深層強化学習アルゴリズムの特徴解析

1 はじめに

近年, 深層学習をはじめとする機械学習の分野がめざましい発展を遂げている. 機械学習は教師あり学習, 教師なし学習, 強化学習の大きく 3 つに分類されるが, 人工知能がゲームをプレイするタスクなどを中心に強化学習の様々なアルゴリズムが提案されている.

強化学習は与えられた問題の状態と行動, そして報酬を正しく定めなければ適切に問題を解くことは難しく, 一見同じに見える問題であっても元の問題に外乱が加わるなど少し条件が変われば同じアルゴリズムでこれらの問題を解いたとしても異なる結果が得られる可能性がある. そこで今回の実験では様々な強化学習アルゴリズムについて元の問題と外乱を加えた問題それぞれで解かせることによって, 外乱を加えた時にそれぞれのアルゴリズムの性能がどう変化するかを確認し, それらの結果について考察した.

2 要素技術

2.1 Q 学習

強化学習のアルゴリズムは大きく分けて 2 つ存在し 1 つは方策反復法と呼ばれ, もう 1 つは価値反復法と呼ばれる. 方策反復法は現在の状態に応じてエージェントがどのように行動するかを決定する戦略である方策を直接更新する方法である. 具体的には成功時に取った行動を重要視してその行動を多く取り入れるように方策を更新していく. 一方, 価値反復法では成功時に報酬を与えて, 現在の状態とそれに対する行動の組に対して報酬に基づく価値を定める. 状態によって定まる価値を状態価値関数と呼び, 状態と行動の組によって定まる価値を行動価値関数または Q 値と呼ぶ. そして次ステップと現ステップの Q 値の差分である TD 誤差を計算してその差分に応じて現在の Q 値を更新することによって, 状態に対する行動の価値を学習していき正しい行動を選択させる方法である. Q 学習は価値反復法の典型的なアルゴリズムであり, Q 値の更新式は以下の式で表される.

$$Q(S_t, a_t) \leftarrow Q(S_t, a_t) + \alpha(R_{t+1} + \gamma \max_a Q(S_{t+1}, a_{t+1}) - Q(S_t, a_t)) \tag{1}$$

ここで t は時刻, S は状態, a は行動, α は学習率, R は報酬, γ は時間割引率である. Q 学習をコンピュータを使って解く場合は行動価値関数を何らかの形で格納する必要があるが, 主には状態を行, 行動を列とした表形式によって格納される.

2.2 Deep Q-learning(DQN)

Q 学習で問題を解く場合, 問題の状態数が多かったり状態がそもそも状態が連続値であると, 初めて直面する状態が多くなるので, Q 値がなかなか更新されず正しい行動を学習するために多くの時間がかかり, さらに表形式として Q 値を格納するためのメモリを莫大に消費してしまうといった問題がある. これらの問題の対応策として連続値などを離散化して表形式を実現する方法はあるが, 離散化する方法は人間によって定める必要があるため, 手間が掛かり状態と行動を正しく設定できない可能性があるので問題を解くことが難しくなってしまう.

DeepMind 社が 2013 年に発表した DQN[1] では Q 値を表形式の代わりにニューラルネットワークによって表現する. つまり, 各状態変数の値を入力としてそれに対する行動ごとの Q 値を出力値として返すネットワークを作ることによって Q 学習の問題を解決した. Q 値を更新する際はネットワークのパラメータを更新することによって行えばよく, 状態が連続値であっても回帰によって初めて直面する状態にもある程度対応できる. またゲーム画面のような画像を入力する状態とすることができるため, 学習する環境ごとに状態を手作業で設定する必要がない. DQN はスペースインベーダなどで知られる Atari のゲームを解くことが出来たため DQN の発表以降, DQN の改良アルゴリズムが多く提案された.

DQN を実装するためには 4 つの工夫がされており, 1 つ目は Experiment Replay と呼ばれる, 状態や行動などの組をメモリに格納してミニバッチ学習で

メモリからランダムに取り出すことによって時間的に相関が高い内容を連続して学習することを防ぎ学習を安定させる方法である. 2 つ目は Fixed Target Q-Network と呼ばれる方法である. DQN の Q 値の更新には (1) 式を見ればわかるように Q 値自身を使うため学習が安定しない傾向にある. そのため行動を決定する Q 値のネットワークと Q 値を更新するための Q 値のネットワークを分けることによって, 学習を安定させている. 3 つ目は Reward Clipping と呼ばれる手法で, すべての問題の報酬を-1, 0, 1 のいずれかにすることで同じハイパーパラメータで学習できるようにしている. 4 つ目はネットワークの学習の際の誤差関数を 0 付近で微分可能かつ外れ値に敏感になりにくい Huber 関数を用いて学習を安定させる方法である.

2.3 Rainbow

DQN によって Atari ゲームをプレイするような難度のタスクを解けるようになったが, それでも学習が安定しにくいといった問題が存在したため, これらの問題を解決するため DQN が発表されてからもその改良手法が数多く提案された. それらのアルゴリズムのうち, Double-DQN(DDQN), Prioritized Experience Replay, Dueling Network, N-Step learning, Noisy Network, Categorical DQN の 6 つの改良手法と元々の DQN を組み合わせた Rainbow と呼ばれるアルゴリズムが 2018 年に発表された.[2] それぞれの改良手法に関する説明は紙面の都合上割愛する.

2.4 Cartpole

Cartpole は OpenAI Gym が提供している強化学習で利用できる環境の 1 つであり, 台車に乗った棒を倒れないように左右に動かしていく制御課題である. 状態はカートの位置, カートの速度, 棒の角度, 棒の角速度の 4 変数がそれぞれ連続値で表現されていて, エージェントの行動は台車を左に動かすか右に動かすかの 2 種類である.

3 数値実験

今回の実験では Rainbow に含まれるアルゴリズムのうち DQN, DDQN, Prioritized Experience Replay DDQN, Dueling Network DDQN, NoisyNet DDQN の 5 つのアルゴリズムとこれらのアルゴリズムをすべて搭載した手法の計 6 つに対して CartPole の環境を変えていき外乱を加えることで実験を行った. もともとの CartPole では台車に加える力を 10 という値で固定して左右に動かしているが, この力の値に平均が 0 の正規分布の値を加えることで行動における台車を動かす力に対してランダムな摂動を与え, 標準偏差の値を変えることによって外乱の大きさを調整した. つまり今回の実験における台車を動かす力は以下の式で表される.

$$F = 10 + \epsilon \tag{2}$$

$$\epsilon \sim N(0, \sigma^2) \tag{3}$$

表 1 に今回の実験で用いたパラメータを示す. 最適化手法については通常の CartPole 環境で各アルゴリズムについて Optuna でパラメータチューニングを行い, その結果を基としている. また今回の実験ではエージェントが Cartpole を 195 ステップ以上制御し続けたら成功として報酬を 1, 195 未満で棒が倒れたら報酬を -1, 倒れていなかったり, 200 ステップに達していない場合は報酬を 0 としており, 10 エピソード連続で制御に成功したら学習自体が成功したとしてその時点で学習を打ち切り, 500 エポックに達しても 10 エポック連続で制御に成功していなかったら学習自体を失敗とした. 各アルゴリズムに対して 100 回実験をしており, 簡単のためニューラルネットワークの入力は画像ではなく CartPole で設定されている前述の 4 状態を入力とした.

表 1: 実験パラメータ

入力サイズ	4
隠れ層サイズ	32
出力サイズ	2
バッチサイズ	32
累積報酬の時間割引率	0.99
最大ステップ数	200
最大エポック数	500
最適化手法	Adadelta もしくは Adam
損失関数	Huber

4 結果

まずこの正規分布の標準偏差を 2, 3, 4, 5, 6, 10, 15 と増やしていき, 学習が完了した際の平均エポック

数の推移を求めることでアルゴリズムの性能を調べた. 図 1 にこの結果を示す. 横軸は摂動の値が従う正規分布の標準偏差の大きさであり, 大きいほど環境に大きな外乱を加えていることを表す. 縦軸は学習が完了した際のエポック数であり, この値が小さいほど速く学習が完了しているのでアルゴリズムの性能が良いことを表す. この図を見ればわかるように DQN は外乱を加えていない環境でもあまり学習されず性能が高くないことが分かり, 外乱を強く加えていない環境では DDQN 単体が一番学習が早く進み逆に 5 つを全部載せしているアルゴリズムの方が遅い結果となった. 一方, 外乱を強く与えた環境ではこの図だけ見れば DDQN 単体と 5 つを全部載せしたアルゴリズムが同程度の速さで学習していることが分かる.

次に通常の CartPole 環境と力に対して標準偏差が 15 の正規分布で得られた値を加えて外乱を与えた環境の 2 つのそれぞれについて, エピソード数に対するステップ数の推移を求めた. 図 2 にこの結果を示す. 横軸はエピソード数であり縦軸はステップ数である. グラフの上昇度合いが速いアルゴリズムほど立ち上りが速く学習されていることを表す. 通常の CartPole 環境ではわずかながら NoisyNet や 5 つを全部のせしているアルゴリズムの方が立ち上りが遅くなっていることが確認された. しかし DQN 以外のアルゴリズムについては 100 エポックに到達する前には平均ステップ数が 175 以上となるほどには学習が速く進むことが分かった.

最後に表 2 に標準偏差が 15 の正規分布で得られた値を加えて外乱を与えた環境について, 各アルゴリズムの学習失敗した数を示す. この図を見ればわかるように, 5 つを全部載せしたアルゴリズムは明らかに他のアルゴリズムと比べて失敗した数が少ないことが分かる. つまり 5 つを全部載せしたアルゴリズムは学習の速度は他のアルゴリズムと比べて遅くはなっているが外乱にも比較的強く学習が進んでいることが確認された.

表 2: 外乱を加えた環境での学習失敗した数

アルゴリズム	学習失敗した数
DDQN	46
DuelingNetDDQN	57
PrioritizedDDQN	65
NoisyNetDQN	78
5 つを全部のせ	28

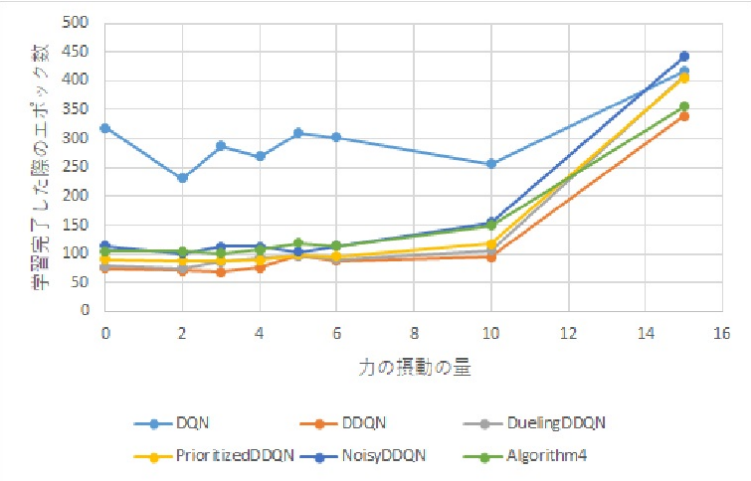

図 1: 摂動の度合いを変えていった際の平均終了エピソード数の推移

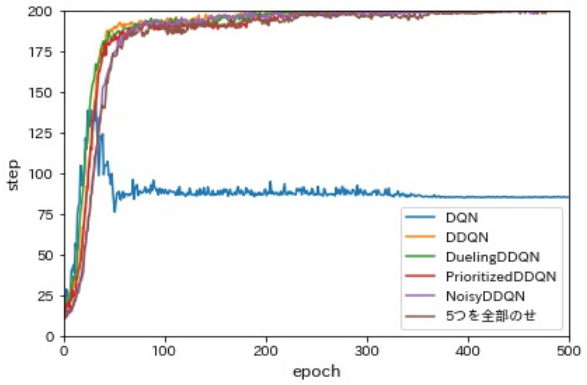

図 2: 通常の CartPole 環境におけるエピソード数に対する平均ステップ数の推移

5 考察

3 つの実験結果から外乱を強く与えていない環境では DDQN 単体のような多くのアルゴリズムを搭載していない手法の方が少し速く学習が進んだのに対して, 外乱を強く与えた環境では 5 つを全部載せしたアルゴリズムのように Rainbow に近いアルゴリズムの方が安定して学習が進むことが分かった. 5 つを全部載せしたアルゴリズムが DDQN 単体よりも学習が遅くなっている理由としては NoisyNet の存在が考えられる. NoisyNet は Q 学習で探索と活用のトレードオフのバランスを保つために用いられる ε-greedy 法に代えて, ネットワーク自体にノイズを加えてノイズの大きさ自体も学習していくアルゴリズムとなっている. このアルゴリズムでは学習のし始めに立ち上りが遅くなる可能性があるので, 今

回の実験ではその影響を受けたものと考えられる. NoisyNet は最終的な学習の向上に寄与するので 5 つを全部載せしたアルゴリズムが外乱に強く, 安定した学習をした結果となったのも NoisyNet の影響ではないかと考えられる.

6 まとめと今後の課題

今回の実験では, Cartpole 環境に外乱を与えた場合とそうでない場合のそれぞれについて, DQN や Rainbow に含まれる DQN の改良アルゴリズムに対する比較実験してそれぞれのアルゴリズムの特徴を解析した. その結果, Rainbow に近いアルゴリズムの方が少し学習速度は遅くなるが外乱に強く安定して学習することが分かった.

今後の課題としては Rainbow に含まれるアルゴリズムである N-step learning と Distributional DQN を比較実験できなかったので, これらのアルゴリズムを加えて実験することや CartPole 以外の学習環境に外乱を加えた環境でアルゴリズムの比較実験を行うことで, 各アルゴリズムの性能を評価することが考えられる.

参考文献

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[2] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Daniel Horgan, Bilal Piot, Mohammad Gheshlaghi Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. *CoRR*, abs/1710.02298, 2017.