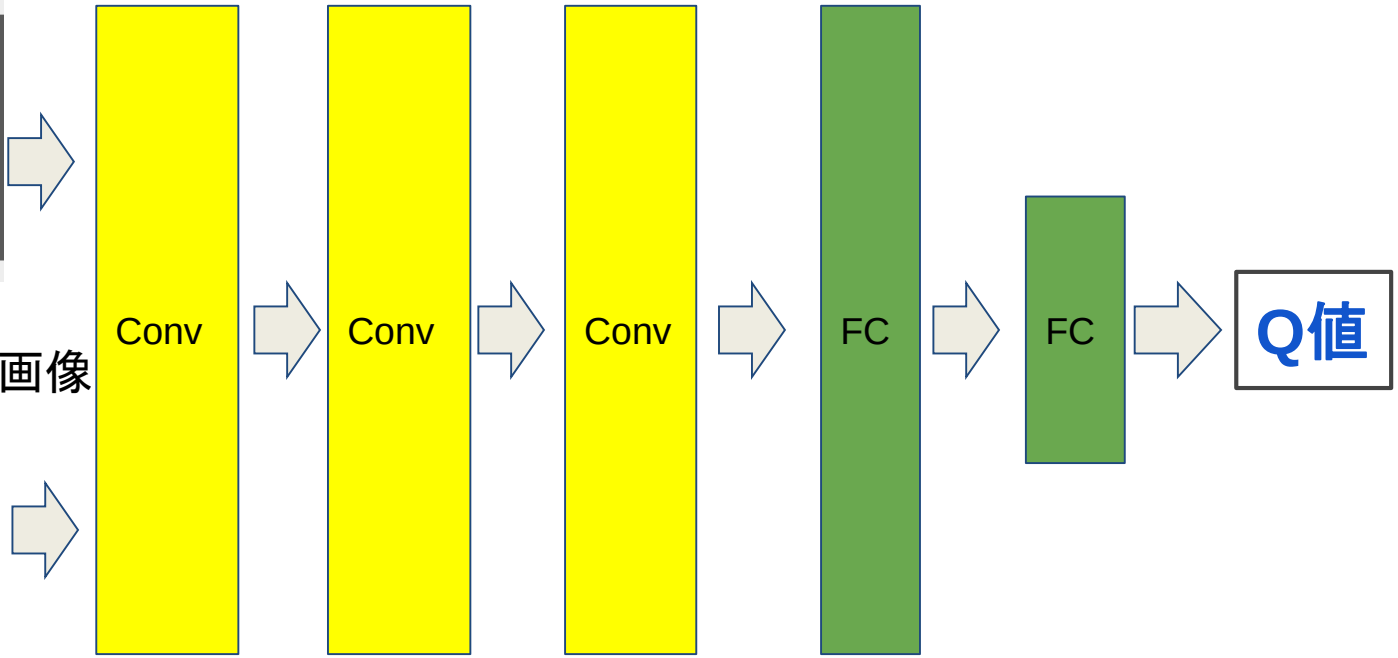
$$Q(s_t, a) \leftarrow Q(s_t, a) + \alpha \left[r_{t+1} + \gamma \max_p Q(s_{t+1}, p) - Q(s_t, a) \right]$$



Q関数をDNNで表すとは

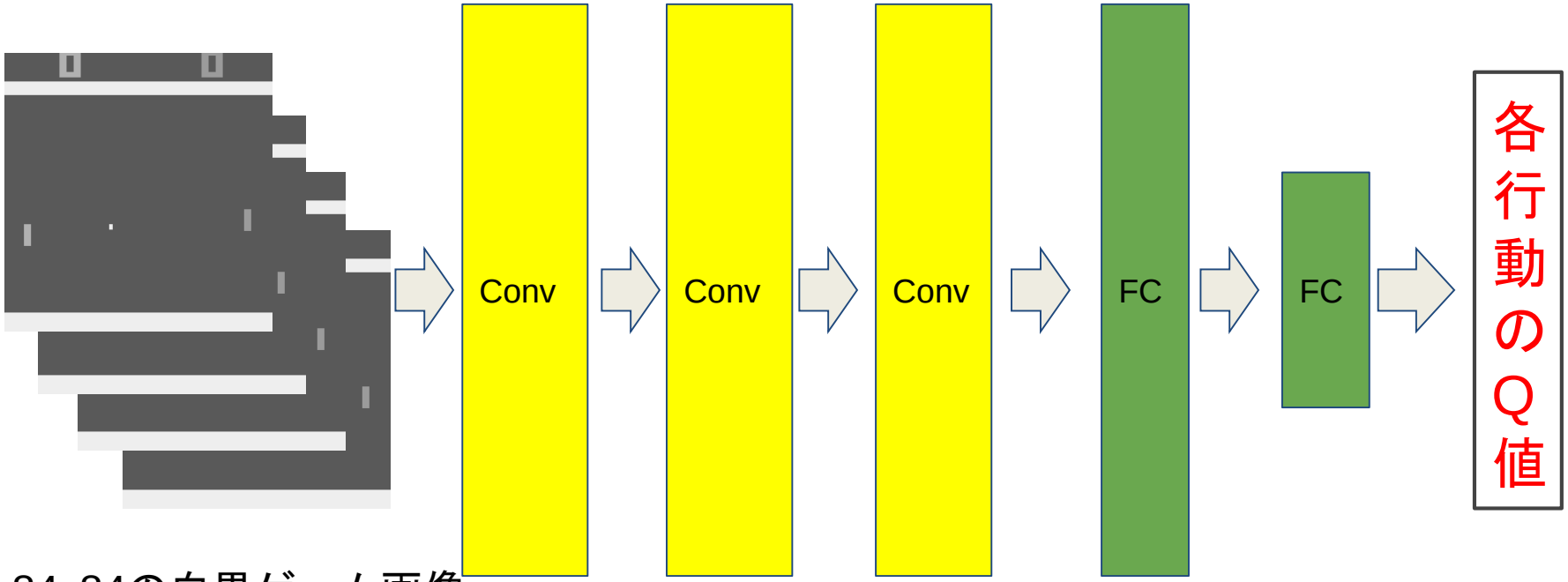
84x84の白黒ゲーム画像

行動a？



行動全てに対してこの計算をするのはコストがかかる

DQNのネットワーク構成



各行動に対してQ値を計算するよりも
一回の計算で全てのQ値を計算するように構成

しかしQ関数をDNNにしただけだと…

さっきの線形モデルの手法に負けている….

	Breakout	R. Raid	Enduro	Sequest	S. Invaders
Naive DQN	**3.2**	1453.0	29.1	275.8	**302.0**
Linear	3.0	**2346.9**	**62.0**	**656.9**	301.3

DQNの工夫

DQNはQ関数を**ディープにしただけではない**

- **Experience Replay**
  保存した経験からランダムに学習に使用する
- **Freezing the Target Network**
  TD誤差の目標値に古いネットワークを使用する
- **Clipping Rewards**
  スコアのスケールを統一する
- **Skipping Frames**
  数フレームごとに行動を選択する

DQNの工夫: Experience Replay

Reinforcement Learning for Robots Using Neural Network [Lin+ 1993] で最初にExperience Replayが提案されている

- 経験の相関性が高いとNNが過学習してしまう

過去の経験をメモリーに溜めて
ランダムに経験を選んで学習に使用する

- 価値を前の状態に伝搬させるには多く学習する必要がある

経験をミニバッチで学習させることで加速！

- いい経験を一回しか使わないのは勿体無い

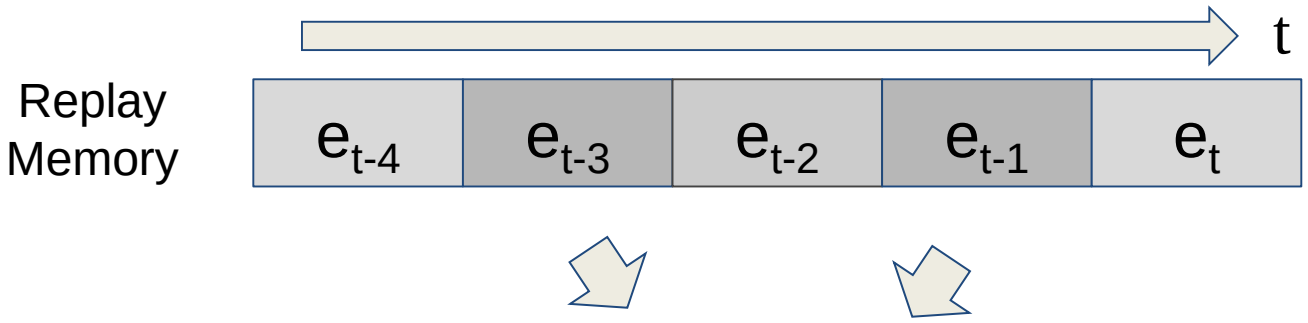
過去の経験を使いまわそう！

DQNの工夫: Experience Replay

昔経験したこと$e_t=(s_t, a_t, r_t, s_{t+1})$をメモリーに貯めて

ランダムに複数（n=32）選んで学習に使うことで

学習を**高速**且つ**安定させた**



DQNの工夫: Freezing the Target Network

ここを古いDNNにする

$$Q(s_t, a) \leftarrow Q(s_t, a) + \alpha \left[r_{t+1} + \gamma \max_p Q(s_{t+1}, p) - Q(s_t, a) \right]$$

TD誤差の目標計算に予測値を使っている

目標値の変動に伴うDNNで表現されたQ関数の
学習不安定性を解消するために

- 誤差計算を行うときの目標のDNNを古いもので固定
- 一定周期（10000step毎）で現在のDNNと同期

DQNの工夫: Clipping Rewards

ゲームによってスコアの大きさが異なると
ハイパーパラメータを変える必要がある

学習率
割引率
などなど

例

- Pongでは一回点を取ると1点もらえる
- Space Invadersでは倒したインベーダの場所に応じて10~30点

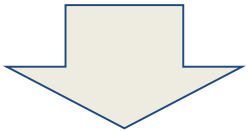
様々なゲームに同じハイパーパラメータで対応するために、

- 負のスコアは-1
- 正のスコアは+1

で統一する

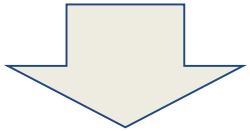
DQNの工夫: Skipping Frames

ALEは60fpsで描画されるが
必ずしも毎フレームで行動選択を行う必要がない



4フレーム毎に前の4フレームを使って行動選択を行う

4フレーム後まで同じ行動を繰り返し選択する
(Space Invaders では3フレーム)



行動選択のコストが減り、多くの経験を積むことができる

DQNと線形モデルの比較

ついに勝った…

	Breakout	R. Raid	Enduro	Sequest	S. Invaders
DQN	**316.8**	**7446.6**	**1006.3**	**2894.4**	**1088.9**
Naive DQN	3.2	1453.0	29.1	275.8	302.0
Linear	3.0	2346.9	62.0	656.9	301.3

Experience ReplayとTarget networkの影響

Experience Replay がスコアの向上にもっとも寄与していた

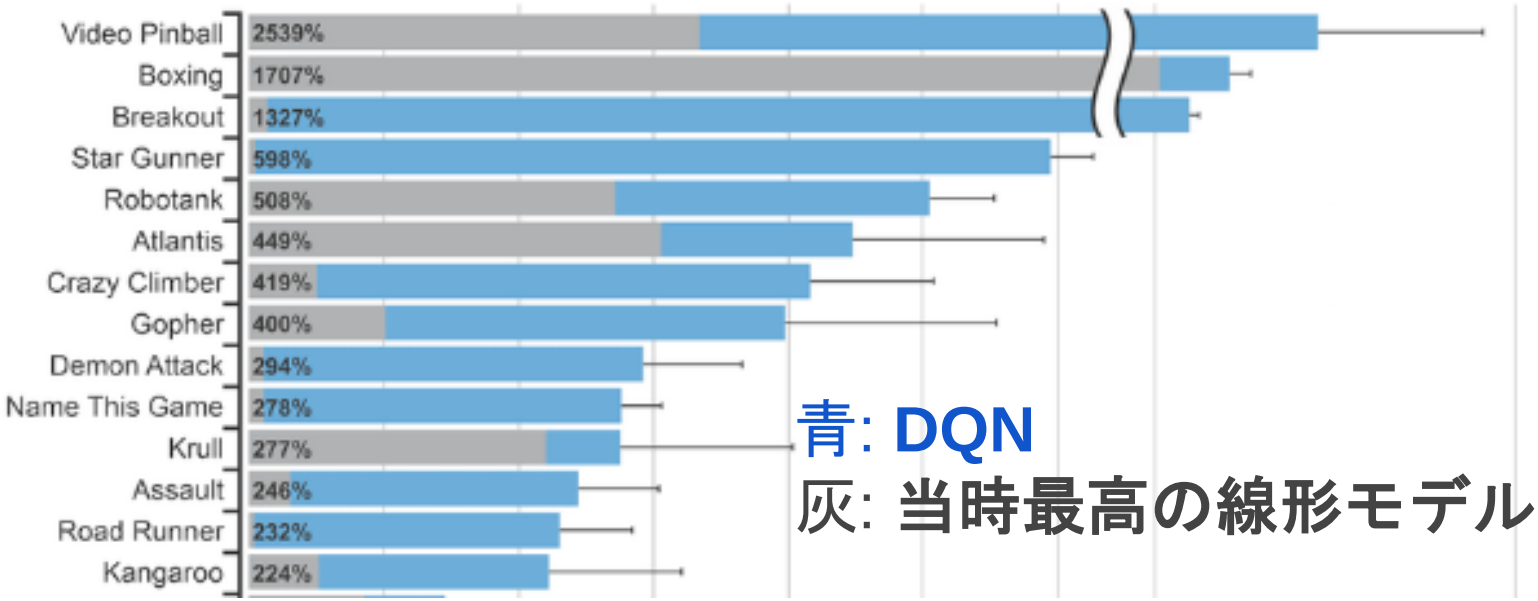
Replay	○	○	×	×
Target	○	×	○	×
Breakout	**316.8**	240.7	10.2	3.2
River Raid	**7446.6**	4102.8	2867.7	1453.0
Seaquest	**2894.4**	822.6	1003.0	275.8
Space Invaders	**1088.9**	826.3	373.2	302.0

Atari 2600 のゲームで学習した結果を人間のスコアと比較

人間より何%高いスコアが取れているかを示している

人間のプレイヤーは熟練者で

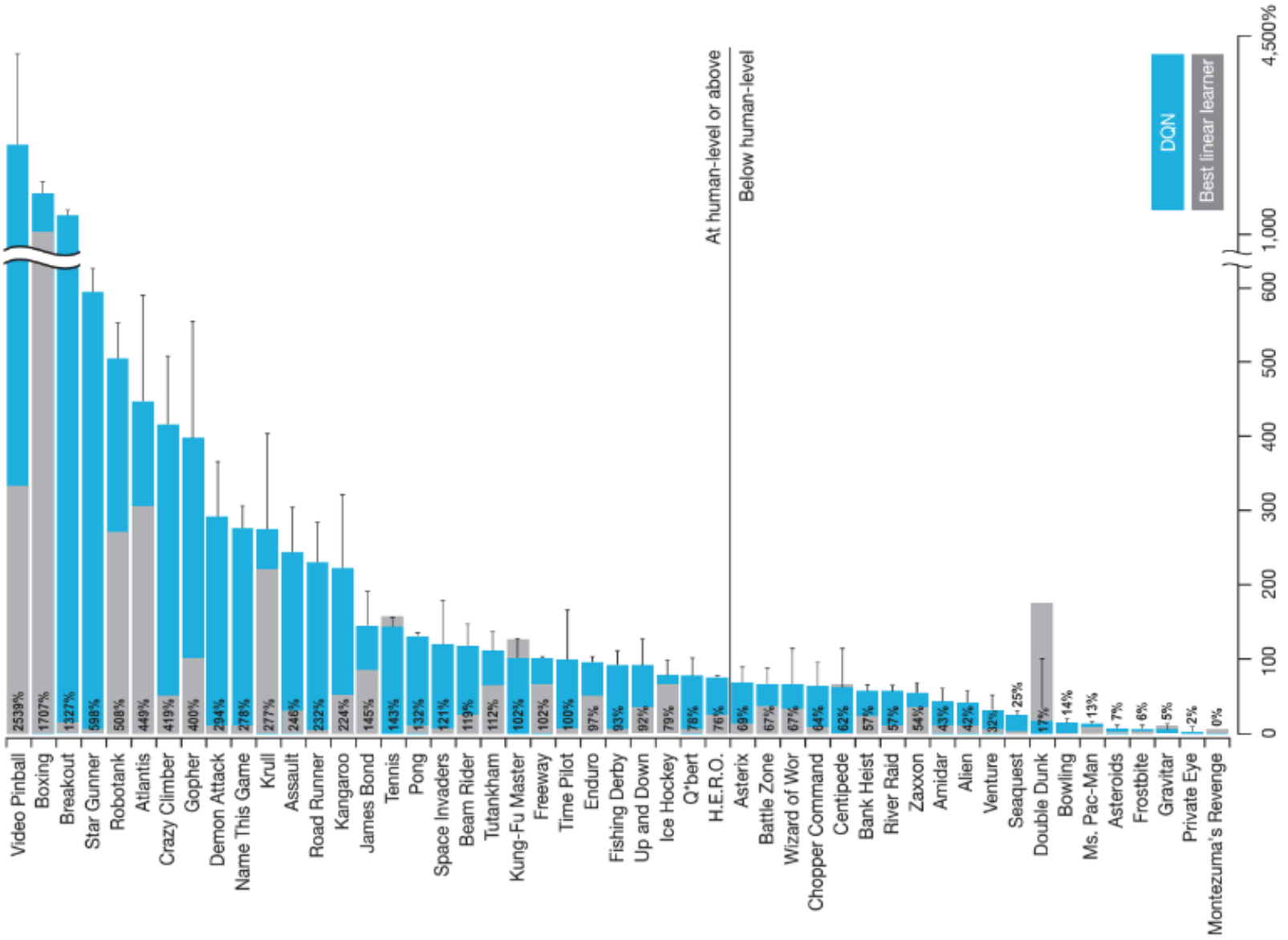
- 効果音なし
- それぞれのゲームで2時間の練習後、20エピソードの平均値



特徴量を設計しなくても

人間より高いスコアが取れるようになった

ただし全てのゲームで勝ってはない



Experience Replay と脳の関係

- 課題後、ノンレム睡眠中のラットCA1領域の**場所細胞の発火率**が上昇することが確認された [Pavlides+]
- さらに、課題中の場所細胞の活動とノンレム睡眠中の場所細胞の活動に**相関がある**ことがわかった [Wilson+]

メモリーリプレイが動物の脳内で行われている

しかし場所細胞の発火は時間順序を保って発火しており
DQNの Experience Replay とは異なる

深層強化学習: まとめ

- **DNNの導入**

DNNで価値関数を表現することで、高次元の入力から直接価値の推定まで**end-to-end**でできるようになった

- **DQNの工夫**
  - ○ **Experience Replay**
    経験を保存してミニバッチで学習することで過学習を避ける
  - ○ **Target Network**
    TD誤差のターゲットの計算に以前のDNNを固定して用いる
  - ○ **Frame Skipping**
    複数フレーム毎に行動選択を行うことでより多くの経験を集める
  - ○ **Clipping Reward**
    報酬のスケールを統一する

内発的動機

目次

- **方策勾配法**

方策を直接求めるには

- **深層強化学習**

DNNによる強化学習によって何ができるようになったか

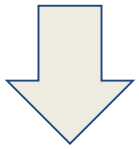
- **内発的動機**

生物は環境からの報酬のみから学習しているのか

内発的動機

人間は**環境からの報酬だけ**で学習しているだろうか？

趣味は外の環境の報酬を最大化するためにやっている？



自分自身の**内発的動機**を満足させるためにやっている

intrinsic motivation

内発的動機と外発的動機

心理学的に以下のように定義できる [Rynan+]

- **内発的動機**

行為それ自身が**本質的にもつ楽しみや満足**のための
動機、興味、挑戦など

e.g. 宿題が面白いからやる

- **外発的動機**

行為自身とは**別の結果を得ることが目的**の
行為をとり続ける動機、操作的価値

e.g. 宿題を親から怒られないためにやる

内発的動機付けするものは何か

かなり古くから議論がなされている

- **最適不一致理論**

知覚と刺激の差異が興味ある対象
最も報酬があるのは新規性が半ば、すなわち既知と完全な新規の間である

- **有能さと自己決定のための動機付け**

自分の能力を最大限に発揮できる自体を追求する行動や自身の能力を向上させようとする行動を駆り立てる源

新規性や驚きが内発的動機に繋がる

内発的動機と大脳基底核

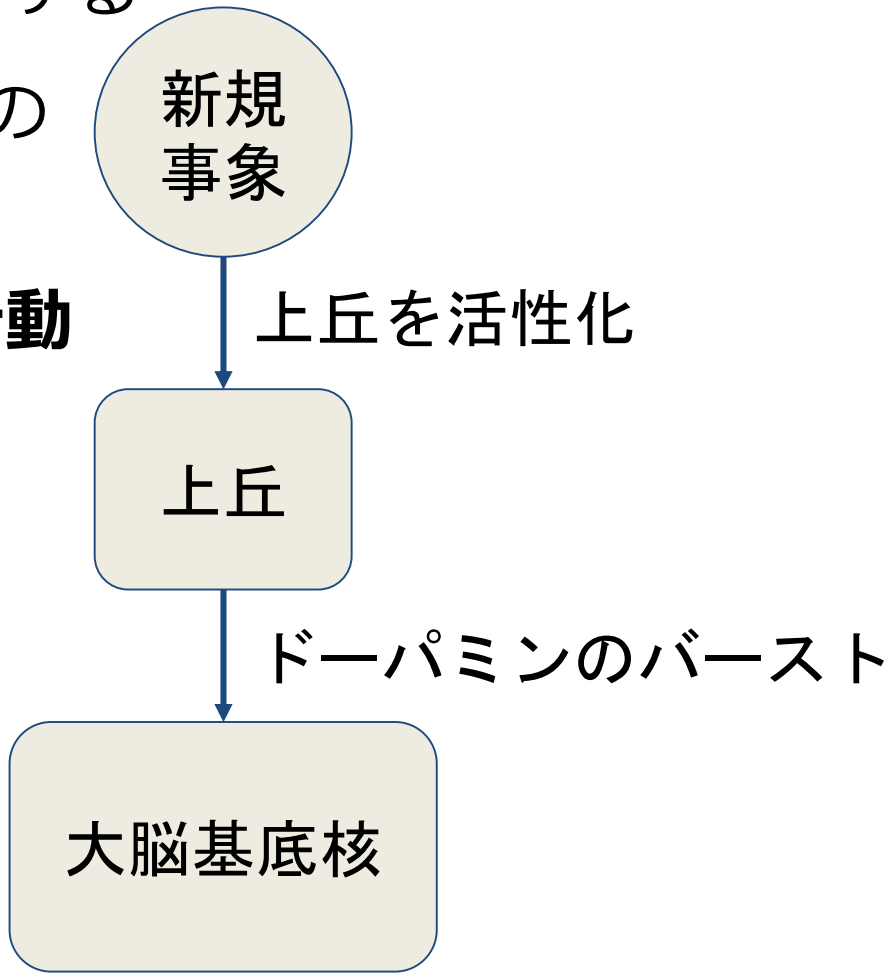
新規事象が上丘を活性化し
ドーパミンのバーストが発生する

このドーパミンが大脳基底核の
以下の情報を想起させる

- **新規事象を引き起こした行動**
- **新規事象のコンテキスト**

繰り返し発生すると上丘は
活性化しなくなる

新しいスキルの学習



Curiosityを強化学習に導入する

A Possiblity for Implementing Curiosity and Boredom in Model-Building Neural Controllers [Schmidhuber 1991]

オンラインモデルベース強化学習において好奇心を導入する
コンセプトを提案している

Curiosityとは環境のモデルを改善したいという動機である

環境モデルによる推定と実際の観測の差から

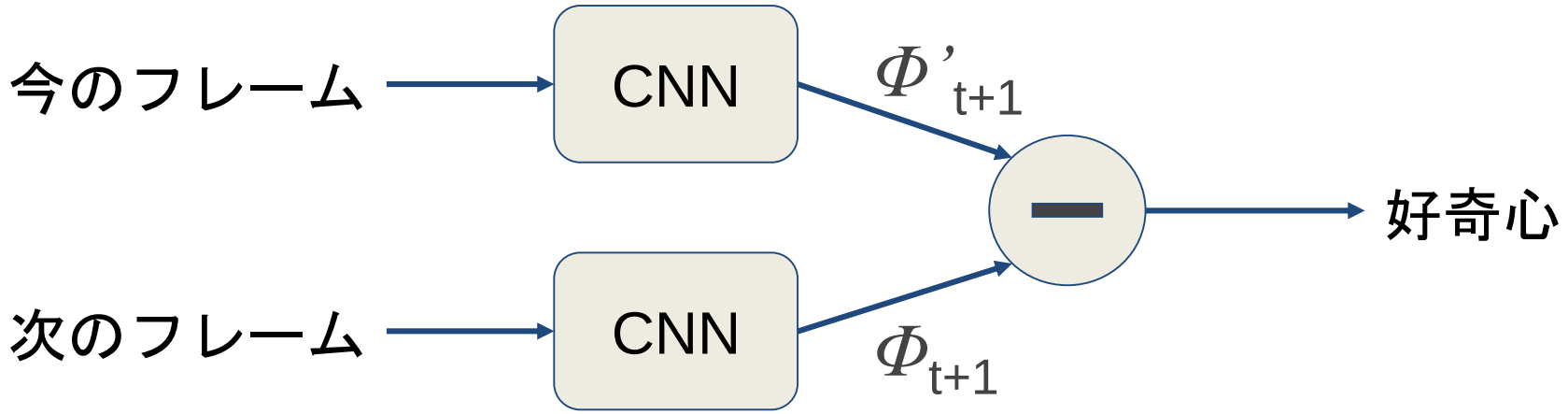
- **誤差の大きい状態**: 正の内部報酬を与える
- **誤差の小さい状態**: 少ない正の内部報酬を与える

飽きるまで（誤差がなくなるまで）探索することで
環境のモデルを改善できる

最新のCuriosity

Curiosity-driven Exloration by Self-supervised Prediction [Pathak+ 2017]

次のフレームの表現を推測して、実際のフレームの表現との差分を好奇心として内部報酬を発生させる手法

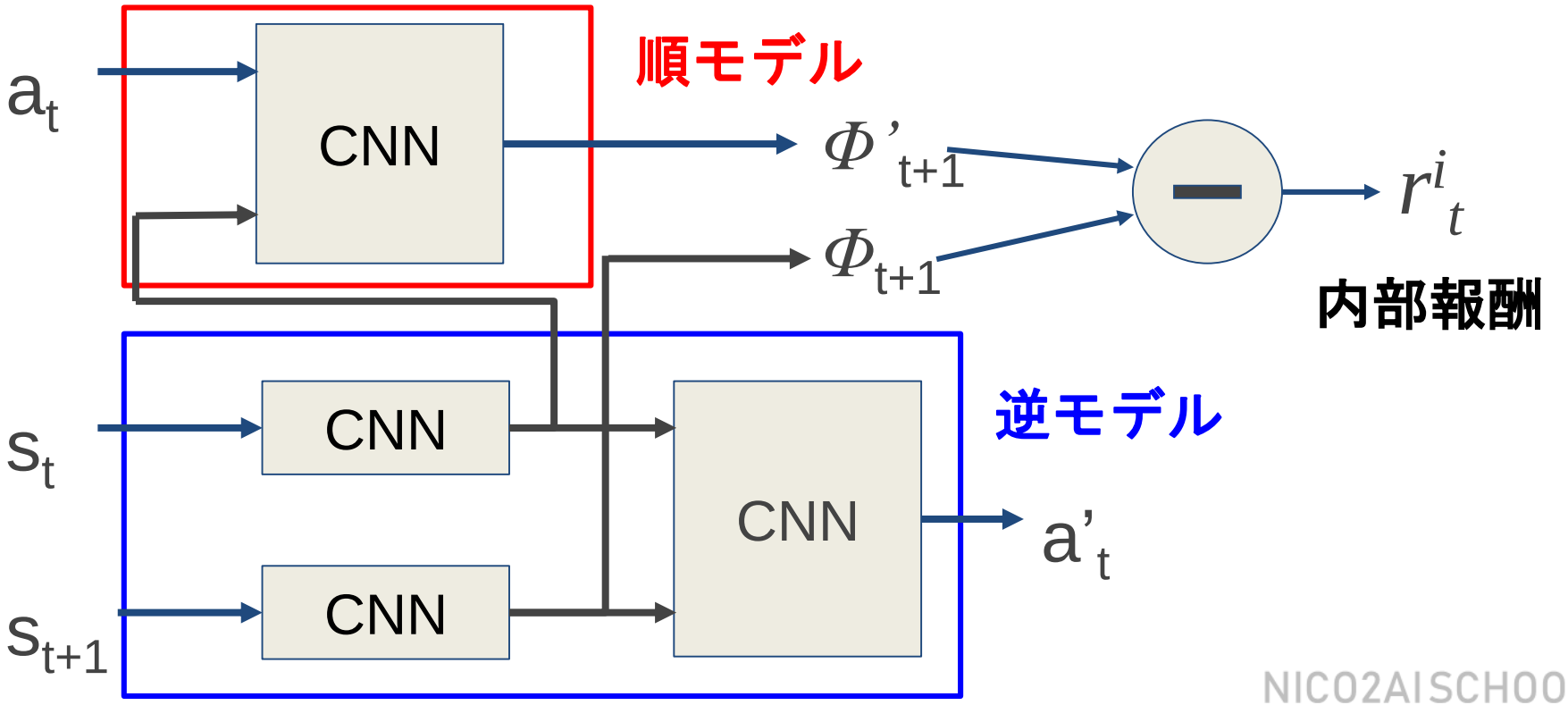


外部報酬一切なしで効率的な探索が行えるようになった

学習モデル

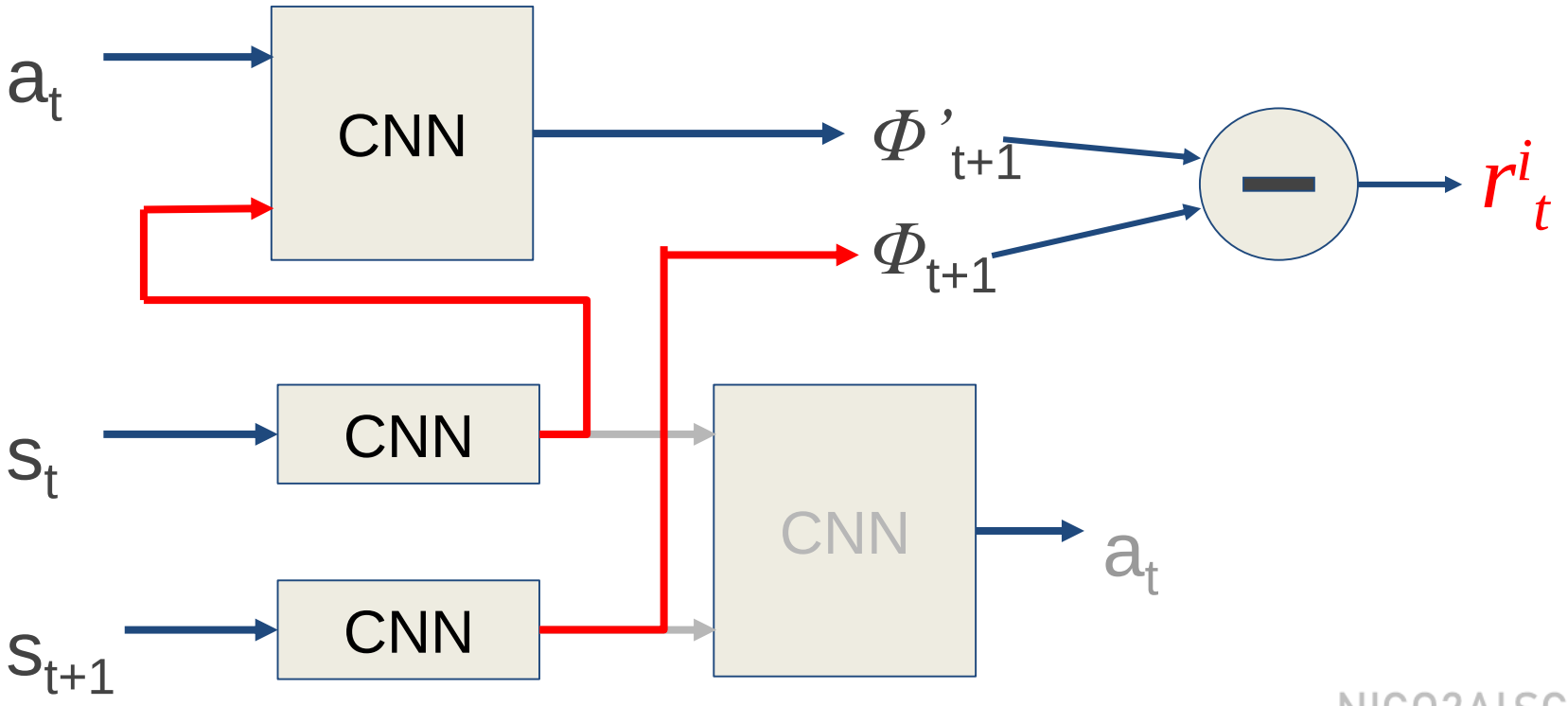
方策と別に以下の二つのものを一緒に学習して内部報酬を計算

- 逆モデル: s_tとs_{t+1}からa_tを推定
- 順モデル: a_tと上で得られる特徴量$\Phi(s_t)$から$\Phi(s_{t+1})$を推定

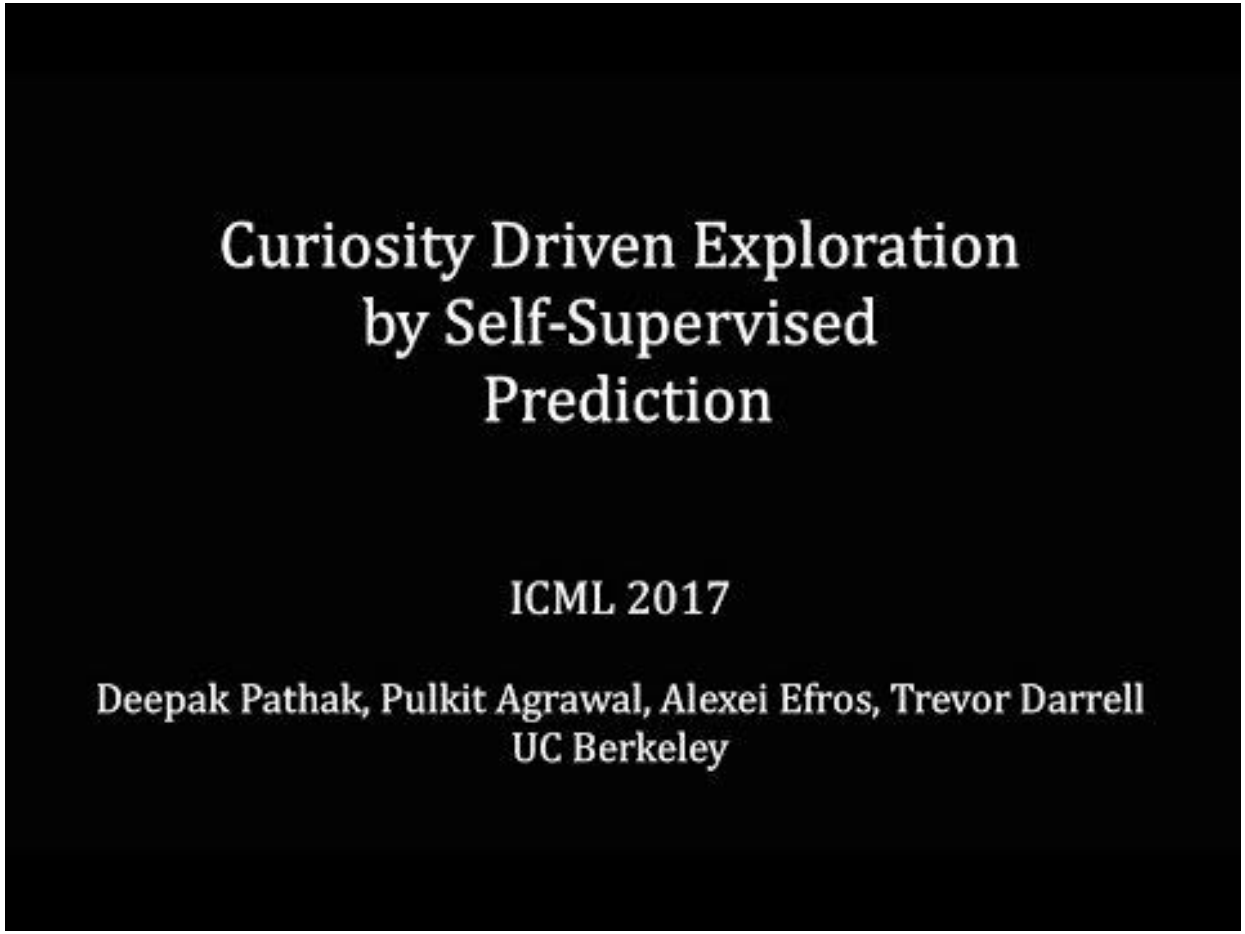


内部報酬の計算

$$r_t^i = \frac{\eta}{2}\|\hat{\phi}(s_{t+1}) - \phi(s_{t+1})\|_2^2$$



動画



https://www.youtube.com/watch?v=J3FHOyhUn3A&t=35s

内発的動機: まとめ

- **内発的動機**

人間は外部からの報酬だけでなく、**新規性や驚き**に対して内発的動機を発生させて学習を行なっている

- **大脳基底核の新スキルの学習**

上丘が**新規事象**に対してドーパミンバーストを起こして大脳基底核の情報を想起できるようにしている

- **好奇心**

観測の推定と実際の観測の差が大きい状態に**好奇心**として内部報酬を与えることで外部の報酬に頼らない探索を行える