StarCraft AI における Deep Q-Network を用いたユニットコントロールの提案

A proposal of learning with Deep Q-Network for StarCraft Unit control AI

中田 季利 † 新井 イスマイル †
Kiri Nakata Ismail Arai

1. はじめに

StarCraft: Brood War[1] は、Blizzard Entertainment が開発したリアルタイム・ストラテジーゲーム (以下 RTS ゲーム) である。RTS ゲームは広大な状態空間上の非完全情報ゲームであり、またリアルタイムで迅速な意思決定が必要である。このため、人工知能の研究対象として大きく注目されており、2010 年から StarCraft AI Competition[2] という大会も開かれ、2014 年には 18 チーム登録されるなど大きな盛り上がりを見せている。また、各大会に出場した AI のバイナリやリプレイが公開されていたり、BWAPI[3] という AI 開発者向けの API が用意されている。

既存の戦闘 AI における弱点として、不利な地形への誘い込みへの弱さや、味方ユニット間の位置関係の悪さが指摘されている。既存の強化学習を用いた手法では、地形や味方ユニットとの位置関係を最大限利用できていないためである。これを解決するために、Convolutional Neural Network(CNN) を活用し大きな状態空間上で直接 Q 学習が可能となる手法である Deep Q-Network を適用する新たな試みを提案する。

2. StarCraft について

StarCraft は、ターン制ではなくリアルタイムに進行し、戦場の指揮官となって兵士 (ユニット) を指揮して戦うゲームである。予め幾らか用意されている戦場 (マップ) 上で対戦を行う。プレイヤーは Terran・Zarg・Protoss の 3 つの種族から 1 つを選択し、その種族からなるユニットを操作し戦うことになる。ユニットには数種類あり、その種類ごとに初期の HP・エネルギー・アーマー・攻撃力などが割り当てられている。HP が 0 になるとユニットが破壊され、勝利とは全ての敵ユニットの HP を 0 にすることである。

StarCraft は複雑なゲームであり、以下に示す探索・生産・建設・戦闘などさまざまな要素が絡んでいる。

- 探索
  ゲーム中プレイヤーは、味方ユニット付近の情報しか得ることができない。偵察機などで偵察し、敵の状況や資源の探索をする必要がある。
- 生産
  マップ上に存在する有限資源である、ミネラルやガスを採掘して、味方ユニットを生産したり、研究を進めてユニットを強化することができる。
- 建設
  資源を消費して、マップ上にユニット生産施設などを建設することができる。
- 戦闘
  ユニットの種類毎に設定されている射程内に敵ユニットが入っているとき、その敵ユニットに攻撃を行うことができる。スキルを持たない基本的なユニットの場合、扱える操作は移動のみとなる。

3. 関連研究

3.1 陣形を考慮した StarCraft AI

StarCraft の戦闘におけるユニット制御に、武田家の八陣形のひとつである横陣隊列を取り入れる手法 [1] が報告されている。実際のゲームではほとんどの場合で複数ユニット対複数ユニットの対戦となる。敵ユニットの数だけ火力が分散してしまうため、同じ性能のユニット同士での戦闘の場合、戦闘に参加しているユニット数がとても重要な要素になる。ユニットは重なりあうことができないため、敵を細い道に誘い込んで、広い出口から迎え撃つような戦術も存在する。これらからわかるように、戦場として選ぶ場所取りやユニットの位置関係は勝敗に大きく起因する要素である。

また既存の StarCraft AI の分析として、直線的な移動をしがち、誘導行動に弱いなどの弱点が指摘されている。敵の居る方向に兵士を垂直方向に広げて配置する横陣という陣形を取り入れた AI は、陣形を考慮せず直線的に移動する AI より強いことが実験で示されているが、狭い通路では横陣が乱れてしまい勝率が下がることも報告されている。

† 明石工業高等専門学校 電気情報工学科, Department of Electrical and Computer Engineering, National Institute of Technology Akashi Japan

[1] http://us.blizzard.com/en-us/games/sc/
[2] http://www.sscaitournament.com/
[3] http://bwapi.github.io/

3.2 ポテンシャル流れを利用した AI

ポテンシャル流れを利用して、敵ユニットを囲むように味方ユニットを配置する手法 [2] が提案されている。マップ上に工夫した移動方向のベクトル場を作ることによって、敵に近づきすぎず、孤を描くように味方ユニットを並べることによって最大限の火力が得られると報告されている。これは、前節で示されている問題点の別の解決法だと捉えることができる。しかし、建造物などを含む複雑な地形のような、より現実的なシチュエーションにも対応する必要があると結論付けられている。

3.3 Deep Q-Network

Deep Q-Network を用いて Atari 2600 のゲームプレイを学習した結果 [3] が報告されている。Deep Q-Network は、Q 学習における行動価値観数 $Q(s,a)$ を CNN によって近似することで、非常に大きな状態空間に Q 学習を適用する手法である。また、Experience Replay という、行動の結果を即時に学習に反映せずに Replay Memory に貯めておき、そこからランダムに取り出したサンプルを用いて学習する手法も取り入れている。CNN とは、2 次元グリッドの情報を 2 次元のまま畳み込みを繰り返して、グリッド上の相対位置によらない特徴の抽出を行うニューラルネットワークである。Atari 2600 のゲームにおいて、最新 4 フレームのゲーム画面を 110x84 の解像度にダウンサンプリングし、さらにグレースケール化した画像を入力に与える。出力はコントローラーへの入力の行動価値となる。各イテレーションでは最も評価値が高い行動を選択するが、一定確率で行動価値を無視して完全にランダムな行動を選択する。また、報酬は各ゲーム毎に定義された獲得スコアを与えている。

Deep Q-Network の詳細なアルゴリズムは以下となる。

1. Replay-Memory D をある大きさ N で初期化する
2. ニューラルネットワークをランダムな重みで初期化する
3. 各イテレーションについて
   (a) 観測された初期状態 $s_1 = \{x_1\}$ からエンコードされた入力 $\phi_1 = \phi(s_1)$ を作る
   (b) $t = 1$ から T までについて (ただし s_T がゲームの終了状態とする)
      i. ϵ の確率で a からランダムな行動 a_t を選択する
      ii. それ以外の場合は、$Q^*(\phi(s_t), a; \theta)$ が最大となるような a_t を選択する
      iii. 行動 a_t を実行し、報酬 r_t と次の入力 x_{t+1} を得る
      iv. s_t, a_t, x_{t+1} を用いて入力 $\phi(t+1)$ をエンコードする
      v. $(\phi_t, a_t, r_t, \phi_{t+1})$ を D に格納する
      vi. D からランダムに 1 つの minibatch である 4 つ組 $(\phi_j, a_j, r_j, \phi_{j+1})$ を取り出す
      vii.
$$y_j = \begin{cases} r_j(\text{終了状態の場合}) \\ b(\text{終了状態でない場合}) \end{cases}$$
      viii. minibatch と y_j を用いて Q 値を更新する

このように構築された Deep Q-Network で 1000 万イテレーション分学習を行った結果、Breakout、Pong、Enduro では人間に勝利できるほど学習ができることが示されている。これらは比較的単純なゲームで、一方、Space Invaders のように比較的長期的な戦略や先読みが必要なゲームでは良い成績にならないことが報告されている。Deep Q-Network を用いて、ロボットの移動制御などを行う応用 [4] などが発表されており、さらなる発展が望まれるが、StarCraft や他の RTS ゲームに適用された報告はまだなされていない。

以上の関連研究から、戦闘において地形や味方ユニットとの位置関係が重要であり、それらを考慮して行動を決定することが必要だと考え、その改善のために Deep Q-Network を利用することを考えた。

4. 提案手法

Wender ら [4] のように、学習を用いる既存のアルゴリズムでは、HP や武器のクールダウンタイムなどの入力は用いているが、味方同士の位置関係やマップの情報をほとんど使っていない。そこで、Deep Q-Network を用い、それらの情報をグリッドマップとして与えることによって、そのような要素の学習を行うことができ、不利な地形に飛び込まないことや、有利な立ち位置で攻撃をすることを学習できるのではないかと考えた。

1. 味方ユニットと敵ユニットが近接しているマップ上の空間を戦闘空間と扱う。どのユニットが戦闘に参加するかは、より上位の AI モジュールで決定されるべきものなので、本研究では決定されているものと扱う。
2. あるユニットを中心に適当な大きさ (WxH とする) のグリッドを戦闘空間として扱う。
3. 各セルに以下の情報を含んだグリッドマップを作成する。

[4] http://research.preferred.jp/2015/06/distributed-deep-reinforcement-learning/

地形

- 侵入可能であるか
- 攻撃を通過させるか
- 標高

ユニットについて

- ユニットの勢力 (味方/敵)
- ユニットの HP
- ユニットの装甲値
- ユニットのサイズ
- 武器のクールダウン値

4. 3 のグリッドを CNN の入力とする。
5. Deep Q-Network の出力は、縦横斜め 8 方向への移動価値となる 8 ノードとする。

このように構築した Deep Q-Network を用い、イテレート学習を行う。

Deep Q-Network の実装には、Deep Learning のフレームワークである Caffe[5] や Chainer[6] などを利用することができる。また、StarCraft の AI 開発用 API である BWAPI や、それを拡張した BWSAL[7] を利用することで簡単に AI を作成することができる。

5. 評価方法

基本的なシチュエーションとして、同数の単一ユニット同士での対戦から検証することを考えている。Protoss の基本ユニットであり、近接戦闘をメインとする Zealot というユニットを複数体用意し、Deep Q-Network を用いた AI とそうではない AI で対戦するシチュエーションで、大量にイテレーション学習する。学習終了後、100 回ほど対戦をさせて勝率を求める。相手 AI は、Albertabot など一般に公開されているものを選ぶ予定である。

どの程度のイテレーション数でどれほどの効果があるかの検討をつけるのが難しい。500 万イテレーション、1000 万イテレーションと学習を進めるごとに対戦実験をし、どの程度のイテレーションが必要かの検証も同時に行う予定である。

6. おわりに

StarCraft の既存のユニットコントロールにおいて、Deep Q-Network を用いて評価関数を学習することによって勝率を高める手法を提案した。

今後は、Deep Q-Network のパラメータや入力の与え方、また畳み込みの方法について深く検証を行っていく予定である。

[5] http://caffe.berkeleyvision.org/
[6] https://github.com/pfnet/chainer
[7] https://code.google.com/p/bwsal/

参考文献

[1] 鎌田徹朗, 橋本剛, 高野誠也StarCraft AI への隊列導入. 情報処理学会研究報告, Vol.2015-GI-33, No.10, pp.1–6 2015.

[2] Tung, Kien, and Ruck. Potential flow for unit positioning during combat in starcraft. 2013 IEEE 2nd Global Conference on Consumer Electronics (GCCE 2013), pp. 10–11. 2013.

[3] Volodymyr, Koray, David, Alex, Ioannis, Daan, and Martin. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[4] Wender and Watson. Applying reinforcement learning to small scale combat in the real-time strategy game starcraft: broodwar. IEEE Conference on Computational Intelligence and Games (CIG 2012), pp. 402–408. 2012.