TETRIS の AI を強化する

cookie4869

技術チャレンジ部とは

・技術関係のさまざまな自主的な取り組みを行う部活動
・プロコンへの参加、アルゴリズム輪講、VR の実験などをやっている
・社員ならだれでも参加できる！

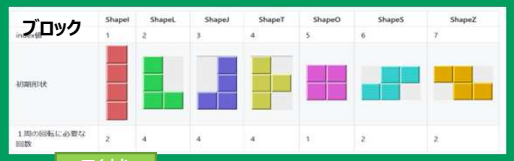
たまたま 掲示板 で 「Tetris で AI を学ぼう」を発見し参加することに。

Tetris の AI を学ぶことに

TETRIS プログラムのしくみ

Start.py から開始。テトリスのプログラムは運営から提供される。
競技者は Block_Controller.py をカスタマイズして競い合う。

	level1	level2	level3	level3+α
ブロック順	固定	ランダム	ランダム	ランダム
初期ブロック	なし	なし	あり	あり
更新頻度	約1秒	約1秒	約1秒	約0.001秒



形状

Start.py
起動条件規定

Game_Manager.py
Game 全体の進行
点数、ルール管理

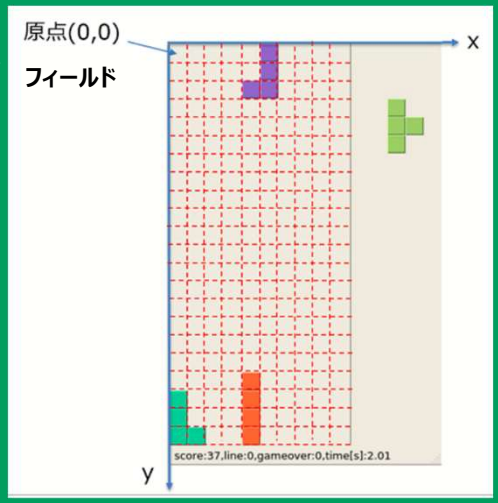
ブロック操作

状態

Board_Manger.py
ブロック（テトリミノ）の管理
フィールドの管理

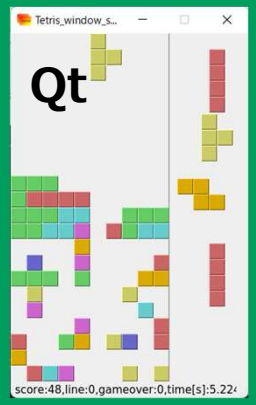
ブロック配置

フィールド状態



画面描画

描画



タイマイベント
動作取得

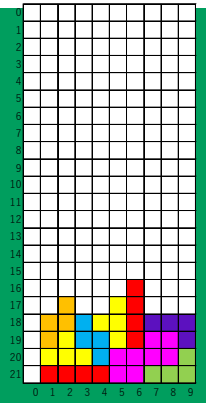
ゲーム情報、フィールド状態

ブロック操作

Block_Controller.py
GetNextMove 関数でフィールド状態から次の操作を決定する
各自変更可能ここを変更して競う

TETRIS の AI のしくみ (推論)

Block_Controller.py の GetNextMove 関数中で下記のような形で処理を行う。



ゲーム情報

GetNextMove

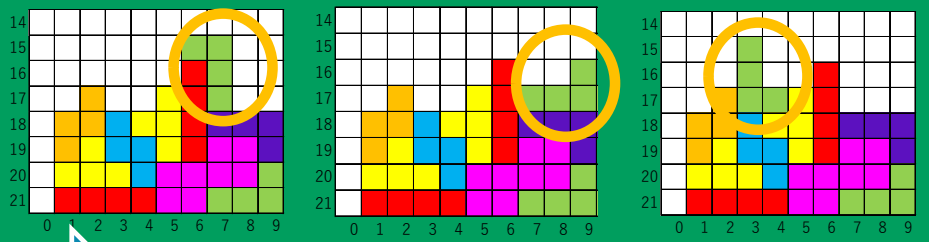
ゲーム状態が渡され次の Action を決定する

フィールド状態

全ての次の局面のフィールド状態

Get_next_func (get_next_states)

次の局面としてあり得る全てのケースを作り出し Model に渡して評価値を出す

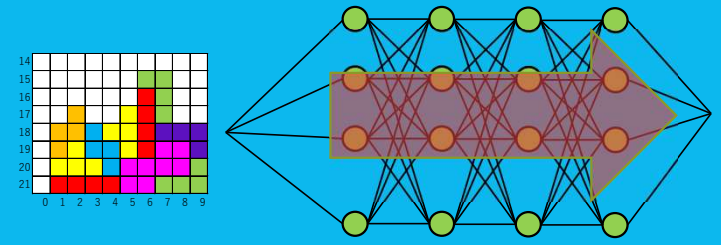


全ての次の局面のフィールド状態

全ての次の局面のフィールド評価値

Model

フィールド状態の評価値を算出するネットワーク
次のシートで説明する学習の結果生成される
Deep Q network, MLP などがある。



フィールド評価値が最大の場合のブロック操作を取得

次のブロック操作

TETRIS の AI のしくみ (学習1)

Block_Controller.py の中で下記のような形で処理を行う。

GetNextMove

ゲーム状態が渡され次の Action を決定する

ゲーム情報

次のブロック操作

次の局面のフィールド評価値が最大の場合のブロック操作を取得
ただし、ε %　の確率でランダム操作

次のブロック操作

上記操作後のフィールド状態作成

次の次の局面のフィールド評価値が最大の場合のブロック操作を取得

現状フィールド状態

全ての次の局面のフィールド状態

全ての次の局面のフィールド状態

全ての次の局面のフィールド評価値

次のフィールド状態

次の報酬

次のフィールド状態

全ての**次の次の**局面のフィールド状態

次の次のフィールド状態

次のフィールド状態

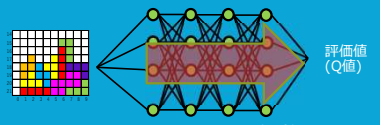
次の報酬情報

Get_next_func (get_next_states)

次の局面としてあり得る全てのケースを作り出し Model に渡して評価値を出す
初期値は全部 1

Model

フィールド状態の評価値を算出するネットワーク
次のシートで説明する学習の結果生成される
Deep Q network, MLP などがある。



Forward 順伝搬

Reward_func

報酬計算。機械学習用の報酬を計算する

Get_next_func (get_next_states)

次の局面としてあり得る全てのケースを作り出し Model に渡して評価値を出す

episode_memory, replay_memory

経験（局面）を記録していく。経験学習用。
次の報酬情報、次のフィールド状態、次の次のフィールド状態を局面ごとに記録する。

TETRIS の AI のしくみ (学習2)

Gameover 時、もしくは規定の局面数に達したら実行される update 関数内で episode_memory を学習する。（経験学習; Experience Learning）

Update

・Gameover 時、もしくは規定の局面数に達したら実行される

・大会は 180 局面で終了のため、180でもよいが、よりいろいろな局面を体験させるため大きくすることもある。

・Update の繰り返しにより Model が完成されていく。1日単位の作業になることも。

報酬情報、
フィールド情報

episode_memory(1ゲーム), replay_memory

経験（局面）を記録していく。経験学習用。
次の報酬情報、次のフィールド状態、次の次のフィールド状態を局面ごとに記録する。

PER: 優先順位つき経験学習 Prioritized Experience Learning

報酬情報、
フィールド情報

512(batch) 個の
報酬情報、
フィールド情報

PER.sampling

過去の履歴と比べて報酬変化の大きいものを優先した形(priority 付)でランダムで replay_memory を 512 (batch) 個抽出

512(batch) 個の
報酬情報、
フィールド情報

512(batch) 個の
報酬と局面評価値の差分

PER.update_priority

「次の次」と「次」の局面の間で報酬変化、評価値変化の割合から上記 sampling に使う優先順位を再設定

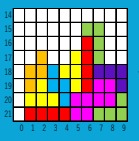
512(batch) 個の
報酬情報
フィールド情報

Model

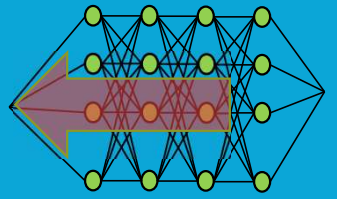
フィールド状態の評価値を算出するネットワーク

報酬情報とフィールド評価値の差を学習

Optimizer により学習率が調整される
Target net として最近　500回の学習で最もよい成績だった Model を比較対象としている。



Backward 逆伝搬



Optimizer 更新指示

Optimizer.step

学習率を決定する。（どの程度フィールド評価値を変化させるか）

TETRIS の AI の強化1

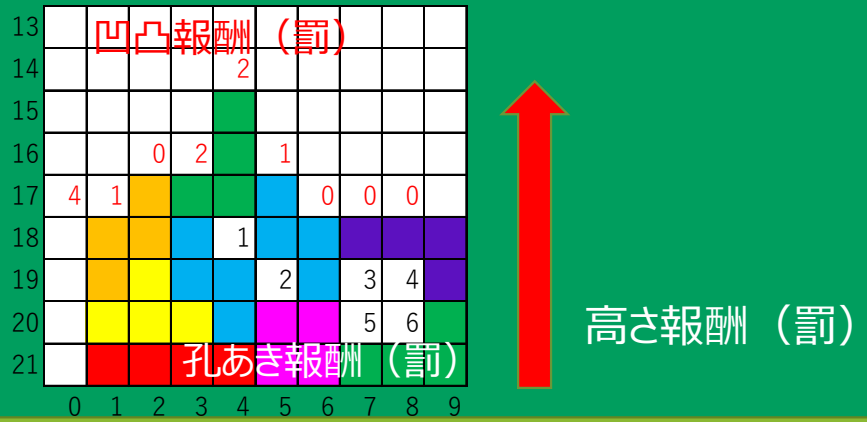
強化のポイントはいくつかある。

Reward_func

報酬計算。

・ラインを消したときの報酬だけでなく、いろいろな報酬を決めることにより、より学習を最適化できる。

・高さが高いと罰、凹凸がひどいと罰、孔が開いたら罰などが考えられる。その罰をどの程度の割合でするかが重要となる。



Tetris Rule

・従来のプログラムは一番上の行から Drop（落下）させているため、下に置く直前で回転させたりすることができない。その機能を導入するという手も。

→T-spin などで高得点にするルールがないためあまりメリットなし。

Model

フィールド状態の評価値を算出するネットワーク

Deep Q network, MLP のどちらにするか

今のところ Deep Q network が安定動作

推論時の使い方も重要

Target net を どのように設定するかも学習の安定度に影響

学習時のランダム操作設定εは少しずつ減らす必要があるが減らし方によっても学習が変化

Optimizer

学習率を変えてみる。

ADAM という学習率調整法が最適。

PER

Batch サイズの変更。大きくすると学習が遅くなる

割引率γにより評価値変化をどれだけ重視するか決定。大きくすると報酬の評価が低下。

β で優先順位の高いものを何次関数で割合を上げるか決定。大きくすると報酬変化の大きいものばかりピックアップ。長い準備を要する場合は大きくしすぎない方がよい。

α で優先順位そのものを重視するか決定。優先順位をつけずに進めると評価に時間がかかるが、網羅的にはなる。

TETRIS の AI の強化2

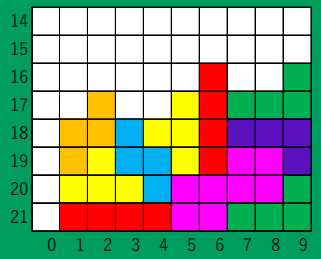
自分は下記のような強化を行った。強化の際は学習状況閲覧ツールの Tensorboardを利用

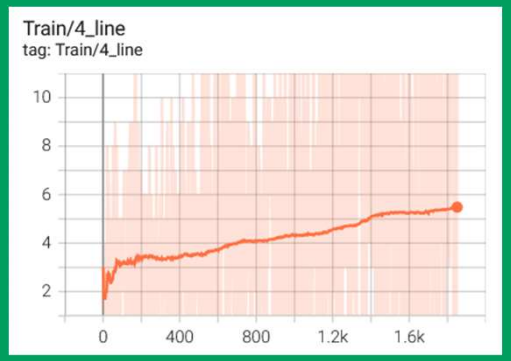
Reward_func

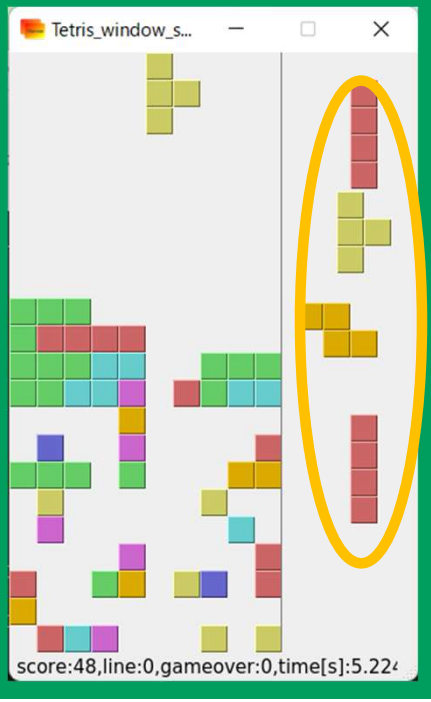
高さについてはある程度許容。
左端についてはブロックをある程度積まないようにすることに対し報酬を与えた。
4ライン消し (テトリスねらいのため)

Model

・推論時に次の局面だけでなく、次の局面も予告ブロックも推論に入れるよう修正。
・Level 3 では動的な Model 変更も導入。





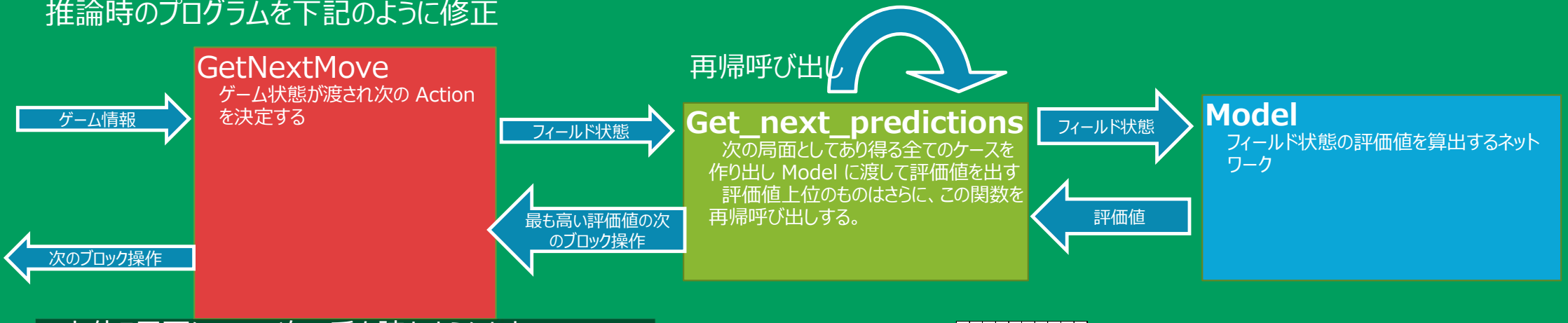


・単純にこれの導入だけではあまり強くならない。強引に導入すると逆に点数は悪化する。

・しかし右の推論修正により劇的に点数が向上。

TETRIS の AI の強化3

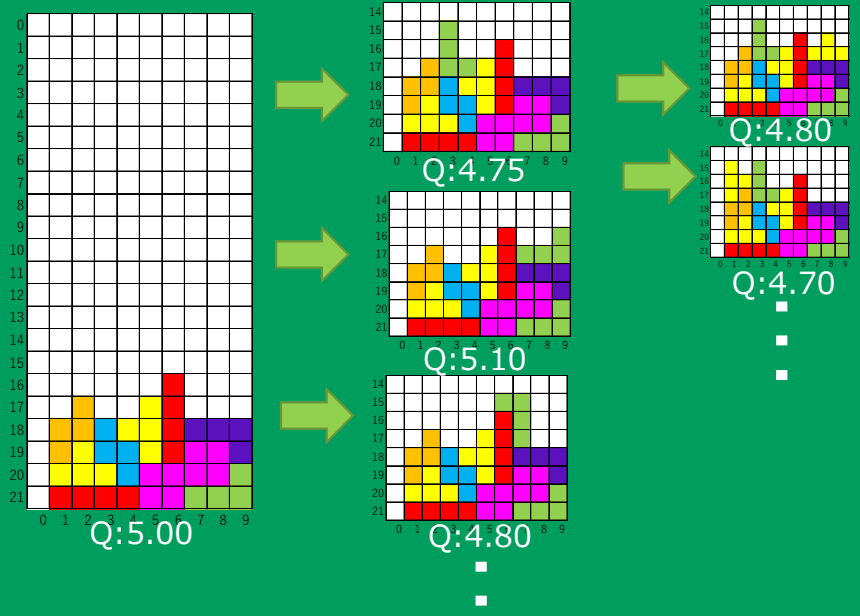
推論時のプログラムを下記のように修正



・上位5局面について次の手を読むようにした。

・5手先まで予告があるが、そこまで読むと大会規定の1秒に推論が間に合わないので、4手先に限定。

・最初は4手先の評価値を最も高いものとしていたが、テトリス(4 lines 消し) が途中にあると4手先の評価値が急降下することがあるため、評価値は手順上のMAX値とした

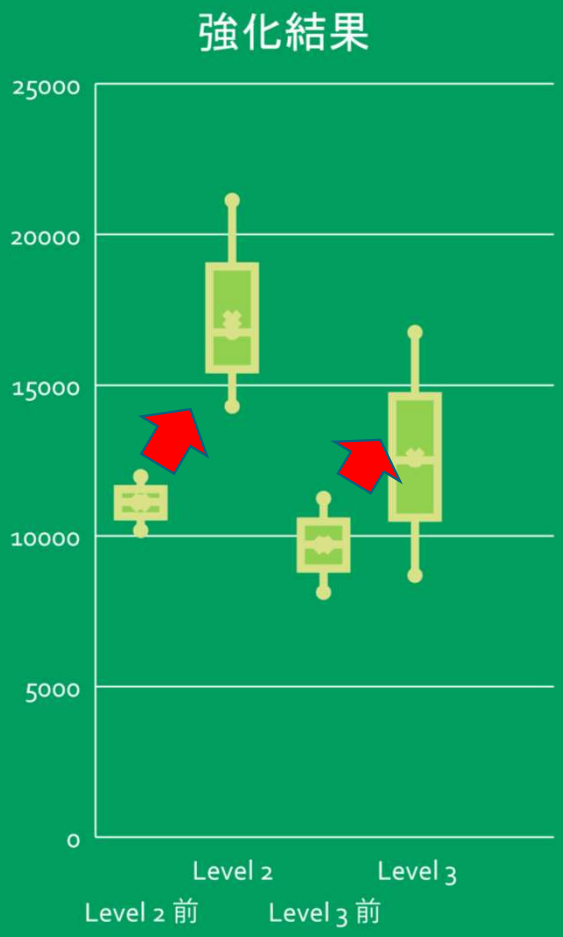


将来は….

・枝刈りと呼ばれる分野のプログラムとなるが、評価値による最適化も可能かもしれない…。

・時間規定による推論中止を導入すればもっと深く読めそう。(PC能力による差を吸収できる！)

結果



強化の結果大幅に点数は向上した。

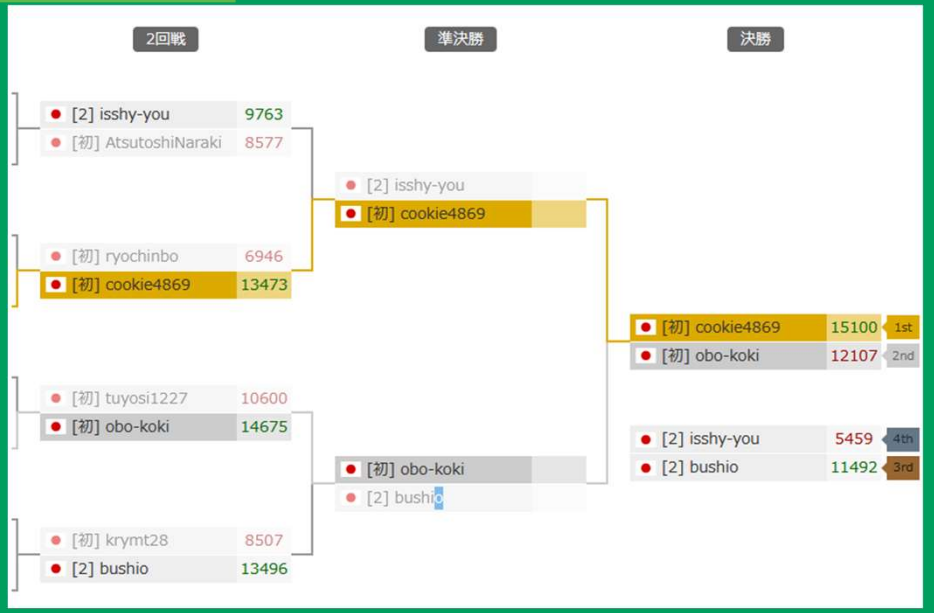
前回大会の Level 2 優勝点数が 12000 だったことを考えると大幅な向上だった。

また、大会では Level 2 のブロック乱数がかなり厳しかったが何とか Level2, Level3ともに優勝することができた。

※Level 3 は局面による Model 切り替え法を採用している。

Level2 大会

Level3 大会



まとめ

- Tetris で Pytorch を使ったAIの機械学習を学ぶことができた
- 機械学習の一連の手順、およびさまざまなパラメータを学ぶことができた。
- 機械学習は時間をかければ、シンプルな報酬体系でも最後はそれなりの学習効果が出る可能性があるが、現実的な時間で対応するには、複雑な報酬体系が必要
- 規定時間内での処理を考えると、まだまだ推論の仕方に最適化の余地がある

- 暑い夏は機械学習には不向きである！！