対戦型2048におけるニューラルネットワークプレイヤの$\alpha\beta$探索による強化

小田　駿斗[1,a)]　松崎　公紀[2,b)]

概要：「対戦型 2048」は，確率的一人プレイヤゲーム「2048」を二人プレイヤゲームに拡張したものである．対戦型 2048 は，攻撃側と防御側でプレイが非対称であるという特徴を持つ．対戦型 2048 プレイヤにはいくつかの実装手法が報告されている．著者らはこれまでに，攻撃側プレイヤが 2 と 4 のタイルのどちらを置くかを自由に選べるルールのもとで，強化学習によりニューラルネットワークプレイヤを作成した．本研究では，上記で作成した評価関数と $\alpha\beta$ 探索を組み合わせ，それぞれ探索深さ $d = 1, 2, 3, 4, 5, 6, 7$ の攻撃側プレイヤと防御側プレイヤを総当たりで対戦し，性能の比較を行った．実験の結果より，探索深さを増やすと性能向上が見られること，双方が同じ探索深さでプレイすると深さを増やすごとに平均スコアが減少すること，タイルを動かした後の盤面を評価関数に入力した方が性能が良いことなどが分かった．

Improving Neural-network Players for Two-player 2048 with $\alpha\beta$-search

Hayato Oda[1,a)]　Kiminori Matsuzaki[2,b)]

Abstract: Game “2048” is a stochastic single-player game, and game “two-player 2048” is a two-player variant of 2048. An important characteristic of two-player 2048 is asymmetry between the offense and defense players. There have been several studies on this two-player 2048. The authors also developed neural-network players under the rule that the offense player can place any of the 2- and 4-tile at any position. In this study, we combined $\alpha\beta$-search with the neural-network evaluation functions and evaluated the performance of the players changing the search depth $d = 1, 2, 3, 4, 5, 6$, and 7. From the experiment results, we found that the performance of players improved when we increased the search depth, the average scores decreased when we increased the search depth of both offense and defense players, and we obtained better results when we fed the states after sliding (and before adding a new tile) to the evaluation functions.

1. はじめに

「対戦型 2048」は，寺田 [5] によって提案された，2048 の 2 人プレイゲームへの拡張である．対戦型 2048 は非対称な 2 人ゲームであり，2 人のプレイヤ（攻撃側と防御側）が選択できる手と目標がまったく異なる．防御側プレイヤは，通常の 2048 と同様に，盤面上のタイルを動かす方向を選択し，できるだけ得点が大きくなることを目標とする．攻撃側プレイヤは，空マスの中からタイルを置くマスを選び，できるだけ得点が小さくなることを目標とする．このような非対称なゲームは対称なゲームに比べると研究が少ない．

対戦型 2048 のプレイヤの作成についていくつかの取り組みがある．そのうち，2048 において有効性が示されている N タプルネットワークに基づくプレイヤが岡と松崎 [3] により作成された．一方，2048 において近年ニューラルネットワークを用いたプレイヤの開発で一定の成果が得られている [1]．この結果を受けて，横山と松崎 [6] は，ニューラルネットワークによる評価関数と TD 誤差学習の組み合わせにより対戦型 2048 プレイヤを作成した．その結果を受けて，著者らは [3] と同じルールのもとで同様の評価実験を行った [4]．

[1] 高知工科大学大学院工学研究科
Graduate School of Engineering, Kochi University of Technology
[2] 高知工科大学情報学群
School of Information, Kochi University of Technology
[a)] 255101k@gs.kochi-tech.ac.jp
[b)] matsuzaki.kiminori@kochi-tech.ac.jp

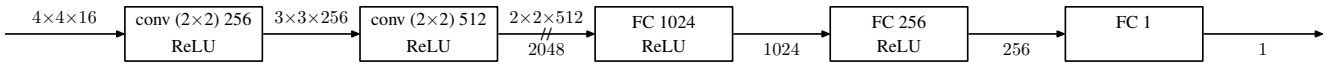

図 1 本実験で用いたニューラルネットワークの構成．“conv (2 × 2) n” は大きさ 2 × 2 の畳み込みフィルタを n 個もつ畳み込み層を表し，“FC n” はニューロン数 n の全結合層を表す．矢印のラベルは中間データの形を表す．第 2 畳み込み層の後に，3 階テンソルを 1 階テンソルにする平坦化処理が行われている．

本研究では，著者ら [4] が作成したニューラルネットワーク評価関数を用いて，$\alpha\beta$ 探索により着手決定するプレイヤを実装した．$\alpha\beta$ 探索における探索深さを $d = 1, 2, 3, 4, 5, 6, 7$ として，総当たりの対戦実験により性能を評価する．実験の結果より得られた知見を以下にまとめる．

- 攻撃側プレイヤ・防御側プレイヤともに，探索深さを増やすと性能向上が見られる．
- 双方が同じ探索深さでプレイするとき，探索深さを増やすと平均スコアが減少する．
- タイルを動かした後の盤面を評価関数の入力とするほうが性能が良い．

2. 対戦型 2048 のルール

「対戦型 2048」は確率的一人ゲーム「2048」を二人ゲームに拡張したものである．対戦型 2048 は完全情報非対称二人ゲームである．ある 1 ゲームにおいて，プレイヤは攻撃側と防御側のいずれかをプレイする．（寺田 [5] の提案では，攻撃側と防御側を入れ替えて 2 度対戦して勝敗を決める形をとっている．）対戦型 2048 にはいくつかの変種がある[*1]が，以下では本論文で用いたルールを示す．

ゲームの各局面は，4 × 4 の大きさの盤面に，2 のべき乗の値をもつタイルが置かれたものとなっている．初期局面において，攻撃側プレイヤは以下のルールに従って 2 つのタイルを置く．続けて，防御側プレイヤと攻撃側プレイヤが交互に，以下のルールに従って手を選択していく．防御側プレイヤが選択できる手がなくなったときゲームは終了する．攻撃側プレイヤの目標はゲーム終了時の得点を小さくすることであり，防御側プレイヤの目標はゲーム終了時の得点を大きくすることである．

攻撃側プレイヤ

本研究における攻撃側プレイヤは，盤面上の空きマスを一つ選び，そこに 2 または 4 のタイルを置く．

防御側プレイヤ

防御側プレイヤは，タイルを動かす方向を上下左右の中から一つ選択する．方向を選択すると，タイルは以下の規則で移動・併合する．いずれのタイルも移動・併合しない方向は選択できない．

- タイルは，選択した方向にできるだけ移動する（「Threes!」のように，1 マスだけ移動するわけではない）．たとえば，右を選択した際に，2,_,4,_ という列は _,_,2,4 へと変化する．
- 選択した方向に，同じ値をもつタイルが連続している場合，併合が起こる．併合により，タイルの値の和をもつ一つのタイルができ，その和が得点に加算される．たとえば，右を選択した際に，2,2,8,8 という列は _,_,4,16 に変化し，4 + 16 = 20 点が得点に加算される．
- 併合によってできたタイルは，その一回の移動の中では再度併合されることはない．たとえば，右を選択した際に，_,4,2,2 という列は _,_,4,4 に変化する．
- 選択した方向に対して同じ値を持ったタイルが連続して 3 枚以上置かれている場合，より移動先に近い 2 枚が併合される．たとえば，右を選択した際に，_,2,2,2 という列は _,_,2,4 に変化する．

3. プレイヤの作成

本研究では，先行研究 [4] で学習を進めたプレイヤのうち，最も強かった攻撃側プレイヤの評価関数と防御側プレイヤの評価関数を用いて，新たに $\alpha\beta$ 探索を組み合わせる．ここでは先行研究 [4] で使用した評価関数と学習方法について説明する．

3.1 評価関数

先行研究および本研究で使用するニューラルネットワークの構成を図 1 に示す．ニューラルネットワークは，畳み込み層 2 層と全結合層 3 層で構成されており，現在の局面を入力として，その局面の評価値を 1 つの値として出力する．

ニューラルネットワークの入力は，盤面を 4 × 4 × 16 の 3 次元の 2 値配列に変換したものである．図 2 に盤面を入力に変換する方法を示す．1 番目の 4 × 4 は空きマスの位置を値 1 で示し，2 番目は 2 のタイルの位置を値 1 で示し，以下同様にして，16 番目の 4 × 4 は 32768 のタイルの位置を値 1 で示す．

3.2 攻撃側プレイヤ

攻撃側プレイヤは，与えられた局面から任意の空きマスに 2 または 4 のタイルを 1 つ配置した盤面を作成し（最大 $(16 - 1) \times 2 = 30$ 通り），それらの盤面をそれぞれ図 1 の

[*1] 本論文で採用したルールは，攻撃側プレイヤにとって最も有利なものであり，このルールではゲームは確率的ではない．ゲームが確率的であるような変種の例として，タイルの数がランダムに決まった後に攻撃側が置く場所を選択するもの（横山と松崎 [6] で採用したルール）がある．対戦型 2048 の最初の提案では，新規タイルの値は 2 固定であるというルールが採用されていた．

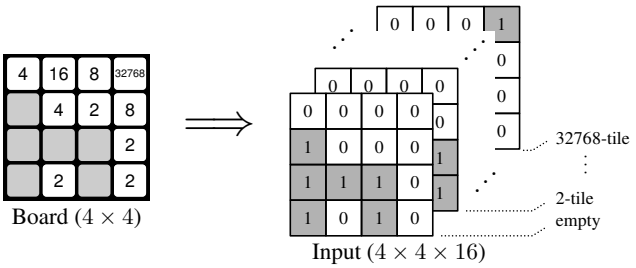

図 2　入力データの成形方法 [1]

表 1　各ステップ終了時の平均スコア

フェーズ	攻撃	防御	攻撃	防御	攻撃	防御
総学習時間	12	36	48	72	84	108
平均スコア	8300	5530	3120	3960	3400	3860

ニューラルネットワークにより評価する．得られた評価値が最も小さくなる位置と値を手として選択する．

3.3　防御側プレイヤ

防御側プレイヤは，以下で述べる学習方法の違いを除いて，既存研究 [1] で作成した通常の 2048 プレイヤと同じである．

防御側プレイヤは，与えられた局面に対して，4 方向に移動・併合を行った後の盤面を生成して図 1 のニューラルネットワークにより評価値を計算する[*2]．選択できない方向を除いて，評価値の最も大きくなる方向を手として選択する．

3.4　学習方法

先行研究では，TD 誤差学習の手法を用いて，ニューラルネットワークの重みを調整した．

学習手順として，まず通常の 2048 におけるプレイヤ（防御側プレイヤ）を 24 時間の学習により作った．その後，攻撃側プレイヤと防御側プレイヤを交互に学習するようにした．攻撃側の学習を 12 時間/ステップ，防御側の学習を 24 時間/ステップとして，（最初の防御側プレイヤの学習を含めて）合計 7 ステップで学習を進めた．つまり，攻撃側は 84 時間，防御側は 108 時間学習したプレイヤが最も学習時間の長いプレイヤであり，最も優れた性能のプレイヤであった．以降では，上記の学習時間の後のプレイヤを用いる．

先行研究 [4] では学習の条件として複数の方法を試したが、その中で最も良い結果が得られたのは，対称性を考慮し，ランダムな手を選ぶ確率を 0 %とする条件で学習したものである．（各ステップ終了時の平均スコアを表 1 に示す．）

[*2] 攻撃側プレイヤと防御側プレイヤで，ネットワークに与える局面の種類が異なることに注意．防御側プレイヤはタイルを移動した直後，攻撃側プレイヤは新しいタイルを出現させた直後の盤面を評価している．

4.　$\alpha\beta$ 探索の追加

1 人用の 2048 プレイヤにおいて，既存研究 [1] ではニューラルネットワークを用いた強化学習に，Expectimax 探索を組み合わせることで性能が向上するという結果が得られている．対戦型 2048 についても，岡と松崎 [3] により作成された N タプルネットワークを用いるプレイヤでは，強化学習と組み合わせる minimax 探索の深さが深いほど性能が向上している．

これらのことから，対戦型 2048 におけるニューラルネットワークプレイヤについても，強化学習に探索を組み合わせることで性能の向上が期待できる．しかし，先行研究 [4] では強化学習のみの実装で探索を組み合わせていなかった．そこで本研究では，先行研究 [4] で得られたプレイヤに $\alpha\beta$ 探索を組み合わせ，性能の向上が見られるかを確認する．

本研究における $\alpha\beta$ 探索プレイヤは，前節で示した攻撃側評価関数または防御側評価関数を用いて $\alpha\beta$ 枝刈りありの Minimax 探索を行う．計算時間の制約から，探索の深さ d は最大 $d = 7$ とした．具体的には，攻撃側プレイヤでは，深さ d が奇数のときには攻撃側評価関数を用い，d が負のときには防御側評価関数を用いる．一方，防御側プレイヤでは，深さ d が奇数のときには防御側評価関数を用い，d が負のときには攻撃側評価関数を用いる．以降では，探索深さ d の攻撃側プレイヤを atk_d と表記し，探索深さ d の防御側プレイヤを def_d と表記する．

なお，$\alpha\beta$ 探索の実装にあたっては，探索速度の向上のため，同一親ノードの子ノードを評価値でソートするムーブオーダリングを実装した．最も時間のかかった $d = 7$ の場合，atk_7 で平均約 8.2 秒，def_7 で平均約 3.4 秒であった．

5.　対戦実験

本研究では，以下に示す 2 つの実験を行った．実験に用いた環境は，実験 1，実験 2 のいずれも表 3 に示すとおりである．

5.1　実験 1：攻撃側・防御側それぞれの評価関数を用いた探索の性能評価

先行研究 [4] では，フェーズごとに攻撃側と防御側を学習している．そこで評価関数の一貫性を確保するため，攻撃側プレイヤと防御側プレイヤの両者について，探索深さを $d = 1, 3, 5, 7$ と奇数に限定した場合（すなわち，攻撃側プレイヤは攻撃側評価関数のみを用い，防御側プレイヤは防御側評価関数のみを用いる）の総当たりの実験を行う．各攻撃側プレイヤと防御側プレイヤの組合せについて，攻撃側が最初の 20 手をランダムにプレイした局面からスタートして 100 回ずつ対戦を行った（合計 1600 試合）．対戦実験の全体は 2 週間で終了した．

実験 1 の結果として，それぞれの組み合わせについて平

表 2　各対戦の対戦結果（平均スコア，最大スコア，達成した最大タイル）

	def_1			def_3			def_5			def_7		
	平均	最大	タイル	平均	最大	タイル	平均	最大	タイル	平均	最大	タイル
atk_1	4051	11328	1024	7031	14388	1024	10588	25324	2048	11244	25404	2048
atk_3	1611	3732	256	3367	9740	1024	5200	13352	1024	7381	13772	1024
atk_5	1270	2868	256	1767	3720	256	3167	6660	512	4673	11236	1024
atk_7	1157	2380	256	1471	2796	256	2005	4896	512	3020	6008	512

表 3　学習に用いた計算機環境

CPU	Intel Core i3-8100 BOX
	(4 コア, 4 スレッド, 3.60GHz)
メモリ	16 GB
GPU	ZOTAC GeForce GTX 1080 Ti Mini
	ZT-P10810G-10P (GPU メモリ 11 GB)
OS	Linux version 4.15.0-139-generic
Python	3.6.9
TensorFlow	1.14.0

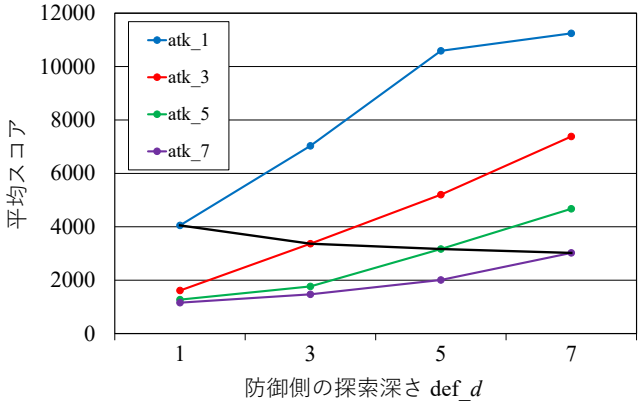

図 3　各攻撃側プレイヤの平均スコア

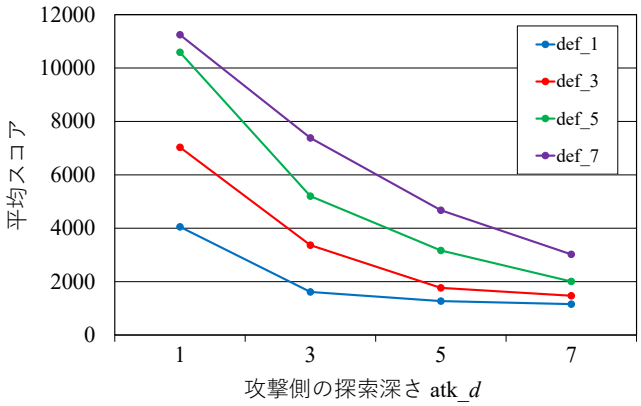

図 4　各防御側プレイヤの平均スコア

均スコア，最大スコア，最大タイルの値を表 2 に示す．平均スコアについて，防御側プレイヤを横軸にプロットしたグラフを図 3 に，攻撃側プレイヤを横軸にプロットしたグラフを図 4 に示す．図 3 には追加で黒色の線が引かれている．これは，攻撃側プレイヤと防御側プレイヤで同じ探索深さの場合の結果を線で結んだものである．

5.2　実験 2：異なる評価関数により探索を行った場合の性能評価

次に，攻撃側プレイヤと防御側プレイヤの両者について，探索深さに $d = 2, 4, 6$ を追加して総当たりの対戦を行う．探索深さが偶数のとき，攻撃側プレイヤは防御側の評価関数を用いて計算を行い，防御側プレイヤは攻撃側の評価関数を用いる．各攻撃側プレイヤと防御側プレイヤの組合せについて，攻撃側が最初の 20 手をランダムにプレイした局面からスタートして 100 回ずつ対戦を行った（合計 3300 試合）．対戦実験の全体は 2 週間で終了した．

対戦実験 2 の結果として，図 3，図 4 に深さ $d = 2, 4, 6$ の $\alpha\beta$ 探索を組み合わせたプレイヤを追加したものを図 5，図 6 に示す．

6.　考察

6.1　最も強い攻撃側プレイヤ

本研究で得られた攻撃側プレイヤのうち，最も強い探索深さ $d = 7$ のプレイヤは，探索を行わないプレイヤと比べて，何れの防御側プレイヤに対しても，平均スコアを約 4 分の 1 に抑えることができている．このことから，攻撃側プレイヤについて，ニューラルネットワークプレイヤに探索を組み合わせることで，性能が大きく向上したことが分かる．

また，深さ 7 の探索を行う攻撃側プレイヤは防御側プレイヤに対して最大でも 512 のタイルまでしか作らせなかった．このことから，攻撃側が 2 と 4 のタイルを自由に置けるルールの下では，2048 のタイルを作ることは不可能であることが予想される．（これは，元の 2048 のゲームにおいて，非常に運が悪い場合にどのようにプレイしても 2048 のタイルを作ることが不可能であることを意味する．）

6.2　最も強い防御側プレイヤ

本研究で得られた防御側プレイヤのうち，最も強い探索深さ $d = 7$ のプレイヤは，探索を行わないプレイヤと比べて，2〜4 倍の平均スコアを出している．このことから，防御側プレイヤについても，ニューラルネットワークプレイヤに探索を組み合わせることで，性能が大きく向上したことが分かる．

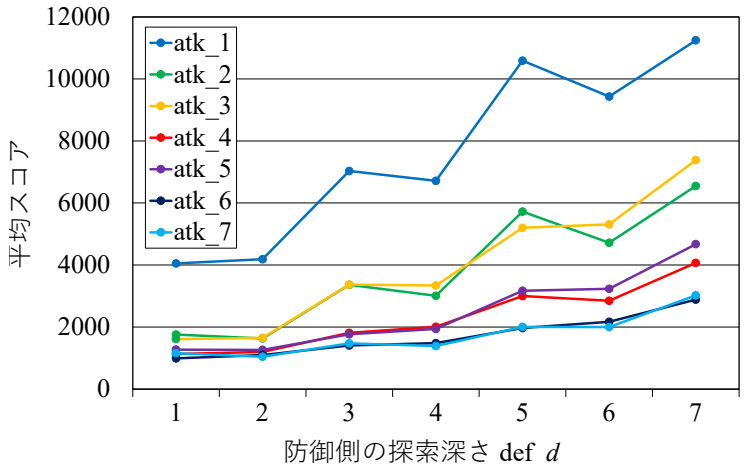

図 5　各攻撃側プレイヤの平均スコア

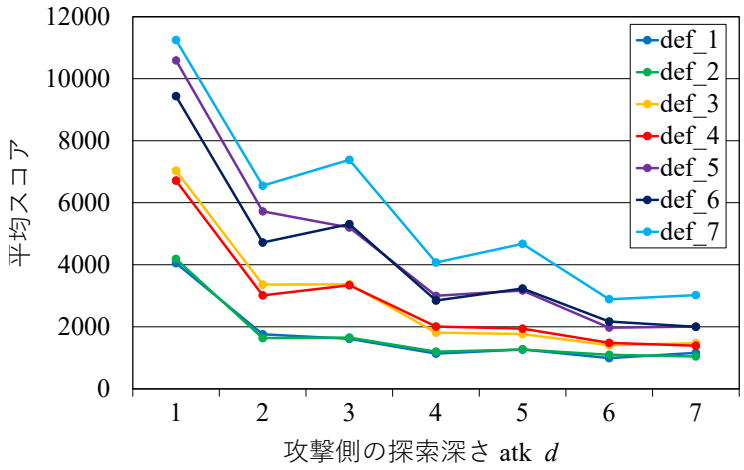

図 6　各防御側プレイヤの平均スコア

6.3　探索深さと強さの関係

図 3 と図 4 より，攻撃側プレイヤ，防御側プレイヤの両方で探索を深くすると強くなるという結果が得られた．攻撃側プレイヤでは，探索を深くしていくと徐々にスコアの減少幅が小さくなっているが，これはゲームの特性上，非常に小さな得点で終えさせることがより困難であるためだと考えられる．

また，図 3 に示した同じ深さの探索を行うプレイヤ同士の対戦において，探索を深くしていくと徐々にスコアが下がるという結果が得られている．探索深さ $d = 1$ から $d = 7$ の結果からの推測では，双方が探索深さを増やしていった際に，平均スコア 3000 点に近いところに収束するのではないかと考える．

6.4　用いる評価関数の影響

図 5 より，防御側プレイヤの探索深さが奇数のときと比べて偶数のときに平均スコアが下がっていることが分かる．防御側プレイヤは平均スコアが高いほど性能が良いので，防御側の評価関数を用いた方が性能が高かったと言える．また，図 6 より，攻撃側プレイヤの探索深さが奇数のときと比べて偶数のときに平均スコアが下がっていることが分かる．攻撃側プレイヤは平均スコアが低いほど性能が良いので，防御側の評価関数を用いた方が性能が高かったと言える．これらの結果から，攻撃側の評価関数と防御側の評価関数を比較すると，防御側の評価関数の方が優れていることが分かる．つまり，タイルを動かした後の盤面を評価関数に入力した方が性能が良いということが示唆される．これは，Szubert と Jaśkowski による 2048 に対する考察 [2] と一致している．

7.　まとめ

本研究では，先行研究 [4] で学習を行った攻撃側と防御側の評価関数をもとに，深さ $d = 1, 2, 3, 4, 5, 6, 7$ の $\alpha\beta$ 探索を行うプレイヤを作成した．総当たりによる実験の結果，探索を深くするごとにプレイヤが強くなること，同じ深さの探索を行うプレイヤ同士の対戦は探索を深くしていくと徐々にスコアが下がること，タイルを動かした後の盤面を評価関数入力した方が性能が良いことなどが結果として得られた．

今後の課題として，探索の深さを更に増やしたときの結

果の確認や，他の条件で作成したプレイヤとの対戦がある．他のゲームにおける探索と比べると，今回の実験で用いた探索の深さは浅い．そのため，今後はより深い探索を行った場合に結果がどう推移するのかを確認したい．ただし，現在より探索を深くするには，ニューラルネットワークの計算の呼び出し方法を工夫するなどして，探索にかかる時間を削減する必要がある．また，N タプルネットワークによる評価関数を用いたプレイヤとの対戦による比較は重要な話題である．優先的に取り組んでいきたい．

参考文献

[1] K. Matsuzaki: Developing Value Networks for Game 2048 with Reinforcement Learning, *Journal of Information Processing*, Vol. 29, pp. 336–346, 2021.

[2] M. Szubert and W. Jaśkowski: Temporal Difference Learning of N-Tuple Networks for the Game 2048, *2014 IEEE Conference on Computational Intelligence and Games*, pp. 1–8, 2014.

[3] 岡　和人，松崎公紀：システム的選択による N-tuple networks の “対戦型 2048” への適用，情報処理学会第 58 回プログラミング・シンポジウム，pp. 193–202, 2017.

[4] 小田駿斗，松崎公紀：攻撃側が置くタイルの数を選択できる対戦型 2048 に対するニューラルネットワークプレイヤの学習，情報処理学会第 63 回プログラミング・シンポジウム, 2022.

[5] 寺田　実：対戦型 2048，情報処理学会夏のプログラミング・シンポジウム [2015] 報告集，pp. 19–22, 2016.

[6] 横山智洋，松崎公紀：ニューラルネットワークと強化学習による対戦型 2048 プレイヤの作成，情報処理学会第 62 回プログラミング・シンポジウム, 2021.