Nタプルネットワークの大きさと学習性能の関係: ミニ2048を用いた実験・評価

寺内 俊輔[a)]　松崎 公紀[b)]

概要：2048では、Nタプルネットワークと TD 学習を拡張した手法及び Expectimax 探索の組み合わせにより優れたプレイヤが作られている。本研究では、ミニ 2048 において構築可能な 1 タプルから 9 タプルまでの各タプル全てからなる組み合わせと、それぞれにおける妥当な形状を用いて、N タプルネットワークの大きさと学習性能の関係を実験的に評価する。特に最先端プレイヤの開発で用いられた手法の観点から、その効果と特性について考察を行う。

キーワード：2048, N タプルネットワーク, Expectimax 探索, ミニ 2048, 強化学習

1. はじめに

「2048」は G. Cirulli によって作られた確率的一人ゲームである [1]．これまでに，2048 のさまざまなコンピュータプレイヤが作られてきた．現在最も成功しているアプローチは，強化学習によってチューニングした N タプルネットワーク評価関数 [2] と Expectimax 探索 [3] を組み合わせるものである．その後, N タプルネットワークと Expectimax 探索の組合せを基礎として，一般的もしくはゲームに特化した改良手法が多数提案されてきた [3], [4], [5], [6]．Guei によって作られた最先端プレイヤは [6] では Expectimax 探索の深さが 6 の場合に平均得点 625 377 を達成した．著者らの研究の大きな目標は，それらの AI 技術のそれぞれについて，どのように働いているかをより詳細に分析し説明することである．本研究では、ミニ 2048 における N タプルネットワークのタプルサイズおよび Optimistic Initialization（OI）の初期値が、プレイヤの性能および探索によるスコアに与える影響について実験的に評価する．

2. ミニ 2048

本研究はミニ 2048 [7] を研究対象として使用する．ミニ 2048 は，3×3 の盤面でプレイされることを除いて，確率的一人ゲーム 2048 と同じゲームである．

2.1 ルール

ミニ 2048 は，3×3 の盤面でプレイされる．初期局面は 9 マスの内どこか 2 マスに 2（確率 0.9）か 4（確率 0.1）の数字タイルがランダムに置かれた盤面からなる．

各局面において，プレイヤは上下左右いずれかの方向を選択する．すると各数字タイルはその方向にできるだけ移動する．移動した結果，2 つの同じ数字のタイルが移動方向に衝突するとこれらは合体してその合計値のタイルとなり，その合計値がスコアに加算される．合体してできたタイルは，同じターンでは別のタイルと合体することはない．例えば，盤面の行が `_2_`，`22_`，`422` であるとき，右を選択するとそれぞれ `__2`，`__4`，`_44` へと変化する．その後，空白のマスのうちのランダムな 1 マスに 2（確率 0.9）か 4（確率 0.1）のタイルが置かれる．

プレイヤはいずれかのタイルが移動または衝突するような方向しか選択することができない．いずれの方向も選択できなくなるとゲームは終了する．このゲームの目標はゲームが終了する前に出来るだけ高得点を獲得することである．

2.2 用語の導入

通常の 2048 と同様にミニ 2048 における 1 ターンは，「移

†1 現在，高知工科大学
Presently with Kochi University of Technology
Presently with
a) 295141a@gs.kochi-tech.ac.jp
b) matsuzaki.kiminori@kochi-tech.ac.jp

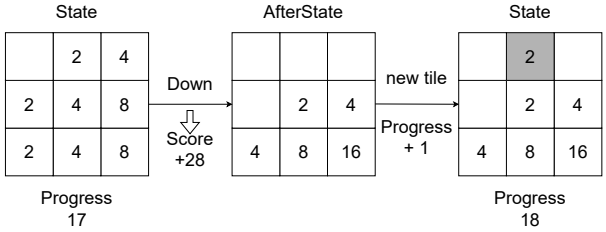


図 1: state，afterstate，progress の例

動・合体ステップ」と「新規タイルステップ」の 2 ステップからなる．これらステップの前後の状態を区別するため，以下の用語を導入する．

state　プレイヤが手を選択する盤面状態（とスコア）を *state* と呼ぶ．

afterstate　プレイヤが手を選択してタイルが移動・合体した直後の盤面状態（とスコア）を *afterstate* と呼ぶ．すなわち，afterstate は新規タイルが出現する前の盤面状態である．

（通常の 2048 同様に）ミニ 2048 では，新しく出現するタイルはランダムに 2 か 4 の値をとる．そのため，単純にターン数をゲームの流さや進行度の指標に用いるには不都合がある．この問題を解決するため，本研究では以下の指標を用いる．

progress　タイルの値の合計値の半分を *progress* [8] と呼ぶ．progress は，1 ターンで 1（新規タイルが 2 の場合）または 2（新規タイルが 4 の場合）だけ増加する．

図 1 は，初期局面から始まるゲームの流れにおいて，state，afterstate，progress について図示したものである．

2.3　完全解析とその結果

ミニ 2048 は確率的一人ゲームであり，その完全解析とは各状態に対して期待スコア求めることである．ミニ 2048 は，到達可能な状態数が 10^9 以下と小さいため，現実的な時間で完全解析ができる．山下ら [7] は，ミニ 2048 の完全解析に最初に取り組み，そこでは幅優先探索による状態列挙と，列挙した状態を用いる後退解析を行った．また，著者ら [8] も完全解析の追試を行い，深さ優先探索による後退解析で，結果の正しさを確認した．

完全解析の結果について，重要なものを以下に示す．初期状態のいずれかから到達可能な state の数は 48 713 519，afterstate の数は 31 431 374 である．初期状態の期待スコアは，5 468.49 である．各 afterstate に対する期待スコアを格納したものを valueDB と呼ぶ．

完全解析で得られる valueDB を用いると，各局面において最適な手を選択するパーフェクトプレイヤを実現できる．ただし，ミニ 2048 は確率的一人ゲームのため，決定的に最善手を選択するパーフェクトプレイヤであっても，ゲームごとにプレイの結果は異なることに注意が必要である．図 2 は，パーフェクトプレイヤが 1 万ゲームを行った際の，progress ごとの生存率とゲーム終了時のスコアを示している．表 1 は，パーフェクトプレイヤが 256，512，1024 タイルに到達したときの進捗状況，スコア，生存率を示している．パーフェクトプレイヤでも，512 タイルに到達した後，1024 タイルに到達する前は，生存率が急激に低下することが分かる．図 2 より，生存率が急激に下がるタイミングがいくつかある．本研究では，そのような生存率が下がる部分を**難易度の高い領域**と呼ぶ．

表 1: Progress, score, and alive ratio of perfect player

Condition	Progress	Score	Alive
256-tile	136	1 750	99.53%
512-tile	263	4 000	73.84%
512-tile & 256-tile	391	5 750	54.40%
512-tile & 256-tile & 128-tile	456	6 500	40.49%
1024-tile	511	9 000	1.07%

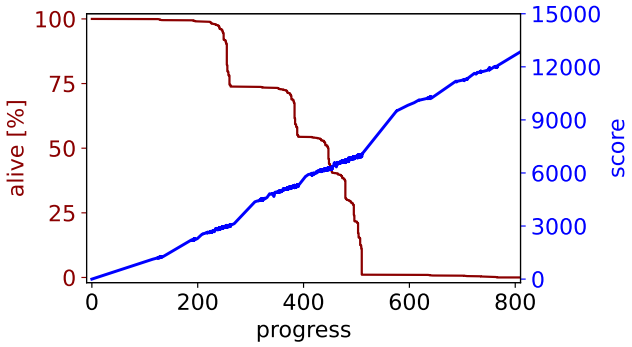


図 2: パーフェクトプレイヤの生存率とスコア [8]

3.　本研究で用いるプレイヤ

3.1　N タプルネットワーク評価関数

2048 における最も成功したプレイヤの多くは、N タプルネットワークに基づく評価関数を強化学習によってチューニングするアプローチを採用している [2]。Guei らの最新のプレイヤ [6] も、Matsuzaki [9] が提案したタプルの組合せをベースに、Expectimax 探索や Multistaging[3]、Optimistic Initialization、Tile Downgrading[6] などの改良を加えることで高い性能を達成している。

本研究では、こうした知見を踏まえ、ミニ 2048 において 1 タプルから 9 タプルまでの N タプルネットワークを構築可能な全てのタプル列挙を行い、

- 我々が妥当と判断した形状のみを選んだタプル集合
- 各サイズごとの連続タイルすべてを使用したタプル集合

という 2 通りの設計方針でネットワークを構築した。

この結果、最大で 18 種類の N タプルネットワークが得られたが、1 タプル、2 タプル、9 タプルにおいては両方の設計が一致したため、プレイヤの種類としては 15 通りと

なった。これらのプレイヤを用いて、N タプルネットワークの大きさ（タプル数）と学習性能の関係を実験的に評価した。

表 2 に、本研究で使用したタプルの構造を示す。また、ミニ 2048 の盤面の持つ対称性（回転・反転）**を活用し、1 つのタプルに対して対称な 8 通りの位置からのサンプリングを行うことで、学習に必要なタプル数の削減を図っている。

N タプルネットワークの重みは，afterstate 間の評価値の差に基づく TD 学習法の改良手法によって調整した．本研究で用いる N タプルネットワークの学習では，以下の技術を用いた．

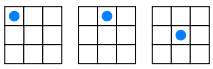
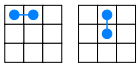
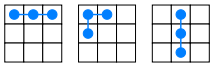
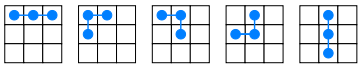
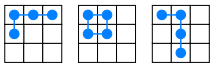
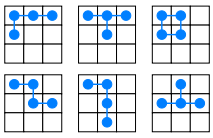
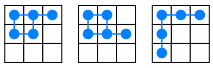
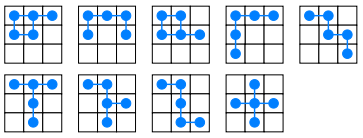
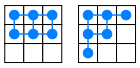
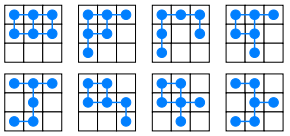
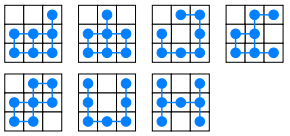
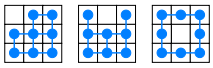
Multistaging　ゲームの進行に応じて重みを参照するテーブルを切り替える．本研究では，2 ステージとし，512 のタイルができる前後でステージを分けた．

Temporal coherence 学習（TC 学習）　TC 学習は学習率自動調整機能を備えた TD 学習で，Jaśkowski [5] が始めて 2048 に導入した．

Optimistic initialization　学習段階での探索を広く行うために，重みを（ゼロではなく）大きな値で初期化する．本研究で用いた N タプルの学習では，すべての afterstate の初期値が 1200 になるように重みを初期化してある．

それぞれの N タプルニューラルネットワークに対して，5×10^8 局面分のデータで学習を行った．いずれのニューラルネットワークも，十分に学習が収束していることを確認してある [8]．

表 2: タプルサイズと形状の一覧

タプル (タプルナンバー)	タプルの形状
1 タプル (6)	
2 タプル (12)	
3 タプル (144)	
3 タプル (2673)	
4 タプル (301)	
4 タプル (44755)	
5 タプル (298)	
5 タプル (896673)	
6 タプル (16)	
6 タプル (26835)	
7 タプル (0)	
7 タプル (248)	
8 タプル (0)	
8 タプル (6)	
9 タプル (0)	

4. Expectimax 探索

Expectimax 探索は，確率的一人ゲームにおける標準的な探索手法である．ミニ 2048 のゲームの進行は，state におけるプレイヤの選択と，afterstate における新規タイルの出現が交互に起こる．したがって，ミニ 2048 のゲーム木は，根が現在の state に対応し，根から葉への各パス上に，afterstate に対応するノード（chance ノード）と state に対応するノード（max ノード）が交互に現れる．本研究では，ミニ 2048 のゲーム木の高さ（探索の深さ）を，各パス上の afterstate に対応するノードの数と定める．例えば，高さ 2 のミニ 2048 のゲーム木は，根，afterstate に対応するノードの層，state に対応するノードの層，afterstate に対応するノードの層，の合計 4 層からなる（図 3）．

Expectimax 探索では，ゲーム木の各ノードに対して次のように再帰的に計算を行う．

- max ノードでは，子要素の値のうちの最大値を計算する．
- chance ノードでは，子要素の値を，その出現確率を用いた重み付き平均を計算する．

Expectimax 探索プレイヤは，Expectimax 探索によって得られた子ノードのうち，評価値の最も大きなものを選択する．

図 3 に深さ 2 の Expectimax 探索の例を示す．

ミニ 2048 のゲーム木では，特に，同じ afterstate が複数出現する．そのような同じ afterstate をまとめる（合流）

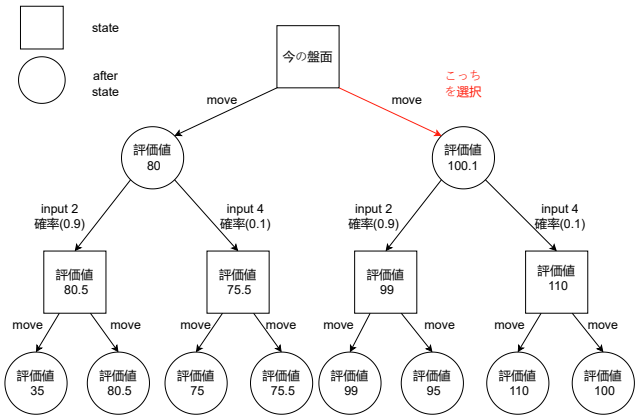


図 3: 深さ 2 の Expectimax 探索の例

工夫を実装した．ただし，合流を考慮しない Expectimax と結果が一致するよう，同じ afterstate であってもゲーム木中の深さが異なる場合には別のものとして扱った．この工夫により，特に深い探索において大幅な高速化が実現された．本研究では [10] で実装した Expectimax 探索を用いた．

5. 実験と分析

本研究では、ミニ 2048 における N タプルネットワークの学習性能を多角的に分析するため、構造の異なる複数のプレイヤを構築し、Greedy および Expectimax 探索による評価を行った。

2 で示した 15 種類のプレイヤを用いて評価を行った。

さらに、Optimistic Initialization（OI）の影響を調べるため、各プレイヤについて OI の初期値を 0、1200、5400 に設定し、それぞれ学習を実施した。すべてのプレイヤは 5×10^8 手分の行動に基づいて学習を行い、乱数シードを変えて 10 体ずつ学習させた。結果として、15 タプル構成 ×3 回（OI の初期値）×10 回（シード）で，計 450 体のプレイヤが作成された。

これらのプレイヤに対して、Greedy プレイおよび Expectimax 探索（深さ 2〜5）による 1000 ゲームプレイを行い、そのログを用いて解析を行なった。

図 4 および 5 には、Greedy プレイと Expectimax 探索（深さ 5）におけるタプル数の変化によるスコアの変化を示す。TN は各タプルにおして平均値の高かったものを選択している。

図 4 を見ると、Greedy プレイにおいては NT5 までのタプル数でスコアが向上していることがわかる。NT5 NT7 まで OI=1200 だけ横ばい、それ以外は下落している、NT8 以降はすべてスコアが低下している。これは図 5 の Expectimax 深さ 5 も同様の傾向が見られ、図では示していないが、Expectimax 深さ 2 4 についても同様の傾向が見られた。また NT6 以降のスコアは Greedy、Expectimax 深さ 2 5 のいずれも OI=1200 のスコアが高くなっている。これは OI=1200 の初期値が効果的に作用した結果であると考えられる。

ではそれぞれのタプルについて、OI の値が変わることでどのようなスコアの変化があったか、またそれらを用いて Expectimax 探索を行った場合にどのようなスコアの変化があったかをについて詳しく見ていく。

NT1 と NT2 については OI の値が変わってもスコアの変化はほぼ見られなかった。Expectimax 探索の深さを深くした場合については、スコアは探索の深さに比例して上昇している。図 6 を見ると NT3 は TN=2673 の方がいずれの OI でも TN144 と比べてスコアが高くなってる。探索の深さを深くした場合については、同様にスコアは探索の深さに比例して上昇しているが上下関係は変わらない。図 7 を見ると NT4 は TN=44755、OI=0 のスコアが最も低く、それに続いて TN=301 の各 OI の値が続き、TN=44755 の値が続いている。探索の深さを深くした場合スコアの順序が入れ替わっている。図 8 を見ると NT5 は TN=896673 が高く、TN=298 が低い傾向にあり、Expectimax 探索の深さを深くした場合で少しスコアの差が縮まっている。図 9 を見ると NT6 は OI ごとのスコアがちかく、TN 間の差がほぼない、これは TN=16 で盤面の情報を保存するのに十分なパラメータ数があるからだと考えられる。探索の深さを深くした場合の変化は小さいがスコアの向上が見られた。図 10、図 11 を見ると NT7、NT8 についてはこれまでの傾向と異なり TN=0 の方がスコアが高くなっている。これにより、パラメータ数を増やして一つの盤面から得られる情報を増やしても得点の向上につながらないことが分かる。また NT8 と NT9 は NT7 と比べて明確にスコアが低下している、これは盤面の汎化性能が低下することが原因だと考えられる。NT9 については OI=1200 のスコアが高くなっている。

全体の傾向として OI の値が探索に与える影響は感じられなかったが、OI=1200 のスコアが高くなっていることから OI の初期値が高すぎず低すぎない適切な値に設定することがスコアの向上に寄与していることが分かる。またタプルサイズを増やすことやタプルの数を増やすことは、一定程度までは得点に寄与するが、あるラインを超えると逆にスコアが低下することが分かる。

NT5 の TN の違いによってどのように変化するのかを見ていく、図 12 は OI=1200 の Greedy プレイの場合は、TN=896673 の方がスコアが高くなっている。正確度は TN896673 の方が少し高くなっているが、絶対誤差はほぼ差がない。生存率を見ると progress=250 辺りの難易度の高い領域で生まれた生存率の差が、正確度の差分を見ると、0.2 ポイントほど progress300 辺りで差があるが、progress250 辺りの難易度の高い領域では正確度の差分があまりないことがわかる。平均得点に 200 点ほどの差があるが、これは絶対誤差を見るに progress=250 辺りの少し差がある所の

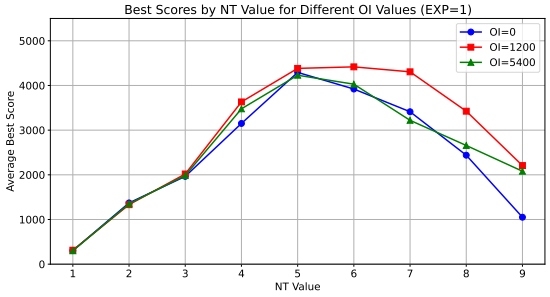


図 4: タプル数の変化によるスコアの変化（Greedy プレイ）

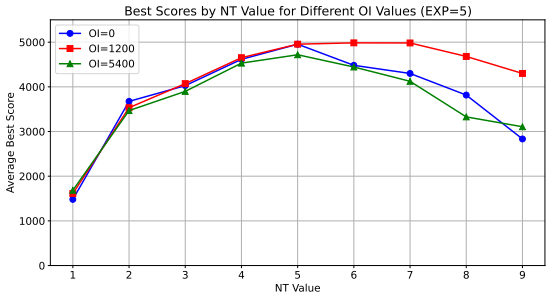


図 5: タプル数の変化によるスコアの変化（Expectimax 深さ 5）

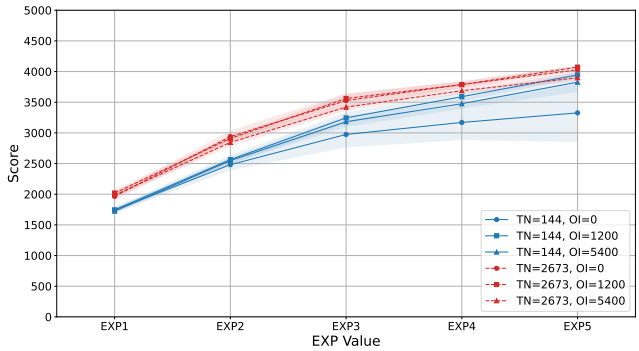


図 6: 実験結果の傾向 NT3

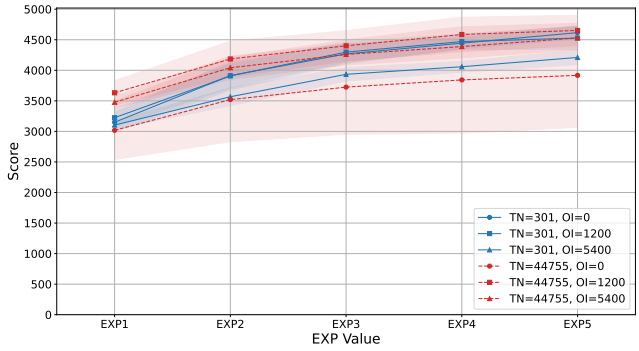


図 7: 実験結果の傾向 NT4

積み重ねがこの差になっていると考えられる。

NT8 の TN の違いによってどのように変化するのかを見ていく、図 13 は OI=1200 の Greedy プレイの場合は、

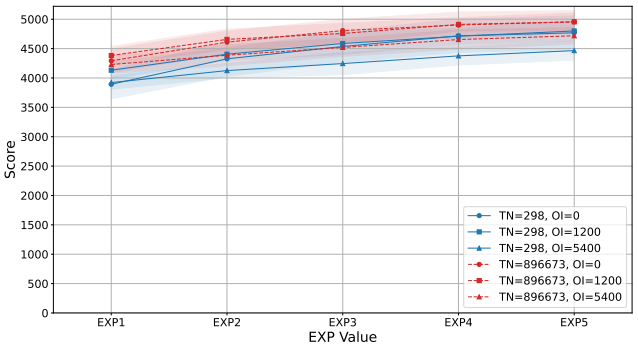


図 8: 実験結果の傾向 NT5

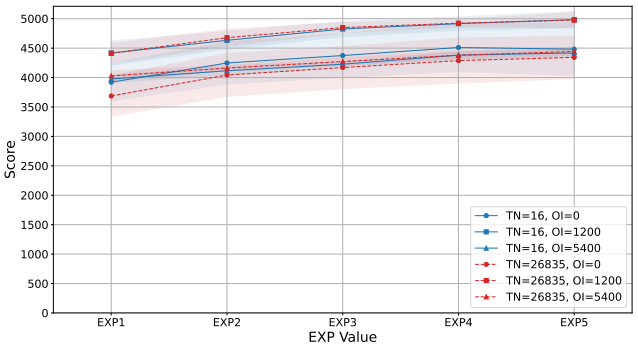


図 9: 実験結果の傾向 NT6

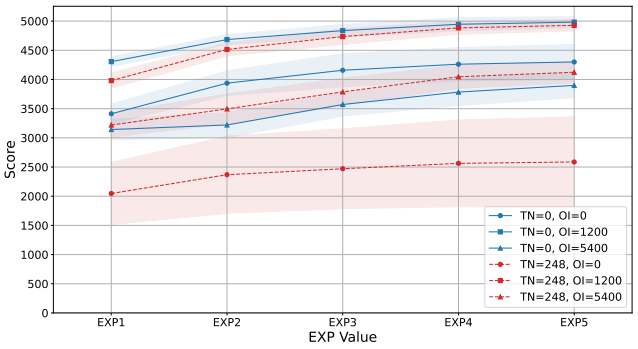


図 10: 実験結果の傾向 NT7

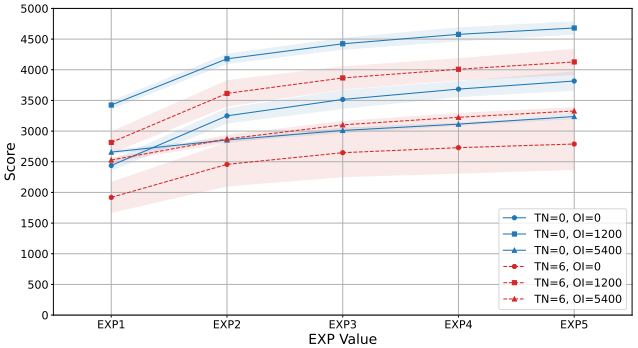
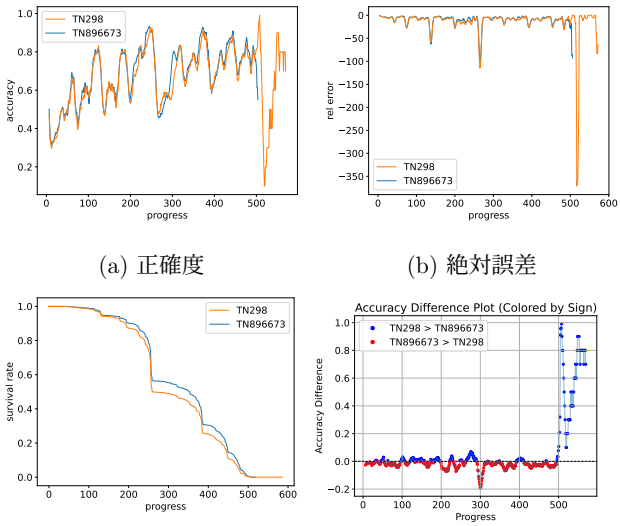


図 11: 実験結果の傾向 NT8

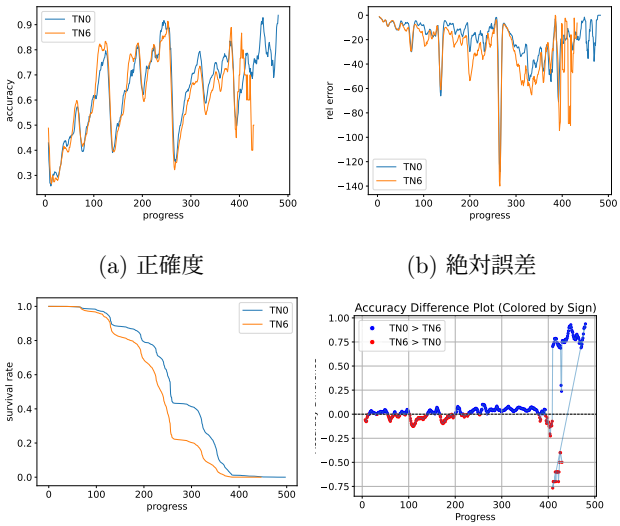
TN=0 の方がスコアが高くなっている。正確度と正確度の差分を見ると、序盤の progress100 130,170 200 辺りに TN6 の方が正確度が高いが、そこの絶対誤差を見ると TN0 の値がほぼ高い。これを見ると、TN0 は間違った選択をし



(a) 正確度　(b) 絶対誤差

(c) 生存率　(d) 正確度の差分

図 12: 5 タプルネットワーク（EXP1）の評価結果



(a) 正確度　(b) 絶対誤差

(c) 生存率　(d) 正確度の差分

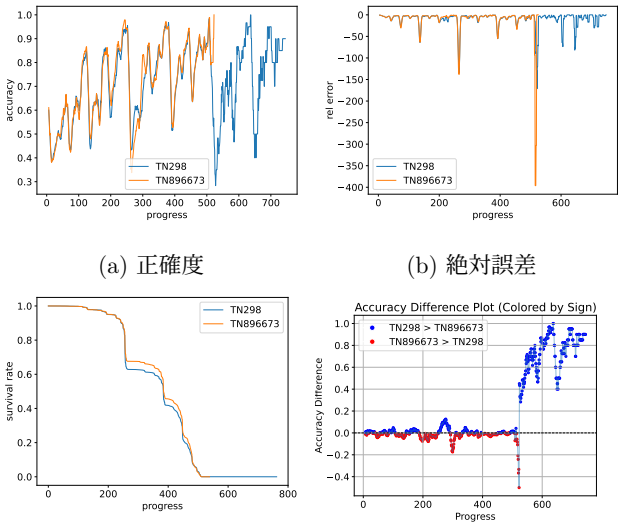
図 13: 8 タプルネットワーク（EXP1）の評価結果

ても評価値はあまり変わらず。TN6 の場合は間違った選択をすると評価値が大きく変わることが読み取れる。

NT5 の EXP5 の TN の違いによってどのように変化するのかを見ていく、図 14 は OI=1200 の Greedy プレイの場合は、TN=896673 の方がスコアが高くなっている。正確度は TN896673 の方が高くなっている progress が多いが、絶対誤差はほぼ差がない。生存率を見ると progress=250 辺りの難易度の高い領域少し差が生まれているが、平均得点の差が生まれる要因がどこにあるのか分からない。

NT8 の EXP5 の TN の違いによってどのように変化するのかを見ていく、図 15 は OI=1200 の Greedy プレイの場合は、TN=0 の方がスコアが高くなっている。正確度は TN0 の方が高くなっている progress が多いが、絶対誤差はほぼ差がない。正確度の差分を見ると波のような形で序盤の正確度が上下に揺れているが、その間に生存率の差が生まれている。絶対誤差のグラフを見ると、TN0 の方が序盤少し高い場面が多いことが分かる。

これらの結果から TN が変わってもプレイヤの動きにはあまり変化がなく学習が正確な判断ができる場合とそうではない場合があるが、良いタプルの組み合わせを選ぶことで、何もないような盤面でのミスが減ることが分かった。またタプルの数を増やすことは 6 タプル以上の場合有効ではないことも確認できた。



(a) 正確度　(b) 絶対誤差

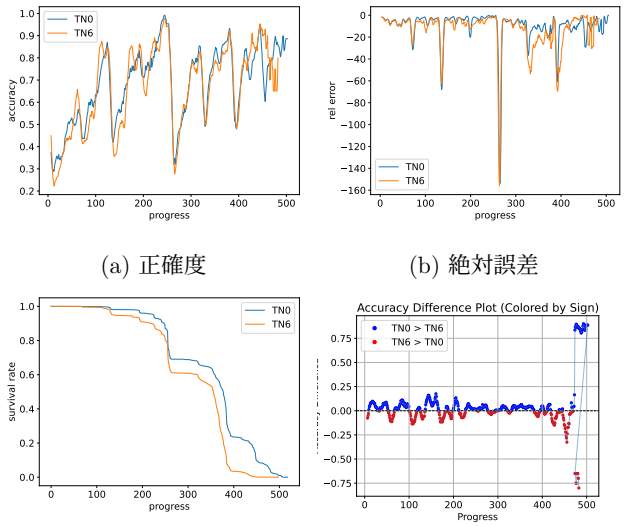
(c) 生存率　(d) 正確度の差分

図 14: 5 タプルネットワーク（EXP5）の評価結果

6. まとめ

本研究では、ミニ 2048 における N タプルネットワークのタプルサイズおよび Optimistic Initialization（OI）の初期値が、プレイヤの学習性能および探索によるスコアに与える影響について詳細に分析した。実験の結果、以下の重要な知見が得られた：

- タプルの数やサイズを適切に増やすことでプレイヤの性能は向上するが、NT6 を超えるとその効果は限定的となり、過剰なパラメータ数は汎化性能の低下を招く可能性がある。
- OI の初期値は、学習初期における探索の幅を調整する上で有効に機能し、特に OI=1200 は多くの構成において高いスコアを示した。過大な初期値（OI=5400）



(a) 正確度　(b) 絶対誤差

(c) 生存率　(d) 正確度の差分

図 15: 8 タプルネットワーク（EXP5）の評価結果

やゼロに設定するよりも、中間的な値が最適である可能性が示唆された。

- タプルの組み合わせ（TN）の違いによっても性能差が生じ、一部の構成では学習済み評価関数が特定の局面において不安定な判断を行っていた。より良い TN の選定によって、盤面評価の精度が改善され、不要なミスの低減につながることが確認された。

謝辞　本研究は JSPS 科研費 JP23K11383 の助成を受けたものである.

参考文献

[1] G. Cirulli. 2048. `http://gabrielecirulli.github.io/2048/`, 2014.

[2] M. Szubert and W. Jaśkowski. Temporal difference learning of N-tuple networks for the game 2048. In *2014 IEEE Conference on Computational Intelligence and Games*, pages 1–8, 2014.

[3] K.-H. Yeh, I-C. Wu, C.-H. Hsueh, C.-C. Chang, C.-C. Liang, and H. Chiang. Multi-stage temporal difference learning for 2048-like games. *IEEE Transactions on Computational Intelligence and AI in Games*, 9(4):369–380, 2016.

[4] K. Matsuzaki. Developing 2048 player with backward temporal coherence learning and restart. In *Proceedings of Fifteenth International Conference on Advances in Computer Games (ACG2017)*, pages 176–187, 2017.

[5] W. Jaśkowski. Mastering 2048 with delayed temporal coherence learning, multi-stage weight promotion, redundant encoding and carousel shaping. *IEEE Transactions on Computational Intelligence and AI in Games*, 10(1):3–14, 2018.

[6] Hung Guei, Lung-Pin Chen, and I-Chen Wu. Optimistic temporal difference learning for 2048. *IEEE Transactions on Games*, 14(3):478–487, 2022.

[7] 山下 修平, 金子 知適, and 中屋敷 太一. 3 × 3 盤面の 2048 の完全解析と強化学習の研究. In **第 27 回ゲームプログラミングワークショップ** *(GPW-22)*, pages 1–8, 2022.

[8] Shunsuke Terauchi, Takaharu Kubota, and Kiminori Matsuzaki. Using strongly solved Mini2048 to analyze players with N-tuple networks. In *2023 International Conference on Technologies and Applications of Artificial Intelligence (TAAI 2023)*, 2023.

[9] K. Matsuzaki. Systematic selection of n-tuple networks with consideration of interinfluence for game 2048. In *Proceedings of the 2016 Conference on Technologies and Applications of Artificial Intelligence (TAAI 2016)*, 2016.

[10] 寺内 俊輔 and 松崎 公紀. ミニ 2048 の完全解析を用いた n タプルネットワーク+expectimax 探索プレイヤの分析. In **情報処理学会プログラミング・シンポジウム予稿集** *(プログラミング・シンポジウム予稿集)*, pages 83–90, 2024.