

強化学習 RL ロボティクス基礎

- 強化学習のデモ

```
# for Reinforcement Learning
cd ~/study/2018karpathy_reinforcejs.git
open index.html
```

OpenAI 提供の強化学習環境 gym を使うと環境構築は楽です。ただし colaboratory 上で動作させるためには StarAI の開発したレンダリング環境が必要です。

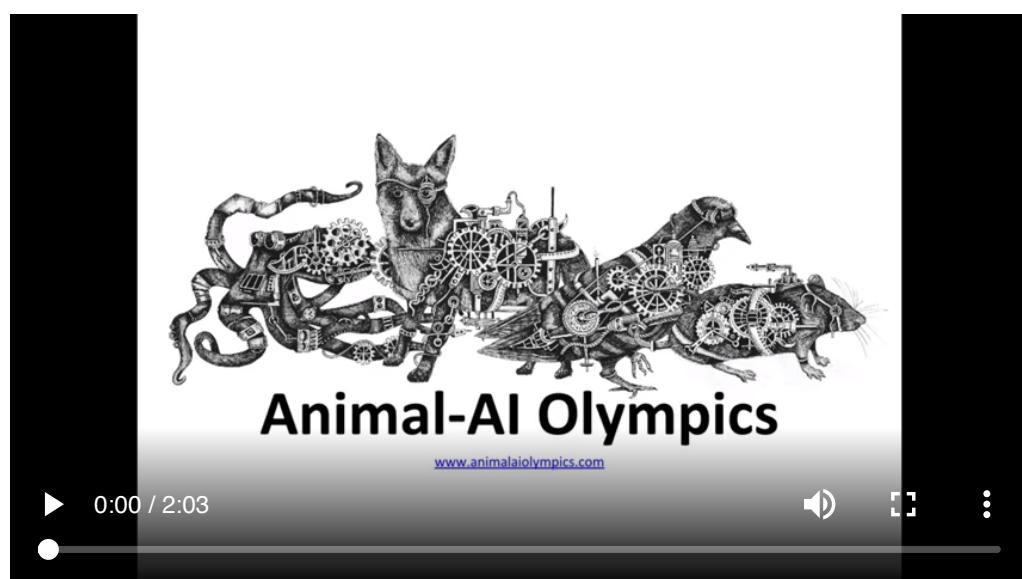
```
import gym
env = gym.make('CartPole-v0')
env.reset()
for _ in range(1000):
    env.render()
    env.step(env.action_space.sample()) # take a random action
```

cartpole 問題を解いてみました

```
cd ~/study/2019tensorflow_models.git/research/a3c_blogpost
# python a3c_cartpole.py --train
python a3c_cartpole.py --algorithm=random --max-eps=4000
```

The Animal-AI Olympics

<http://animalaiolympics.com/>



Unity Obstacle Tower Challenge



強化学習という言葉は古い言葉ですが機械学習の文脈では、環境とその環境におかれた動作主（エージェントと言ったり、ロボットシステムだったりします）が、環境と相互作用しながらより良い行動を形成するためのモデルです。動作主は、環境から受け取った現在の状態を分析して、次にとるべき行動を選択します。このとき将来に渡って報酬が最大となるような行動を学習する手法の一つです。

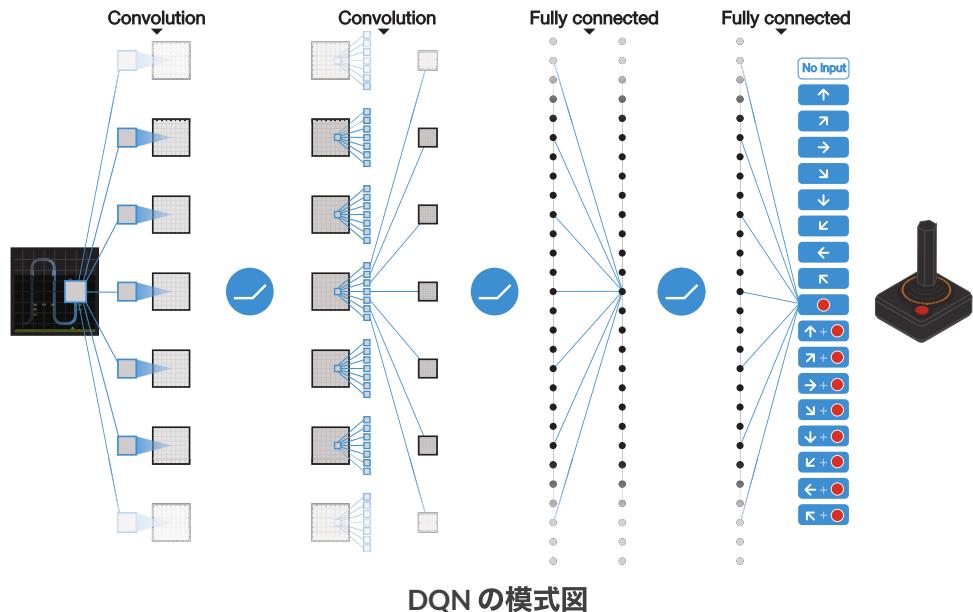
2015年には、Google傘下のDeepMindというスタートアップチームが開発した囲碁プログラムAlphaGoがプロ棋士のイ・セドル氏に勝利し話題になりました。AlphaGoは強化学習を基本技術の一つとして用いています。

- エージェントと環境、マルコフ決定過程 MDP, POMDP, 効用関数, ベルマン方程式, 探索と利用のジレンマ, SARSA:
- 値算, 方策, Q学習, モデルベース対モデルフリー:
- 深層Q学習:
- ゲームAIへ(AlphaGo, AlphaGoZero, OpenAI Five):
- セルフプレイ:
- 最近の発展 A3C, Rainbow, RDT, World model:

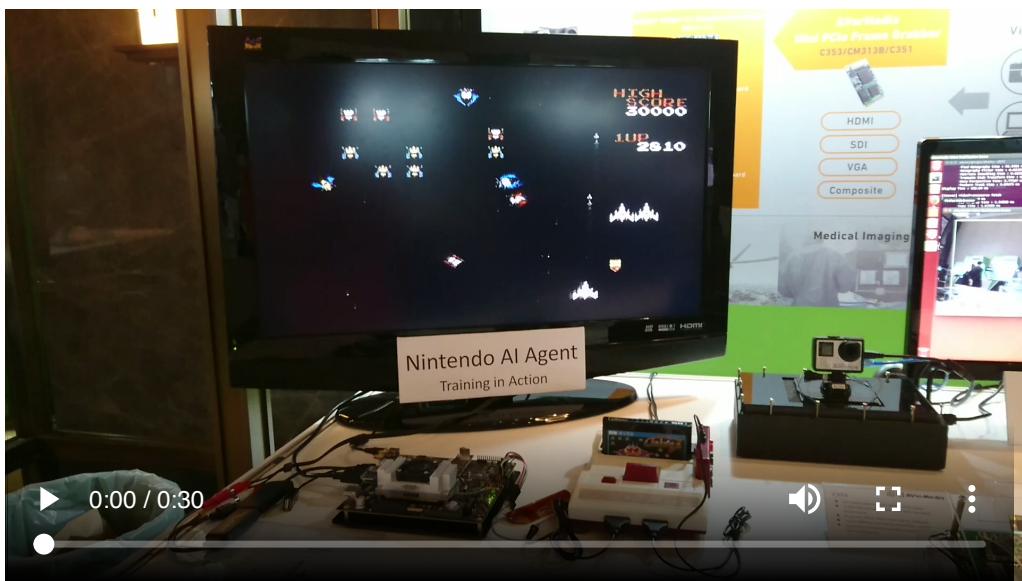
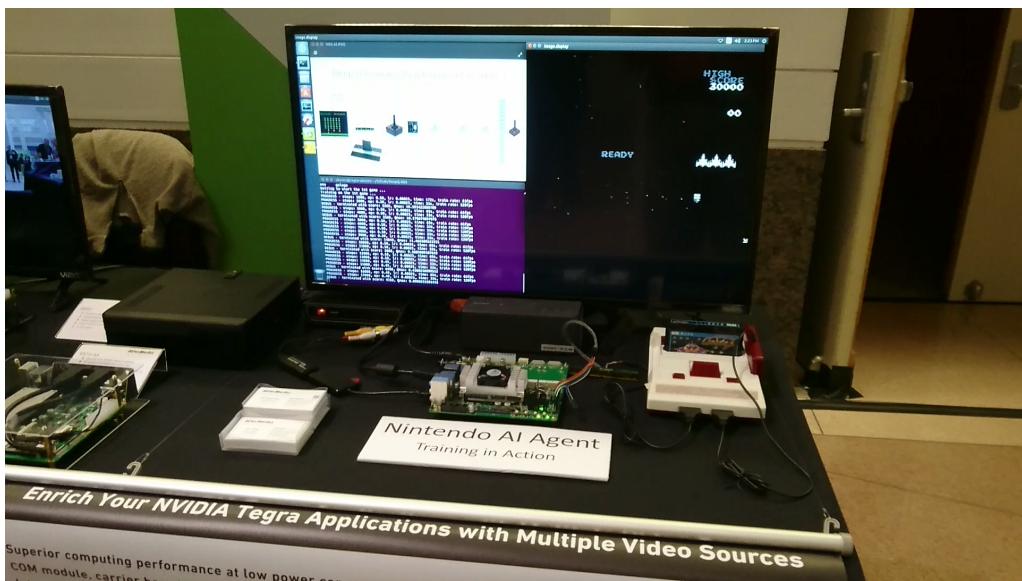
複雑な状況をどう理解して解決するのか？

- 強化学習というニューラルネットワークモデルがあるわけではない
- 動的で複雑な環境に対処 → **強化学習 + DL** → 一般人工知能への礎
- DQN ATARIのビデオゲーム, <https://www.nature.com/articles/nature14236>
- AlphaGo 囲碁, <https://www.nature.com/articles/nature16961>
- AlphaGoZero 围碁, <https://www.nature.com/articles/nature24270>

Deep Q Network



DQN の模式図



- **Q 学習** Q learning に DNN を採用
- CNN が LeNet, @1998LeCun そうであったように、強化学習 RL も昔からの技術 @Sutton_and_Barto1998
- ではなぜ、今になって囲碁や自動運転に応用できるようになったのか？
- ⇒ コンピュータの能力、データ規模、アルゴリズムの改良、エコシステム(ArXiv, Linux, Git, ROS, AMT, TensorFlow)

強化学習

- 強化学習 ⇒ 意思決定
- エージェント agent が 行動(行為) action をする
- 行動によって 状態 が変化する
- 環境 から与えられる 報酬 によって目標が決定
- 深層学習: ⇒ 表現, 表象
- 教師信号として目標が与えられる
- 目標を達成するために外部状況の 表現 を獲得

強化学習 + 深層学習 = 人工知能

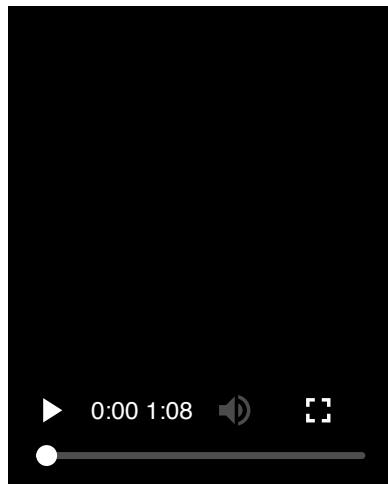
- 強化学習 ⇒ 目標の設定
- 深層学習 ⇒ 内部表象の獲得機構を提供

用語の整理

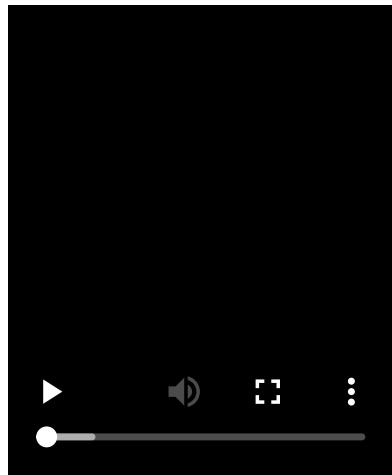
- 教師信号なし 報酬信号 reward signal
- 遅延フィードバック
- 価値 Value
- 行為 Action
- 状態 State
- TD 学習
- Sarsa
- Q 学習
- アクタークリティック
- 報酬 R_t : スカラ値
- 時刻 t でエージェントのとった行動を評価する指標
- エージェントは累積報酬 cumulative reward の最大化する
- 報酬仮説: 目標は累積期待報酬の最大化として記述可能

デモ

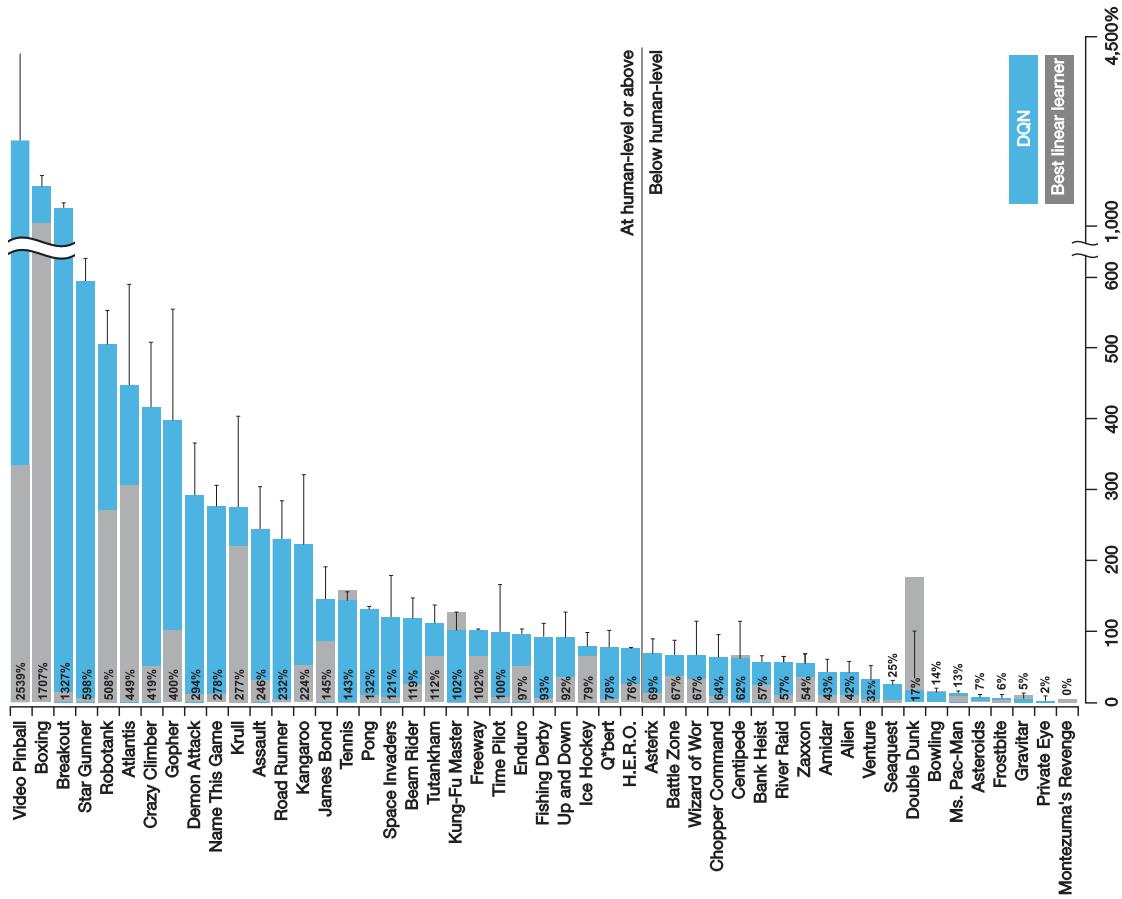
- ブロック崩し: <https://www.youtube.com/watch?v=V1eYniJ0Rnk>
- スペースインベーダー: <https://www.youtube.com/watch?v=W2CAghUiofY>
- OpenMind selfplay: <https://www.youtube.com/watch?v=OBcjhp4KSgQ>
- DQN の動画 スペースインベーダー



- DQN の動画 ブロック崩し



DQN 結果



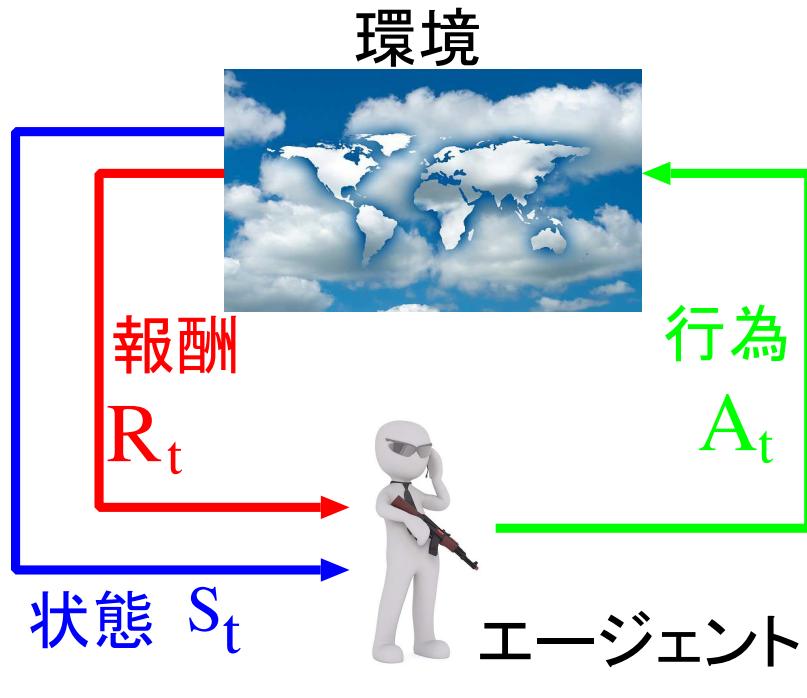
なぜ DQN には難しいのか？



人間にはできて強化学習には難しいこと

- Montezuma's Revenge の動画 <https://www.youtube.com/watch?v=Klxsg9JM5tY>
 - Private Eyes の動画 <https://www.youtube.com/watch?v=OfyS-Wj1M78>
-

エージェントと環境



<https://pixabay.com/en/globe-clouds-sky-background-earth-3382522/>,
<https://pixabay.com/en/weapon-guard-soldier-protection-1816313/>

- エージェント: 学習と意思決定を行う主体
- 行動 A_t を行い
- 環境の 観察 O_t を行う
- 環境からスカラ値の 報酬 reward R_t を受け取る
- 環境: エージェント外部の全て
- エージェントから 行為 A_t を受け取り
- エージェントに 観察 O_{t+1} を与え
- エージェントへ 報酬 R_{t+1} を与える

エージェントの要素

- 方策 Policy
- 価値関数 Value function
- モデル エージェントが持つ環境の表象

方策 policy

- 方策: エージェントの行為
- 決定論的方策: $a = \pi(S)$
- 確率論的方策: $\pi(a|s) = p(A_{t=a}|S_{t=s})$

価値関数

- 将来の報酬予測
- 状態評価(良/悪)
- 行為の選択

$$v_\pi(S) = \mathbb{E}_\pi \{ R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_{t=s} \}$$

強化学習のモデル

- 値値ベース
- 方策:なし
- 値値関数:あり
- 方策ベース
- 方策:あり
- 値値関数:なし
- アクター=クリティック Actor Critic
- 方策: あり
- 値値関数: あり
- モデルフリー
- 方策, 値値関数: あり
- モデル: なし
- モデルベース
- 方策, 値値関数: あり
- モデル: あり

探索と利用のジレンマ Exploration and exploitation dilemma

- 過去の経験から、一番良いと思う行動ばかりをしていると、さらに良い選択肢を見つけることができない **探索不足**
 - 更に良い選択肢ばかり探していると過去の経験が活かせない **過去の経験の利用不足**
-

目標, 収益, 報酬

- エージェントの目標は累積報酬を最大化すること (報酬仮説)
- **報酬仮説** Reward Hypothesis
- 目標: 期待報酬の最大化
- 時刻 t における報酬 R_t : スカラ値
- 時刻 t におけるエージェント行為の評価

逐次の意思決定 Sequential Decision Making

- 目標 Goal: 総収益を最大化する行動を選択すること
- 行為, 行動 Actions は長期的結果
- 収益は遅延することもある
- 直近の報酬を選ぶよりも、長期的な報酬を考えた方が良い場合がある

収益 Return

- **収益** return G_t : 割引付き収益

$$G_t = R_{t+1} + \gamma R_{t+2} + \dots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

- 割引率 The discount $\gamma \in \{0, 1\}$: 現時点から見た将来の報酬を計算するため
- **遅延報酬** delayed reward の評価

- 0に近ければ **近視眼的** 評価
- 1に近ければ **将来を見通した** 評価

価値関数 Value Function

- 状態価値関数 v と 行動価値関数 q
- 状態価値関数 state-value function:

$$v_\pi(s) = \mathbb{E}_\pi \{ G_t | S_{t=s} \}$$

$$v_\pi(s) = \mathbb{E}_\pi \{ R_{t+1} + \gamma v_\pi(S_{t+1} | S_t = s) \}$$

- 行動価値関数 action-value function:

$$q_\pi(s, a) = \mathbb{E}_\pi \{ G_t | S_t = s, A_t = a \}$$

$$q_\pi(s, a) = \mathbb{E}_\pi \{ R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1}) | S_t = s, A_t = a \}$$

最適価値関数 Optimal Value Function

- 最適状態価値関数:

$$v_*(s) = \max_\pi v_\pi(s)$$

- 最適行動価値関数:

$$q_*(s, a) = \max q_\pi(s, a)$$

- ベルマン方程式 一般に非線形になるので難しい

最適価値関数 Optimal Value Functions

- 最大の価値を与える関数

$$Q^*(s, a) = \max_\pi Q^\pi(s, a) = Q^{\pi^*}(s, a)$$

- 最適価値関数 Q^* が得られれば最適方策 π^* を求めることができる

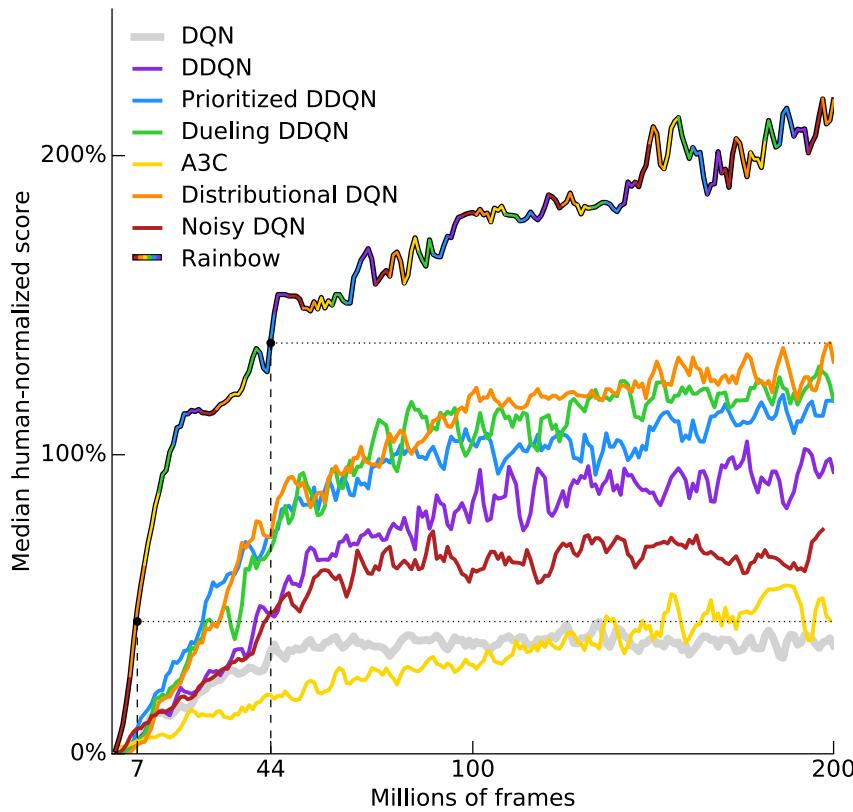
$$\pi^*(s) = \operatorname{argmax}_a Q^*(s, a)$$

- 全ての意思決定における最適価値:

$$\begin{aligned} Q^*(s, a) &= r_{t+1} + \gamma \max_{a_{t+1}} r_{t+2} + \gamma^2 \max_{a_{t+2}} r_{t+3} + \dots \\ &= r_{t+1} + \gamma \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1}) \end{aligned}$$

- ベルマン方程式 Bellman equation:

$$Q^*(s, a) = \mathbb{E}_{s'} \left\{ r + \gamma \max_{a'} Q^*(s', a') | s, a \right\}.$$



すでに結果が古いのですが Rainbow の性能

Game	DQN	A3C	DDQN	Prior. DDQN	Duel. DDQN	Distrib. DQN	Noisy DQN	Rainbow
alien	634.0	518.4	1033.4	900.5	1,486.5	1,997.5	533.3	6,022.9
amidar	178.4	263.9	169.1	218.4	172.7	237.7	148.0	202.8
assault	3489.3	5474.9	6060.8	7,748.5	3,994.8	5,101.3	5,124.3	14,491.7
asterix	3170.5	22140.5	16837.0	31,907.5	15,840.0	395,599.5	8,277.3	280,114.0
asteroids	1458.7	4474.5	1193.2	1,654.0	2,035.4	2,071.7	4,078.1	2,249.4
atlantis	292491.0	911,091.0	319688.0	593,642.0	445,360.0	289,803.0	303,666.5	814,684.0
bank_heist	312.7	970.1	886.0	816.8	1,129.3	835.6	955.0	826.0
battle_zone	23750.0	12950.0	24740.0	29,100.0	31,320.0	32,250.0	26,985.0	52,040.0
beam_rider	9743.2	22707.9	17417.2	26,172.7	14,591.3	15,002.4	15,241.5	21,768.5
berzerk	493.4	817.9	1011.1	1,165.6	910.6	1,000.0	670.8	1,793.4
bowling	56.5	35.1	69.6	65.8	65.7	76.8	79.3	39.4
boxing	70.3	59.8	73.5	68.6	77.3	62.1	66.3	54.9
breakout	354.5	681.9	368.9	371.6	411.6	548.7	423.3	379.5
centipede	3973.9	3755.8	3853.5	3,421.9	4,881.0	7,476.9	4,214.4	7,160.9
chopper_command	5017.0	7021.0	3495.0	6,604.0	3,784.0	9,600.5	8,778.5	10,916.0
crazy_climber	98128.0	112646.0	113782.0	131,086.0	124,566.0	154,416.5	98,576.5	143,962.0
defender	15917.5	56533.0	27510.0	21,093.5	33,996.0	32,246.0	18,037.5	47,671.3
demon_attack	12550.7	113,308.4	69803.4	73,185.8	56,322.8	109,856.6	25,207.8	109,670.7
double_dunk	-6.0	-0.1	-0.3	2.7	-0.8	-3.7	-1.0	-0.6
enduro	626.7	-82.5	1216.6	1,884.4	2,077.4	2,133.4	1,021.5	2,061.1
fishing_derby	-1.6	18.8	3.2	9.2	-4.1	-4.9	-3.7	22.6
freeway	26.9	0.1	28.8	27.9	0.2	28.8	27.1	29.1
frostbite	496.1	190.5	1448.1	2,930.2	2,332.4	2,813.9	418.8	4,141.1
gopher	8190.4	10022.8	15253.0	57,783.8	20,051.4	27,778.3	13,131.0	72,595.7
gravitar	298.0	303.5	200.5	218.0	297.0	422.0	250.5	567.5
hero	14992.9	32464.1	14892.5	20,506.4	15,207.9	28,554.2	2,454.2	50,496.8
ice_hockey	-1.6	-2.8	-2.5	-1.0	-1.3	-0.1	-2.4	-0.7
kangaroo	4496.0	94.0	11204.0	10,241.0	10,334.0	9,555.5	7,465.0	10,841.0
krull	6206.0	5560.0	6796.1	7,406.5	8,051.6	6,757.8	6,833.5	6,715.5
kung_fu_master	20882.0	28819.0	30207.0	31,244.0	24,288.0	33,890.0	27,921.0	28,999.8
montezuma_revenge	47.0	67.0	42.0	13.0	22.0	130.0	55.0	154.0
ms_pacman	1092.3	653.7	1241.3	1,824.6	2,250.6	2,064.1	1,012.1	2,570.2

すでに結果が古いのですが Rainbow の性能

教科書、参考文献

- An Introduction to Reinforcement Learning, Sutton and Barto, 1998, MIT Press, 1998
<http://incompleteideas.net/book/the-book.html>, Sutton and Barto (1998) (2018年第2版出版) 翻訳 強化学習 <https://www.amazon.co.jp/dp/4627826613>
- Algorithms for Reinforcement Learning, Szepesvari, Morgan and Claypool, 2010
<https://sites.ualberta.ca/~szepesva/papers/RLAlgsInMDPs.pdf>
- デービッド・シルバーの講義 <http://www.cs.ucl.ac.uk/staff/d.silver/web/Teaching.html>

- ジヨン・シュルマンのビデオ講義 <https://www.youtube.com/watch?v=oPGVsoBonLM>
- 「これからの強化学習」 <https://www.amazon.co.jp/dp/4627880316/>
- Algorithms for Reinforcement Learning, Szepesvari, Morgan and Claypool, 2010
<https://sites.ualberta.ca/~szepesva/papers/RLAlgsInMDPs.pdf>