

論文題目

未来局面の提示による強化学習 AI の 判断の可視化

Visualization of Reinforcement Learning AI Decision
Making through Future State Proposals

指導教授

萩原 将文 教授

慶應義塾大学 理工学部 情報工学科

令和 令和五年度 年度

学籍番号 62019277

村上 花恋

目次

あらまし	1
第1章 はじめに	2
第2章 関連研究	4
2.1 強化学習 [6]	4
2.1.1 ボードゲームへの応用	5
2.1.2 AlphaZero[1]	6
2.1.3 ボードゲーム AI の問題点	9
2.2 XAI	9
2.2.1 概要	9
2.2.2 ボードゲームにおける XAI	12
2.2.3 contrastive explanation	12
2.2.4 importance	13
2.2.5 ボードゲーム学習支援	13
第3章 提案手法	15
3.1 実装	15
3.2 importance	18
第4章 評価実験	21
4.1 connect4[30]	21
4.2 alphazero_baseline[33]	22
4.3 データ実験	23
4.3.1 データセット	23

4.3.2	比較手法	24
4.3.3	評価指標	24
4.3.4	実験結果	25
4.4	システム実験	25
4.4.1	実験手順	25
4.4.2	評価指標	26
4.4.3	実験結果	27
第 5 章 結論		29
謝辞		31
参考文献		32
付録		36
付録 A データ実験の詳細		36
A.1	使用したモデルの詳細	36
A.2	対戦結果の詳細	36
A.3	評価指標の詳細	37
A.3.1	fcount	38
A.3.2	fdcount	38
A.4	データ実験における提案指標の計測	39
A.5	データ実験に使用したモデルの詳細	40
A.6	グレーボックス的手法	40
付録 B 各種アルゴリズムの詳細		42
B.1	比較手法のアルゴリズム	42
B.2	提案手法におけるニューロ補間	42
B.3	システム実験における提案手法の変更	42

付録 C	alphazero_baseline	46
C.1	ニューラルネットワーク	46
C.2	alphazero_baseline のパラメータ更新	47
付録 D	システム実験の詳細	50
D.1	パラメータ	50
D.2	異なる期間	50

あらまし

2024 年現在、チェス、囲碁、将棋などの多くの二人用ボードゲームにおいて AI は人間を遥かに凌駕するようになった。しかし、AlphaZero[1] の登場から 5 年が経過した今もなお、その十分な説明手法は登場していない。本論文では AI の予測する未来図を複数提示することで AI の判断根拠の可視化を試みた。題材として connect4 を選択し、AI 同士の対戦データを用いたデータ実験と被験者からのデータを用いたシステム実験の二種類を行った。データ実験では提案手法が比較手法よりも高い予測精度を示し、システム実験では提案手法は ? において比較手法よりも高い評価を得た。

第1章

はじめに

近年の AI の発展は目覚ましく、画像分類や異常検知などの単純なルールで記述する事が困難なタスクや、更には長らく人間に固有の技術であると考えられてきた画像や文章の生成の分野においてさえ、高い性能を発揮するまでに AI 技術は成長した。特に直近 3 年の stable diffusion[2], Instruct GPT[3] の登場は専門家間に留まらず一般社会に大きな影響を与えた。この「優れた AI に対して人間はどう接するべきか」という命題を考える際には既に人間を大きく凌駕した AI が存在する領域において手法の構築や実験を行うのが適当である。そのため、本論文では囲碁, 将棋, チェスなどの主要なボードゲームにおいて人間を大きく凌駕したパフォーマンスを誇る AlphaZero[1] を題材に、説明性付与を試みる。

優れた AI が社会で広く実用化され、受容されるためには AI の判断や生成物(以下単純に「出力」という表現を使用する)が生み出される過程の透明性、信頼性の担保が必要不可欠である。説明性が必要とされるのは医療や金融などの慎重な判断が求められるべき分野のみではない。上述のように画像生成・文章生成 AI が上述の様に一度オープンソース化された 2024 年現在においても新技術の国際的な協調の観点から G7 広島サミットにおいて AI の透明性を確保する重要性が指摘されており [4]、日本国内においても「信頼できる AI」を実現する必要性が認識されている [5]。また、将来的に優れた AI が実用化されていく過程で「AI に学ぶ」という探求心や学習意欲からも説明性へのニーズが広く湧き起こる事が予測される。現に囲碁, 将棋, チェスなどの主要なボードゲームをオンラインでプレイできるサービスにはゲーム終了後の振り返り (感想戦) にお

いて AI の判断や AI による想定図を閲覧できるサービスが数多く提供されている。(ここに参考文献を入れる) 研究においてもボードゲームの学習支援という形で「AI に学ぶ」試みが存在する。しかし、後述するようにそれらの研究は各ゲームのドメイン知識等を使用したものが多い。

そこで本論文ではなるべくドメイン知識を用いず、AI の判断を可視化する手法を考案し、その有効性を検証した。手法の詳細は第 3 章に記載しており、実験の詳細は第 4 章に記載している。

第2章

関連研究

この章ではまず、既存のボードゲーム AI について AlphaZero を中心に強化学習的枠組みからその理論を説明する。次に Alpha Zero の問題点とそれを補完する既存手法とその課題について述べる。

2.1 強化学習 [6]

強化学習はタスクを選択をする主体と環境のやり取りとして定式化し、その相互作用から学習する形でタスクに取り組む分野である。状態 s と行動 a が次の状態 $s' (= T(s, a))$ (T は遷移関数) と環境から与えられる報酬 r が決定されると仮定する。その仮定の下、環境から与えられる報酬の合計 G (以下収益と記載) を最大化する。報酬を大きくするためには状態 s に応じて適切な (より大きな報酬をもらえる可能性が高い) 行動を選択する必要がある。ある状態 s である行動 a をとった場合の収益に対して見積もりをとり、見込まれる値が最も大きい行動を選択することでより大きな収益を獲得できると期待できる。このようなある状態である行動を取った場合の収益の見積もりを $Q(s, a)$ とした場合、

$$a_m = \operatorname{argmax}_a Q(s, a) \quad (2.1)$$

となる a_m を選択することによって収益の最大化が期待される。また、ある状態から獲得できる収益の合計の予想値 $V(s)$ (以下状態価値関数と表記) は、最適な行動 a_m を取った場合の値として推定される。

$$V(s) = Q(s, a_m) (a_m = \operatorname{argmax}_a Q(s, a)) \quad (2.2)$$

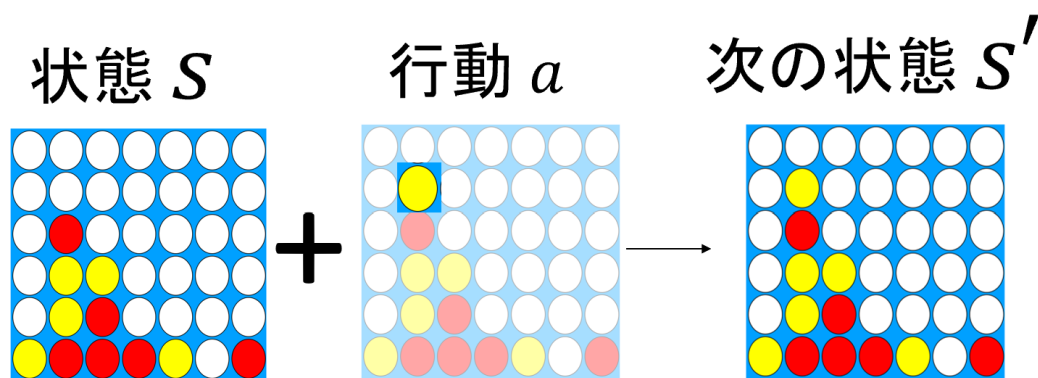


図 2.1: ボードゲームにおける強化学習モデル

強化学習手法によってタスクの最適化を図る際にはこの $V(s), Q(s, a)$ を正しく推定することが直接的な目標となる。 $V(s), Q(s, a)$ は主体が実際に環境とやり取りを行う（タスクを実行していく）中で改善されていき、Temporary Difference 法や Monte Carlo 法等が基本的な $V(s), Q(s, a)$ の更新則である。また、DQN[7] や Rainbow[8] 等はニューラルネットワークを使用して $V(s), Q(s, a)$ を推定することでより高い性能を発揮している。

2.1.1 ボードゲームへの応用

ボードゲームでは通例、状態 s は盤面の状況、行動はプレイヤーの選択、報酬はゲームの最後に勝敗として与えられる。状態 s (ゲームの状況) と行動 a (プレイヤーの選択) によって盤面は次の状態 s' に遷移し、次の行動 a' (他のプレイヤーによる選択) を受け付ける、というサイクルにゲームの進行を定式化して表現することができる。また、上述した強化学習における $V(s)$ の推定は「ある盤面はプレイヤーにとって勝利に近いのか」を表現していると解釈される。AlphaGo[9] や StockFish[10]、DeepLearningShogi[11] では状態 s は最新 N ステップの盤面や手数などのゲームのプレイにおいて重要な情報である。状態は行列等の形式に抽象化され、行動は次にプレイヤーが打つ箇所の座標となる。

本論文で使した alphazero_baseline における入力は最新の盤面の状態を空白を 0, 先番 (赤) の石の位置を 1, 後番 (黄) の位置を -1 として抽象化した 6×7 の

行列となる。また、後述する connect4 のルール上の制約により次にプレイヤーが打つ箇所 (行動) は列の数と同数の 7 つに限定される。

2.1.2 AlphaZero[1]

AlphaZero は 2016 年に登場し、元世界チャンピオンであるイ・セドルに対して四勝一敗の成績を収めた AlphaGo の汎用版である。AlphaZero は先述の $V(s)$, $Q(s, a)$ を推定する際にニューラルネットワークと PV-モンテカルロ木探索システムを使用する。

2.1.2.0.1 ニューラルネットワーク AlphaZero 内のニューラルネットワークに対する入力は最新 8 ステップの盤面 ($\{s_{-7}, \dots, s_0\}$, s_{-i} は i ステップ前の盤面, s_0 は現在の盤面) であり、出力は方策 $P(\{s_{-7}, \dots, s_0\})$ と局面評価 $V(\{s_{-7}, \dots, s_0\})$ の二種類である。ネットワークは 1 つの畳み込み層と 20 の残差結合ネットワークで構成されている。方策は「現在の状況 s_0 から次にどこを選択すべきか」を表現しており、次に選択すべき座標を確率分布の形式で表現する。alphazero_baseline における選択肢は列の数と等しい 7 であるため、 1×7 の行列となる。方策内の値が大きさが AI によるその着手の評価と解釈され、成分が大きい座標を次に選択することが推奨される。例えば方策が $\{0, 0.1, 0.2, 0, 0, 0.7, 0.8\}$ であるとき、方策中の最も大きい成分は 7 番目の 0.8 であるため、プレイヤーは次に 7 列目を選択する事が推奨される。また、局面評価 $V(\{s_{-7}, \dots, s_0\})$ は「(過去 8 ステップ分の状態を含めた) 現在の状況 s_0 は勝利に近いのか」を表現しており、値が上限に近ければ近い程、現在の状況 s_0 が次の着手を選択するプレイヤーにとっての勝利に近いことを表している。

訓練とかどうするん？

2.1.2.0.2 PV-モンテカルロ木探索 PV-モンテカルロ木探索ではニューラルネットワークから得た方策 $P(s)$ (以下 $s = \{s_{-N+1}, \dots, s_0\}$ と表記する) と局面評価 $V(s)$ をシミュレーションによって改善する。PV-モンテカルロ木探索ではシミュレーションによって s をノード、各行動 a を枝とした決定木を構築する。最



1. ノードを探索したことがない場合ニューラルネットワークから出力された方策 $P(s)$ と局面評価 $V(s)$ を返却する
2. ノードを探索したことがある場合以下の puct スコア $U(s, a)$ に従い、子ノード s_c を選び、 s_c に対して再帰的に探索を行いその結果である $P(s_c), V(s_c)$ (s_c は再帰処理の結果たどり着く決定木の端のノード) を返却する。 ($N(s), N(s, a)$ は

それぞれ $s, (s, a)$ に対して探索を行った回数)

$$U(s, a) = C(s)P(s, a)\frac{\sqrt{N(s)}}{1 + N(s, a)} \quad (2.3)$$

$$C(s) = \log \frac{1 + N(s) + C_{\text{base}}}{C_{\text{base}}} + C_{\text{init}}(C_{\text{base}}, C_{\text{init}} \text{ はハイパーパラメータ}) \quad (2.4)$$

このようなプロセスによって構築された決定木を用いてモデルは対戦を行う。

2.1.3 ボードゲーム AI の問題点

StockFish[10] や DeepLearningShogi[11] などの従来のボードゲーム AI はそのゲーム固有の知識 (ドメイン知識) に基づくものが多く、「ある条件を満たすときにある選択をする」と言ったようにその挙動をルールとして表すことが可能である。一方で AlphaZero のニューラルネットワークと木探索を組み合わせた手法では方策と局面評価の根拠を得ることができない、つまり AlphaZero の問題点として説明性の欠如が挙げられるのである。説明性の欠如は AI の判断に対する責任の不在を意味し、優れた性能を持つシステムのより、ハイレベル、ハイリスクなタスクへの実用化に対する障害となる。

2.2 XAI

2.2.1 概要

XAI とは explainable AI (説明可能 AI) の略語であり、AI を人間に対して説明可能なものにする、もしくは説明可能な AI を構築する領域である。本論文は AI の判断根拠として先読みを示す意味で XAI の分野に属する研究であると言える。ここでの「説明可能性」の語は「人間に理解できる形での説明を与える能力」[12] と定義され、「いつ、どのような、どのように」説明を与えるかによってさらに細かく分類される。「いつ」、つまりどの時点で説明を与えるか、に関しては既存のネットワークに対して新たに説明を加える「事後的」説明と初めから動作の根拠を示せるようにネットワークやシステムを構築する「事前的」説明に分類できる [12]。「どのような」、つまり説明の内容については「大域的」説明と「局所的」説明の二つに分類される。「大域的」説明は行動 a を選択する主体 (モデル) の全体的な方針について述べるものである一方で、「局所的」説明は主体 (モデル) の個々の判断について説明する [13]。が存在する。また、「どのように」つまり説明を表現する形態としては saliency map[14], Grad-CAM[15] といった視覚的な可視化や、あとで教師データと予測結果との関係の数値的な

Algorithm 1 PV-MCTS in AlphaZero (Part 1: Exploration)



```

1:  $t$ : 決定木
2:  $T$ : 遷移関数
3:  $N(s, a)$ :  $(s, a)$  の組み合わせを探索した回数
4:  $Q(s, a)$ : 行動価値関数 ( $\text{Explore}(s)$  の平均)
5:  $W(s, a)$ : 行動価値の総和 ( $W(s, a) = Q(s, a)N(s, a)$ )
6:  $P(s, a)(= P(s_n), s_n = T(s, a))$ :
7: ニューラルネットワークから出力された方策
8:  $V(s, a)(= V(s_n), s_n = T(s, a))$ :
9: ニューラルネットワークから出力された局面評価
10: function EXPLORE( $s_{\text{start}}$ )
11:   Set  $s_{\text{now}} = s_{\text{start}}$  and  $a_{\text{now}} = a_m$ 
12:   for each simulation do
13:      $\zeta \leftarrow \text{emptylist}$ 
14:      $s_{\text{current}} \leftarrow s_{\text{start}}$ 
15:     while  $s_{\text{current}}$  がゲームの終了状態でない場合 do
16:        $a_t \leftarrow \text{TreePolicy}(s)$ 
17:       ( $T$  は遷移関数)
18:        $(s_{\text{current}}, a_t)$  を  $\zeta$  の末尾に追加
19:        $s_{\text{next}} \leftarrow T(s_{\text{current}}, a_t)$ 
20:        $s_{\text{current}} \leftarrow s_{\text{next}}$ 
21:     end while
22:      $G \leftarrow V(s_e)$ 
23:     ( $s_e$  は  $s_{\text{start}}$  から探索してたどり着いたノード  $s_e$ )
24:     BACKPROPAGATE( $\zeta, G$ )
25:   end for
26: end function

```

Algorithm 2 PV-MCTS in AlphaZero (Part 2: Backpropagation)

```

1: function TREEPOLICY( $s$ )
2:   if  $s$  が探索されていない子ノードを持つとき then
3:      $s_c \leftarrow T(s, a)$  ( $s_c$  は未探索のノード)
4:     INITNODE( $s_c$ )
5:       $a$ 
6:   else
7:     以下の PUCT スコア  $U(s, a)$  を計算
8:      $U(s, a) = C(s)P(s, a) \frac{\sqrt{N(s)}}{1+N(s, a)}$ 
9:      $C(s) = \log \frac{1+N(s)+C_{\text{base}}}{C_{\text{base}}} + C_{\text{init}}$ 
10:    ( $C_{\text{base}}, C_{\text{init}}$  はハイパーパラメータ)
11:    ( $N(s) = \Gamma N(s, a)$ )
12:    以下のように  $a$  を求める
13:     $a = \operatorname{argmax}_a (Q(s, a) + U(s, a))$ 
14:      $a$ 
15:   end if
16: end function
17: function BACKPROPAGATE( $\zeta, G$ )
18:   for each node-action pair  $(s, a)$  in  $\zeta$  do
19:      $N(s, a) \leftarrow 0$ 
20:      $W(s, a) \leftarrow 0$ 
21:      $Q(s, a) \leftarrow 0$ 
22:   end for
23: end function
24: function INITNODE( $s$ )
25:   for each action  $a$  from  $s$  do
26:      $N(s, a) \leftarrow N(s, a) + 1$ 
27:      $W(s, a) \leftarrow W(s, a) + G$ 
28:      $Q(s, a) \leftarrow \frac{W(s, a)}{N(s, a)}$ 
29:   end for
30: end function

```

定量化 [16] が存在する。本論文において構築するシステムは「事後的」「局所的」「視覚的」説明を提供する。

また、本論文は XAI 分野の中でも特に強化学習方面に対して説明を加える領域を XRL(explainable Reinforcement learning) と呼ぶ。XRL の試みは様々な強化学習システムを対象とし、

2.2.2 ボードゲームにおける XAI

McGrath ら [17] はチェスにおける人間の知識が AlphaZero にどれだけ反映されているかを訓練段階やネットワークの深さなどの多様な指標で調査している。Lee ら [18] は AI の着手に対してゲームの固有の知識 (ドメイン知識) を用いてモデルの挙動に対する解説文の自動生成の試み等がそんざいする。しかし、このようなドメイン知識は必ずしも AI の挙動と相関が無いことも指摘されている [17]。AI による画像分類タスクの可視化手法として用いられていた saliency map[14] や Grad_CAM[15] を強化学習に用いる例も存在する [19][13][20]。しかしそれらのニューラルネットワークの活性を根拠とした指標は木探索部分との繋がりが弱く、最終的に決定木を用いて意思決定を行うシステムの動作根拠を直接的に説明できない。また、これらの画像分類用の手法は

- 本来ゲームに存在する時系列の要素を説明に含められない
- 時系列を無視してゲーム画面や盤面の一部を変更する必要がある

という問題点が存在する。

2.2.3 contrastive explanation

上述の問題点を解決するために、本論文では AlphaZero が構築する決定木を用いた contrastive explanation (対象説明、比較説明) を提供する。contrastive explanation は事象を説明する方法論の一つであり、ある事象 a が起こった際にその理由を直接説明する代わりに「他の事象 \bar{a} が起こらなかった理由」を説

明することで間接的にある事象 a の原因を説明するアイデアである [21]。Jacovi らの [21] では自然言語処理の分類タスクにおいて本来の入力データと編集されたデータの出力を比較することで、入力の中のどの部分がモデルの判断に寄与しているかを示している。Mishra らの [22] では医療タスクにおけるニューラルネットワークの判断を決定木に近似した上で AI の判断 a から派生する予想と別の判断 b から派生する予想を同時に提示する形で AI による判断の妥当性を示している。を実装している。ロボットなし? また、Gajcin らの [23] では異なるモデルの挙動の違いをユーザーに示す目的で contrastive explanation が用いられている。

2.2.4 importance

ゲームやタスクにおいて勝負の分かれ目となりうる場面や、ミスをしやすい場面、危険な事故が起きやすい場面を特定することは非常に有用である。また、このような AI モデルが他の場面よりも重要度が高いと判断する状況を収集することは、モデルの挙動に対する効率的な調査を可能にする。Torraró ら [24] や Amir ら [25] は一つ先の行動価値関数 $Q(s, a)$ が次の選択によって大きく左右されるような状態を重要度の高い局面として定義している。しかし、このような定義は imp が悪い方に大きく影響されるリスクや下のように状態に対称性があるような場合に適切な判断が阻害されるリスクがある。Guo ら [26] は重要度 I を収益 G が確定するまでの一連の流れ (エピソード) やエピソード内の時点 t の重要度を収益 G との関連度の大きさから定めている。Guo らの定義は状態の符号化や回帰などのデータの加工段階が多く、指標自体の説明性に疑問が残る。AI モデルに対して説明を加える際にはその手法自体もなるべく簡明である方が望ましいと考えられる。

2.2.5 ボードゲーム学習支援

本論文は高度な AI の動作を人間に理解させることを目標としており、学習支援の側面を含んでいると言える。既存の AI を用いたボードゲーム学習支援シス

テムとしては先述の Lee ら [27] やオンラインサービスである DecodeChess[18] などによる解説文自動生成や、Richard[28] らや Richard[29] らの人間側の悪手を自動的に検知しその理由を個別に指摘するモデルが提案されている。しかし「ボードゲームに対する XAI」の段での内容と同様にその多くがゲームのドメイン知識に依存しており、指導の内容も人間の知識に依存したものになってしまうという欠点がある。

第3章

提案手法

関連研究の章ではニューラルネットワークや決定木が内包する説明性の欠如という問題点と説明を加える既存手法の持つ課題について述べた。本論文ではそれらを踏まえた

- 本来ゲームに存在する時系列の要素を含む
- 評価基準が勝敗に直結する
- 人間のドメイン知識に依存しない

説明手法を提案する。本手法はある状態 s と行動 a の組に対して AI が予想する未来図 $o(s)$ とそこに至るまでの道筋を取り出すことで AI の判断の意図を可視化することを目標とする。

3.1 実装

本手法は AlphaZero システムのニューラルネットワークと木探索部分のうち、主に木探索の部分を対象に適用される。本手法では決定木の判断を説明する際に最も有用な部分を決定木から抽出することを主な目的としており、アルゴリズムは決定木構造を持つ多くのシステムに応用可能である。アルゴリズムの流れは以下の通りである。図 3.1、図 3.2 はステップごとのアルゴリズムのイメージ図である。1:説明を付与する状態と行動の組 (s_{start}, a) を選択する。このとき注目している状態 s_{now}, a_{now} はそれぞれ

$$s_{now} = s_{start}, a_{now} = a \quad (3.1)$$

① (s_{start}, a) を決定

② 訪問回数上位のノードを収集

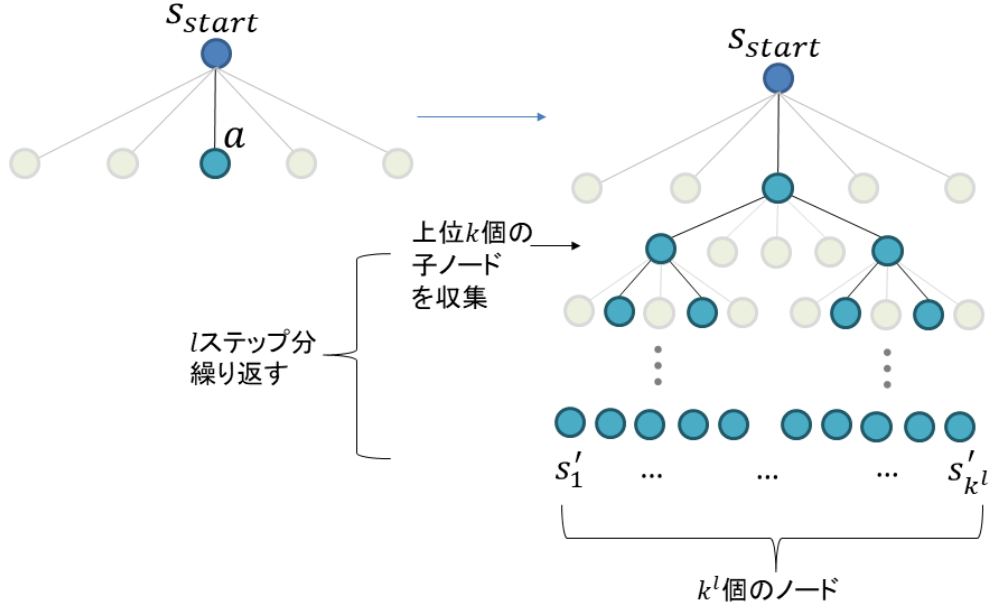


図 3.1: 提案手法のイメージ

とする。2:以下の流れを l ステップ分繰り返す決定木中の注目しているノード s_{now} における訪問回数 $N(s_{now})$ 中の上位 k 個分の行動 a_0, a_1, \dots, a_{k-1} (a_i は $k+1$ 番目に有望な行動とする) を取り出す。 $s_{next_0}, s_{next_1}, \dots, s_{next_{k-1}}$ ($s_{next_i} = T(s_{now}, a_i)$, T は遷移関数) の各ノードに対して step2 の操作を繰り返す。プログラム上は探索の開始地点 s_{start} から step2 中にたどり着く各ノード s_{middle} までの軌道 $\text{traj}_{s_{middle}} (= [a'_{t_0}, a'_{t_1}, \dots, a'_{t_{l-1}}])$ 、 t_i は i 番目のステップを表す) を記録しておく。

3:集めた k の l 乗個のノード $s'_0, s'_1, \dots, s'_{k^l-1}$ のそれぞれ s'_i ($i = 0, 1, \dots, k^l - 1$) に対して以下の操作を再帰的に繰り返す。 s'_i における方策 $P(s'_i)$ 中の最も有望な行動 $a_{promising}$ と $s'_{next_i} = T(s'_i, a_{promising})$ を記録する。そのようにして記録した $s'_{next_0}, s'_{next_1}, \dots, s'_{next_{k^l-1}}$ のそれぞれ s'_{next_j} ($j = 0, 1, \dots, k^l - 1$) にも同様の動作を盤面ノードが決定木の端に辿り着くまで行う。step2 と同様に探索の開始地点 s_{start} から step3 中にたどり着く各ノード s_{middle} への軌道 $\text{traj}_{s_{middle}} (=$

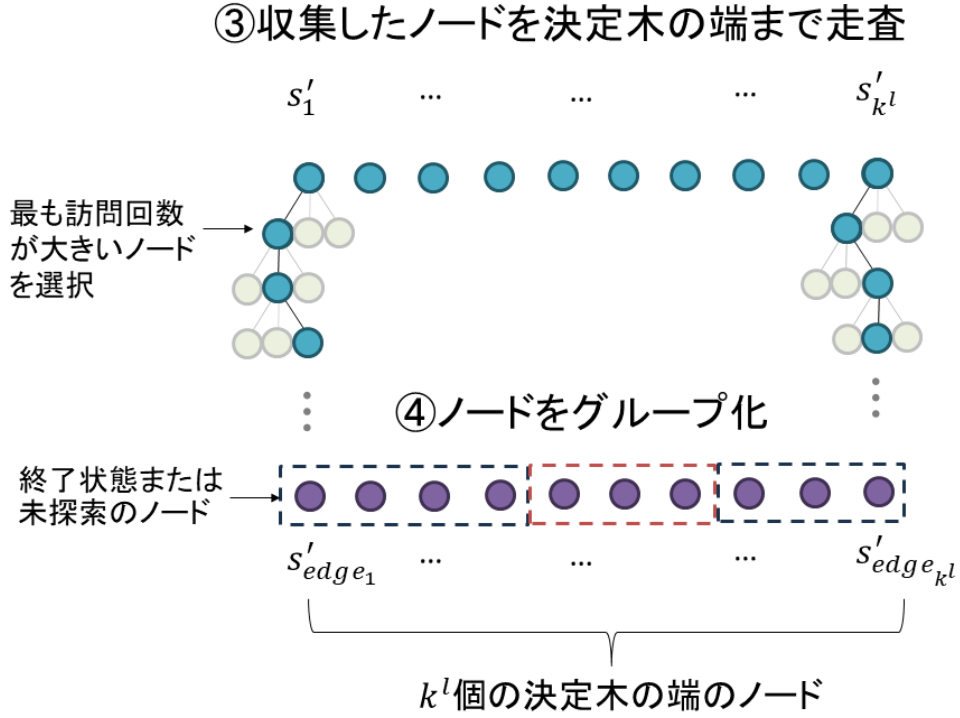


図 3.2: 提案手法のイメージ

$[a'_{t_0}, a'_{t_1}, \dots, a'_{t_{l-1}}, a'_{t_l}, a'_{t_{l+1}}, \dots, a'_{t_p}]$ 、 t_i は i 番目のステップを表す)を記録しておく。4:step3によってたどり着いた k^l 個のノードによる集合 $S = s_{edge_0}, s_{edge_1}, \dots, s_{edge_{k^l-1}}$ を任意の共通項 c によっていくつかの副集合 S_0, S_1, \dots, S_q に分ける。5:共通項で括られた各集合 S_0, S_1, \dots, S_q のうち、最も要素数が多いもの S_{max} 中の各要素 $s_{e_0}, s_{e_1}, \dots, s_{e_u}$ と各要素に対応する軌道 $\text{traj}_{s_{e_0}}, \text{traj}_{s_{e_1}}, \dots, \text{traj}_{s_{e_u}}$ を抽出する。このアルゴリズムを要約するとある局面 s と行動 a の組み合わせから最も辿り着きやすい結末 $O(s, a)$ を抽出し、 $O(s, a)$ に至るまでの複数の道筋を抽出すると表現できる。調査を行う者が複数の軌道を観察できることで、共通する傾向や法則性を見出せるというメリットが存在する。これは最も可能性が高い一つの分岐を示す、といったような情報が単一である手法には不可能である。下に提案手法のイメージと疑似コードを記載する。

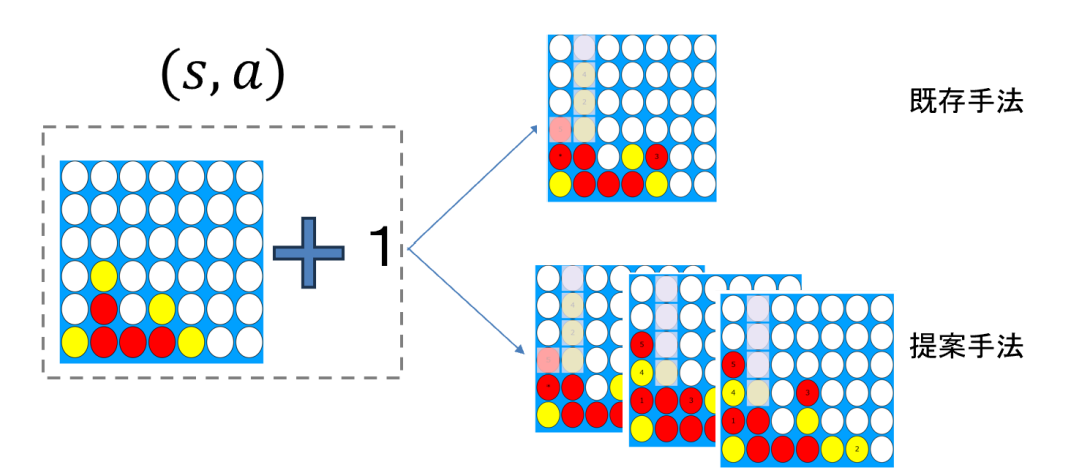


図 3.3: 提案手法のイメージ

次章で示すように connect4 タスクにおいて収集される軌道の集合においては末端部分のいくつかの選択が共通している傾向が確認された。そのため状態と行動の組 (s, a) から $O(s, a)$ という結果を予測する過程で $O(s, a)$ から末尾の数手を取り除いた版面 d にたどり着く傾向の発見などの知識獲得が記載される。

3.2 importance


前章で述べたように、エピソード中の各状態 s の重要度 $I(s)$ の定義を以下のように定めた。

$$I(s) = V[Q'](Q' = [\max Q(s, a), \text{secondMax} Q(s, a), \dots, \text{thirdQuantile} Q(s, a)]) \quad (3.2)$$

つまり、現在の状態 s に対する行動の集合 $A(= a_0, a_1, \dots, a_N)$ とした際の行動価値関数の集合 $Q(s, a_0), Q(s, a_1), \dots, Q(s, a_N)$ のうち大きさが上位 75 % の成分で構成される集合の分散として重要度 $I(s)$ を定義する。これは次の一手で辿り着きうる収益の予想の揺らぎの幅を意味しており、先述した盤面の対称性や悪い選択肢の影響が大きくなる可能性の軽減が期待できる。

Algorithm 3 提案手法のアルゴリズム (part1)

```

1:  $t$ : 手法を適用する探索木
2:  $T$ : 状態遷移関数
3:  $l$ : 盤面の収集を行う手数
4:  $k$ : 一度に集める盤面の数
5:  $\zeta(s, s')$ : ノード  $s$  から  $s'$  までの軌道
6: function MYALGORITHM( $s_{start}, a, l, k$ )
7:    $s_{now} \leftarrow s_{start}$ 
8:    $a_{now} \leftarrow a$ 
9:    $Z_0 \leftarrow \text{COLLECTBOARDS}(s_{now}, a_{now}, l, k)$ 
10:  ( $Z_0 = \zeta(s_{start}, s'_0), \dots, \zeta(s_{start}, s'_{k^l-1})$ )
11:   $Z \leftarrow$  empty list
12:  for each  $\zeta(s_{start}, s'_i)$  in  $Z_0$  do
13:     $\zeta(s_{start}, s_{edge_i}) \leftarrow \text{TRAVERSE}(s'_i, \zeta(s_{start}, s'_i))$ 
14:     $\zeta(s_{start}, s_{edge_i})$  を  $Z$  の末尾に追加
15:  end for
16:  収集された終了状態の集合  $S(= s_{edge_0}, s_{edge_1}, \dots, s_{edge_{k^l-1}})$  を任意の共通
    項  $c$  で副集合  $S_0, S_1, \dots, S_q$  に分割する
17:  最も要素数の多い副集合  $S_{max}$  中の各要素  $s_{e_0}, s_{e_1}, \dots, s_{e_u}$ 
18:  に対応する軌道の集合  $Z_{max}(= \zeta(s_{start}, s_{e_0}), \zeta(s_{start}, s_{e_1}), \dots, \zeta(s_{start}, s_{e_u}))$ 
    を保存
19:    $Z_{max}$ 
20: end function

```

Algorithm 4 提案手法のアルゴリズム (part2)

```

1: function COLLECTBOARDS( $s, a, l, k$ )
2:    $s_{now} \leftarrow T(s, a)$ 
3:    $Z \leftarrow$  empty queue
4:   if  $s$  が未探索のノード ( $N(s) = 0$ ) または終了状態のとき then  $\hookleftarrow Z$ 
5:   end if
6:   訪問回数  $N(s)$  から上位  $k$  の行動  $a_0, a_1, \dots, a_{k-1}(= \alpha)$  を取り出す
7:   for each  $a_i$  in  $\alpha$  do
8:      $s_{next_i} \leftarrow T(s_{now}, a_i)$ 
9:      $\zeta(s_{now}, s_{next_i})(= s_{now}, a_i, s_{next_i})$  を  $Z$  の末尾に追加
10:  end for
11:  if  $l=1$  then  $\hookleftarrow Z$ 
12:  end if
13:   $i \leftarrow 1$ 
14:  while  $i < l$  do
15:    for each  $\zeta(s_{now}, s_j)$  in  $Z$  do
16:       $\zeta(s_{now}, s_j)$  を  $Z$  からポップ
17:      if  $s$  が未探索のノード ( $N(s) = 0$ ) または終了状態のとき then
18:         $\zeta(s_{now}, s_j)$  を  $k$  回  $Z$  の末尾に追加
19:        continue
20:      end if
21:      訪問回数  $N(s)$  から上位  $k$  の行動  $a_0, a_1, \dots, a_{k-1}(= \alpha)$  を取り出す
22:      for each  $a_i$  in  $\alpha$  do
23:         $s_{next_j} \leftarrow T(s_j, a_i)$ 
24:         $\zeta(s_{now}, s_{next_i})(= \zeta(s_{now}, s_j).append(a_i, s_{next_i}))$  を  $Z$  の末尾に追
        加
25:      end for
26:    end for
27:  end while  $\hookleftarrow Z(= \zeta(s_{start}, s'_0), \dots, \zeta(s_{start}, s'_{k^l-1}))$ 
28: end function
29: function TRAVERSE( $s, \zeta(s_{start}, s)$ )
30:    $s_{now} \leftarrow s$ 
31:    $\zeta_r \leftarrow \zeta(s_{start}, s)$ 
32:   while  $s_{now}$  が探索済みかつ終了状態でない do
33:      $a_t \leftarrow \operatorname{argmax}_a N(s_{now}, a)$ 
34:      $s_{now} \leftarrow T(s_{now}, a)$ 
35:      $\zeta_r \leftarrow \zeta_r.append(a_t, s_{now})$ 
36:   end while
37:    $\zeta_r$  を  $\zeta(s_{start}, s)$  の末尾に追加
38:    $\zeta(s_{start}, s) \leftarrow \zeta_r$ 
39:    $s \leftarrow s_{now}$ 
40:    $N(s) \leftarrow N(s) + 1$ 
41:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
42:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
43:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
44:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
45:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
46:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
47:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
48:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
49:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
50:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
51:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
52:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
53:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
54:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
55:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
56:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
57:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
58:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
59:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
60:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
61:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
62:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
63:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
64:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
65:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
66:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
67:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
68:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
69:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
70:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
71:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
72:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
73:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
74:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
75:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
76:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
77:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
78:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
79:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
80:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
81:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
82:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
83:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
84:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
85:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
86:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
87:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
88:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
89:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
90:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
91:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
92:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
93:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
94:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
95:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
96:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
97:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
98:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
99:    $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加
100:   $\zeta(s_{start}, s)$  を  $\zeta(s_{start}, s)$  の末尾に追加

```

第4章

評価実験

提案手法の有効性を示すため2種類の実験を行った。いずれもタスクの対象として connect4 を扱っている。一つ目の実験はコンピュータ同士の対戦データを用いて提案手法による想定図の妥当性の実証を試みた(以下データ実験と表記)。二つ目の実験は自作の connect4 学習支援システムを用いて提案手法のユーザーインタフェースを含んだ優位性の実証を試みた(以下システム実験と表記)。この章ではまず実験の際に用いたタスクである connect4 と使用したモデルである alphazero.baseline について述べる。そののちにデータ実験、システム実験の詳細と結果を記載する。

4.1 connect4[30]

connect4 はボードゲームの一種である。ルールは五目並べに極めて近く、二人のプレイヤーが交互に互いの駒を盤上に置き、最終的に縦、横、もしくは斜めに4つの石を並べたプレイヤーの勝利となる。ただし、五目並べや連珠との相違点として「重力」の存在が挙げられる。この「重力」とは各プレイヤーが石をボード上の最も下の行または既に置かれた石の上にしか置けないという規則を表している。そのため、各プレイヤーの選択肢はボードの列の数と等しい。connect4 の一般的なボードの広さは 6×7 であり 6×7 のコネクト4については1988年に Allis[31] により知識ベースの手法による先番勝利が証明された。Tromp[32] による connect4 の $\alpha - \beta$ 木探索によって導かれた各盤面の最善手とそのデータも一般に公開されている。また、connect4 は盤上に全ての情報が開示されてお

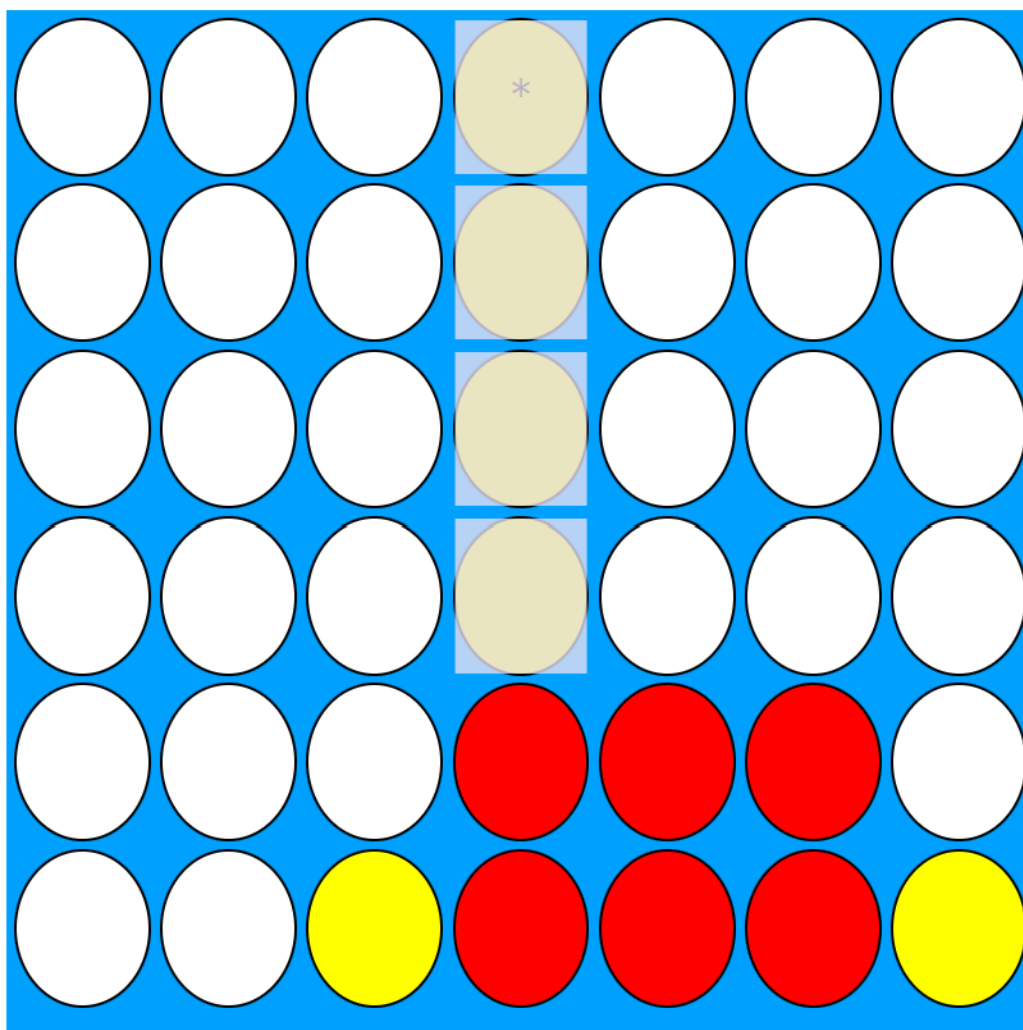


図 4.1: connect4

り、結果もどちらか一方の勝利または引き分けのみであるため 2 人ゼロ和完全情報ゲームに分類される。実験では提案手法を connect4 へ適用した際の step4 における共通項 c として最終的に四つ以上繋がっている石の位置を用いた。

4.2 alphazero_baseline[33]

alphazero_baseline は alphaZero を connect4 用に簡易的に模したネットワークであり、入力は最新の盤面 s_0 、出力は 1×7 (7 はボードの列の数) の方策 $P(s)$ (

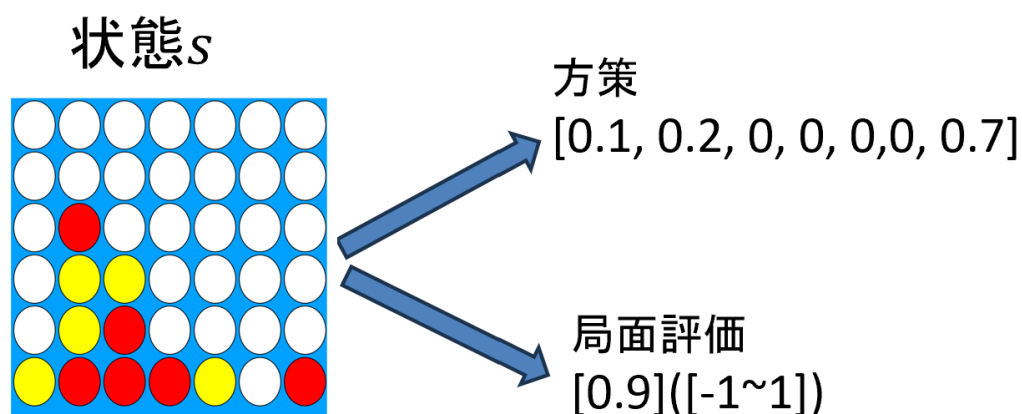


図 4.2: alphazero_baseline の入出力

以下 $s = s_0$ と定める) とスカラー値の局面評価 $V(s)$ (値域は -1 から 1) である。方策は確率分布であり、要素の値が大きいインデックスを選択することで勝利に近づくと予想される。局面評価は入力に対する評価を表しており、1 が入力の手番のプレイヤーにとっての勝利、-1 が敗北の予想を表している。

4.3 データ実験

提案手法と後に述べる比較手法によるゲーム結果（後で勝敗も含める）の予測精度を比較した。

4.3.1 データセット

alphazero_baseline モデル同士の対戦データ 2000 局分 (盤面数:61049) を使用した。いずれもいずれも弱い AI が先番のデータを使用している。これは弱い AI が後番の場合評価関数の変動が極めて小さくなることと、弱い側が先番を選択することが指導において一般的とされるためである。

4.3.2 比較手法

提案手法と比較手法はそれぞれ以下の方式で予測を行う。

- 比較手法: 探索の開始地点から最も訪問回数が大きい選択肢を選び、決定木の端までたどり着いた際にそこで四つ繋がっている石の組み合わせ、位置を記録する。
- 提案手法: 集めた盤面における 4 つ繋がっている石を集計し、出現頻度が高い 4 つの個別の石の位置と出現頻度が高い二つの組み合わせを記録する。組み合わせを二つ記録する理由は下の図のように最終的に繋がっている組み合わせが二つある可能性を考慮するためである。

4.3.3 評価指標

両手法による予測の精度を独自に定義した二つの定義 $fcount$ 、 $fdcount$ によって計測した。 $fcount$ 、 $fdcount$ の詳細な定義は付録 A に記載した。ここでは大まかな定義を述べる。

4.3.3.0.1 $fcount$ 予測の石の組み合わせ単位での精度を示しており、手法による予測 O' と実際の集合 O の積集合 $O \cap O'$ が空集合 ϕ の場合 0、そうでない場合は 1 となる。また、 O' と O が両方とも空集合 ϕ である場合 (実際の結果が引き分けであり、かつそれを正しく予測できている場合) $fcount$ は 1 となる。

4.3.3.0.2 $fdcount$ 予測の個別の石単位での精度を表しており手法による予測 O' と実際の集合 O の積集合 $O \cap O'$ の要素数 $n(O, O')$ を 4 で割った値である。値の最大値は 1 であり、 $n(O, O')$ が 1 を超える場合も $fdcount$ は 1 として扱う。 $fcount$ と同様に O' と O が両方とも空集合である場合 $fcount$ は 1 となる。

4.3.4 実験結果

結果は以下となり、組み合わせ単位の指標である fdcount において比較手法よりも高い精度を示した。表??に表 A.4 に実験結果を示した。いずれの場合も fcount において提案手法は比較手法より高い値を示した。補間の詳細は付録 B に記載した。

表 4.1: 実験結果:データ実験

手数 (盤面数, 補間の有無)	fcount		fdcount	
	提案手法	比較手法	提案手法	比較手法
19-24(9862, 無)	0.60	0.43	0.55	0.62
19-24(9862, 有)	0.63	0.44	0.61	0.63
13-24(21022, 無)	0.52	0.37	0.55	0.56
13-24(21022, 有)	0.55	0.37	0.55	0.56

4.4 システム実験

提案手法の人間に対する有効性を示すため以下のような自作したコネクトフォーの学習支援システムを用いて実験を行った。実験対象者は計 22 人の 10 代～20 代学生（男性 17 名:女性 5 名）となった。

4.4.1 実験手順

実験は被験者一人あたりにつき三回行われ、前半二回が第一段階（提案手法による学習）、後半一回が第二段階（被験者同士の対戦）となった。ここでは第一段階である提案手法の学習について述べる。

4.4.1.0.1 第一段階（提案手法による学習） 提案システムを用いた学習はさらに

- AI システム（alphazero_baseline）との対戦

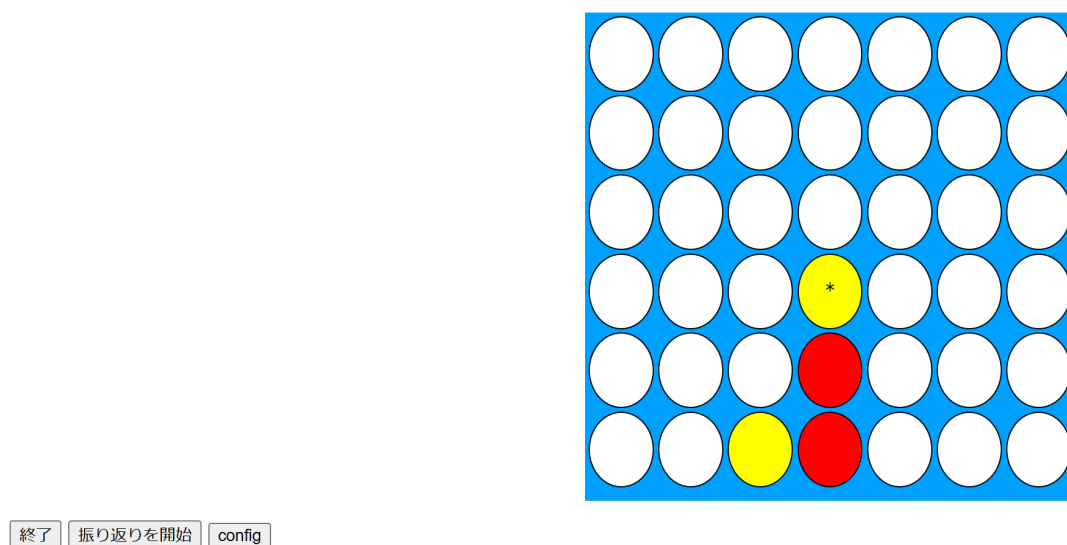


図 4.3: 開始画面

- 提案手法によるゲームの振り返り

の2ステップに分けられる。AIシステムとの対戦が終了するとシステムは「振り返りモード」に移行する。ユーザーはゲームの任意の地点において提案手法または比較手法が提案する予想図を見ることができる。実験の際には被験者をグループA（提案手法による予想図を見せるグループ）とグループB（比較手法による予想図を見せるグループ）に分類した。数字が描かれたボタンは列のインデックスを表しており、各ボタンを押した際にその列を選択した場合の想定図と局面評価を確認できる。

4.4.2 評価指標

システム実験の評価指標は主観評価と客観評価の二つに分けられる。主観評価は被験者による五段階評価であり、「タスクの熟達度に関連する質問」と「タスクの楽しさ・面白さに関連する質問」の二つに分けられる。具体的な質問事項は付録Dに記載する。客観評価はグループAの被験者とグループBの被験者の対戦成績である。

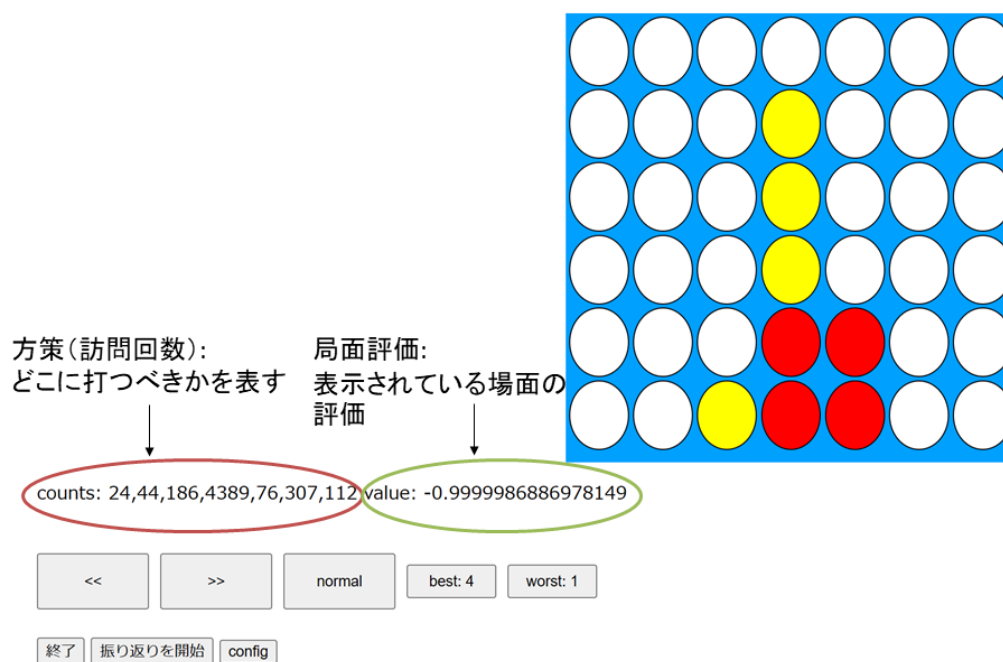


図 4.4: 振り返り画面

4.4.3 実験結果

表 4.2: あああといいいの予測誤差

	2019		2018		2017	
モデル	ああ	いい	ああ	いい	ああ	いい
Naive	1	1	1	1	1	1
TCN	1.0895	0.9032	1.4791	0.9198	1.2888	0.8555
LSTM	1.0384	0.9295	1.4917	0.9725	1.1627	0.8541
提案手法	1.0977	0.8698	1.3824	0.9439	1.2061	0.8516

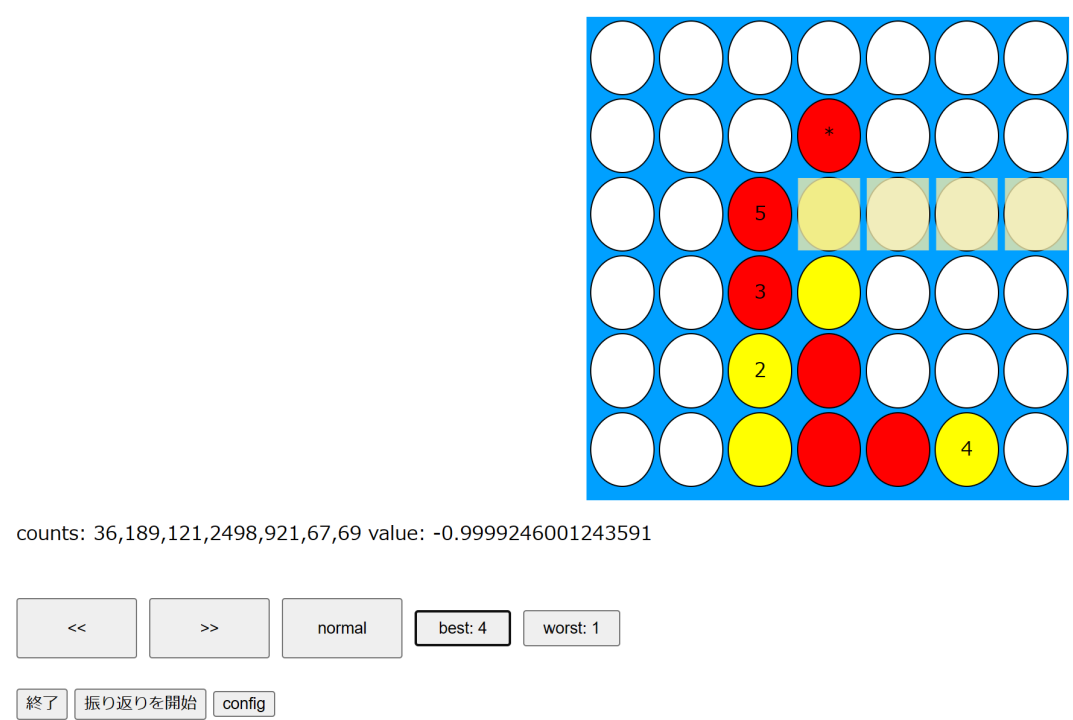


図 4.5: 予想図の表示

第5章

結論

本論文では
今後の課題を以下に挙げる．

- の向上
必要がある．
- への応用
を行いたい．

- の改善

今後，取り組みたい．

謝辞

本研究を行うにあたり親身に相談に乗っていただき，ご指導してくださった萩原将文教授，ならびに共に問題解決，議論，相談，および実験に付き合ってくださいました研究室の先輩方，同期の皆様，実験に参加してくださった大学の友人達に深く感謝いたします。誠にありがとうございました。

参考文献

- [1] et al. Silver David. “A general reinforcement learning algorithm that masters chess, shogi and Go through self-play”. In: *Science* 362.6419 (2018).
- [2] Andreas Blattmann Robin Rombach, Patrick Esser Dominik Lorenz, and Björn Ommer. “High-resolution image synthesis with latent diffusion models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (2021).
- [3] Jeff Wu Long Ouyang et al. “Training language models to follow instructions with human feedback”. In: *Advances in Neural Information Processing Systems* 35 (2022).
- [4] “G7 Hiroshima Leaders ’ Communiqué”. In: (2023).
- [5] “新しい資本主義のグランドデザイン及び実行計画”. In: (2023), pp. 16–19.
- [6] Richard S. Sutton and Andrew G. Barto. “reinforcement learning”. In: (2018), pp. 251–272.
- [7] et al. Mnih Volodymyr. “Playing atari with deep reinforcement learning”. In: *arXiv preprint arXiv* 1312.5602 (2013).
- [8] et al. Hessel Matteo. “Rainbow: Combining improvements in deep reinforcement learning”. In: *Proceedings of the AAAI conference on artificial intelligence* 32.1 (2018).
- [9] et al. SILVER David. “Mastering the game of Go with deep neural networks and tree search”. In: *nature* 529.7587 (2016).
- [10] T. Romstad, M. Costalba, and et al. J. Kiiski. “A strong open source chess engine”. In: <https://stockfishchess.org/> (2024 年 1 月 15 日確認) () .

- [11] T. Yamaoka. “DeepLearningShogi”. In: <https://github.com/TadaoYamaoka/DeepLearningShogi> (2024 年 1 月 15 日確認) ().
- [12] Doshi-Velez, Finale, and Been Kim. “Towards a rigorous science of interpretable machine learning”. In: *arXiv preprint arXiv 1702.08608* (2017).
- [13] Tobias Huber et al. “Local and Global Explanations of Agent Behavior: Integrating Strategy Summaries with Saliency Maps”. In: *Artificial Intelligence* 301.103571 (2021).
- [14] Hou, Xiaodi, and Liqing Zhang. “Saliency detection: A spectral residual approach”. In: *2007 IEEE Conference on computer vision and pattern recognition* (2007).
- [15] et al Selvaraju Ramprasaath R. “Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization”. In: *Proceedings of the IEEE international conference on computer vision* (2017).
- [16] et al Pang Wei Koh Kai-Siang Ang. “On the accuracy of influence functions for measuring group effects”. In: *arXiv preprint arXiv 1711.11279* (2017).
- [17] et al. McGrath Thomas. “Acquisition of Chess Knowledge in AlphaZero”. In: *Proceedings of the National Academy of Sciences* 119.47 119.47 (2022).
- [18] et al. Zeev Fine Ofer Sharni. “DecodeChess”. In: <https://decodechess.com/> (2024 年 1 月 15 日確認) (2017).
- [19] Anurag Koul Sam Greydanus and Alan Fern Jonathan Dodge. “Visualizing and Understanding Atari Agents”. In: *International conference on machine learning* (2018).
- [20] Yuanfeng Pang and Takeshi Ito. “Visualizing and Understanding Policy Networks of Computer Go”. In: *Journal of Information Processing* 29 (2021).

- [21] Swabha Swayamdipta Alon Jacovi, Yanai Elazar Shauli Ravfogel, and Yoav Goldberg Yejin Choi. “Contrastive Explanations for Model Interpretability”. In: *arXiv preprint arXiv* 2103.01378 (2021).
- [22] Utkarsh Soni Aditi Mishra and Chris Bryan Jinbin Huang. “Why? Why not? When? Visual Explanations of Agent Behaviour in Reinforcement Learning”. In: *arXiv preprint arXiv* 2014.02818v2 (2021).
- [23] Rahul Nair Jasmina Gajcin, Radu Marinescu Tejaswini Pedapati, and Ivana Dusparic Elizabeth Daly. “Contrastive Explanations for Comparing Preferences of Reinforcement Learning Agents”. In: *arXiv preprint arXiv* 2112.09462 (2021).
- [24] Lisa Torrey and Matthew Taylor. “Teaching on a budget: Agents advising agents in reinforcement learning”. In: *In Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems* (2013).
- [25] Ece Kamar Ofra Amir and Barbara J. Grosz Andrey Kolobov. “Interactive teaching strategies for agent training”. In: *International Joint Conferences on Artificial Intelligence* (2016).
- [26] Xian Wu Wenbo Guo and Xinyu Xing Usman Khan. “EDGE: Explaining Deep Reinforcement Learning Policies”. In: *Advances in Neural Information Processing Systems 34* (2021).
- [27] David Wu Andrew Lee and Mike Lewis Emily Dinan. “Improving Chess Commentaries by Combining Language Models with Symbolic Reasoning Engines”. In: *arXiv preprint arXiv* 2212.08195 (2022).
- [28] Simon Viennot¹ Kokoro Ikeda¹ and Naoyuki Sato¹. “Detection and labeling of bad moves for coaching go”. In: *2016 IEEE Conference on Computational Intelligence and Games (CIG)* (2016).

- [29] Masahiko Osawa Vincent Richard and Michita Imai. “Determining Strategies behind Moves in the Game of Go”. In: *Cloud Network Robotics* 117.95 (2017).
- [30] James Dow Allen. “The Complete Book of Connect 4”. In: (2010).
- [31] Victor Allis. “A Knowledge-based Approach of Connect-Four”. In: (1988).
- [32] John Tromp. “Connect-4 Data Set”. In: *UC Irvine Machine Learning Repository*, <https://archive.ics.uci.edu/dataset/26/connect+4> (2024 年 1 月 15 日確認) (1995).
- [33] Bo Zhou. “AlphaZero baseline - ConnectX”. In: <https://github.com/PaddlePaddle/PARL/tree/master> (2020).
- [34] Peter Cnудde. “1k connect4 validation set”. In: *Kaggle*, <https://www.kaggle.com/petercnudde/connect4-validation-set> (2024 年 1 月 15 日確認) (2020).
- [35] Peter Cnудde. “Scoring connect-x agents”. In: *Kaggle*, <https://www.kaggle.com/code/petercnudde/connect-x-agents/notebook> (2024 年 1 月 15 日確認) (2020).

付録 A

データ実験の詳細

A.1 使用したモデルの詳細

第三章で述べたとおり使用した対戦データは弱い AI を先番とし、強い AI を後手としている。AI の強さは一手ごとの探索を行う時間 (time) と付録 C で述べる C_{puct} の値によって調整した。time と C_{puct} はいずれも値が大きい程モデルは強くなると考えられる。対戦データ生成時のパラメータは以下の表の幅からゲームごとにランダムな値を採用した。これはパラメータの値を変化させることでゲームデータに多様性を持たせるためである。

表 A.1: 対戦データのパラメータ

モデル	強	弱
time	3-5	0-2
C_{puct}	0.8-1	0-0.5

A.2 対戦結果の詳細

2000 ゲームのうち 1983 ゲームは強い AI(後番) の勝利となった。またゲームごとの手数は 75 % のゲームが 36 手以内で終了している。以下にゲームの終了手数の累計グラフを示す。

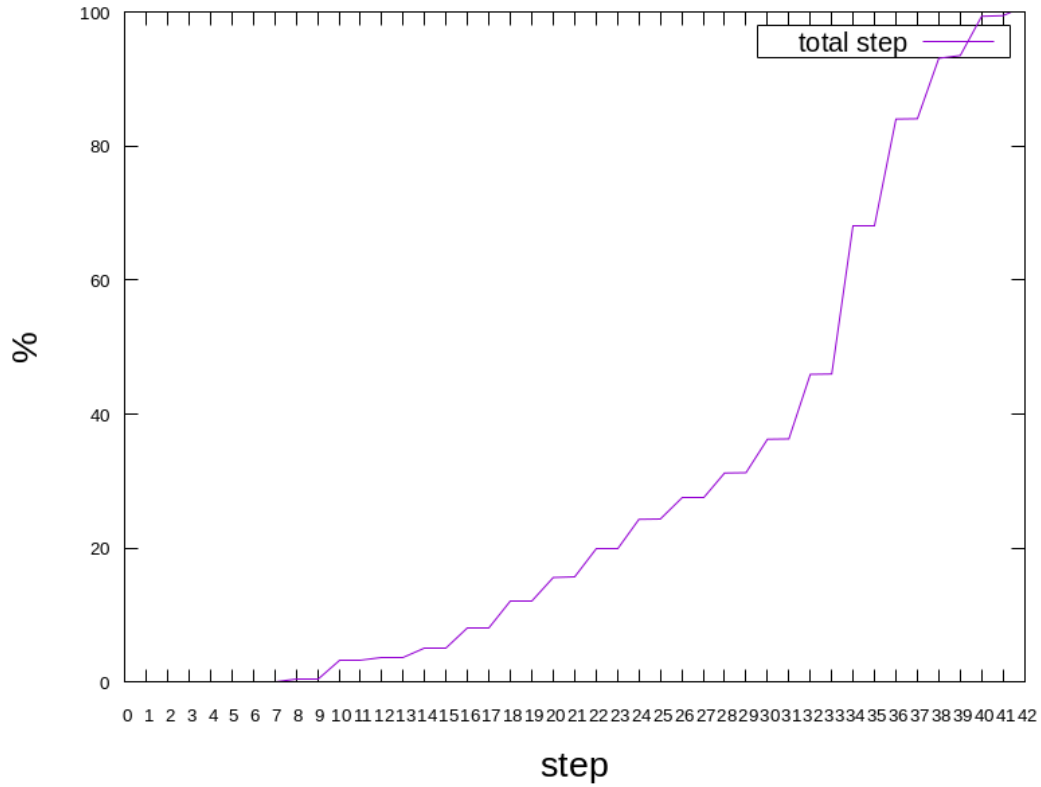


図 A.1: 終了手順の分布

A.3 評価指標の詳細

本論文が提案する提案指標 (fcount, fdcount) の計算において、盤面の座標は以下のように定められる。ゲーム終了状態には以下のように4つ以上連続してつながっている石の座標 F_s (fatal stone の集合) とその組み合わせ F_g (fatal group の集合) を記録する。下の図の盤面は 17, 18, 19, 20, 24, 31, 38 の7つの fatal stone と $[17, 18, 19, 20]$, $[17, 24, 31, 38]$ の二つの fatal group を持つ fcount, fdcount は実際のゲームにおける fatal stone と fatal group の集合 (それぞれ R_s, R_g とする) と AI の予測による fatal stone と fatal group の集合 (F_s, F_g) を比較する指標である。どちらも値が高ければ高い程予測の精度が高いことを表す。

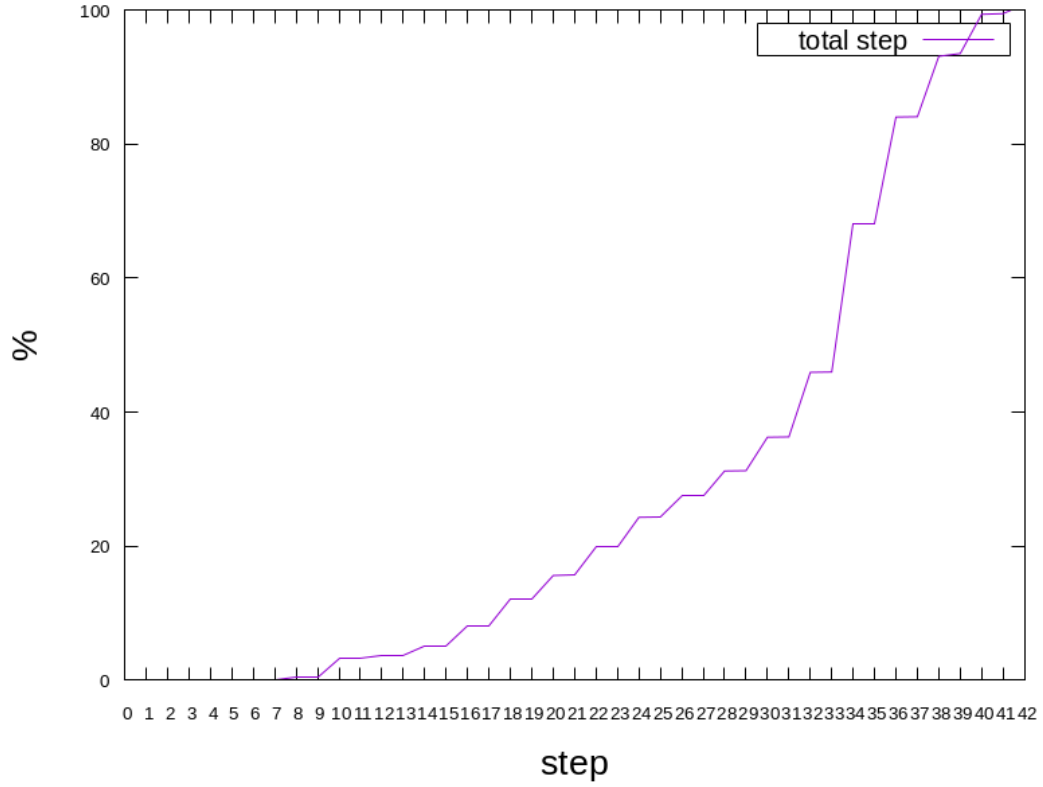


図 A.2: 終了手順の分布

A.3.1 fcount

fcount は fatal group の精度を示す二値 (0 または 1) の指標である。

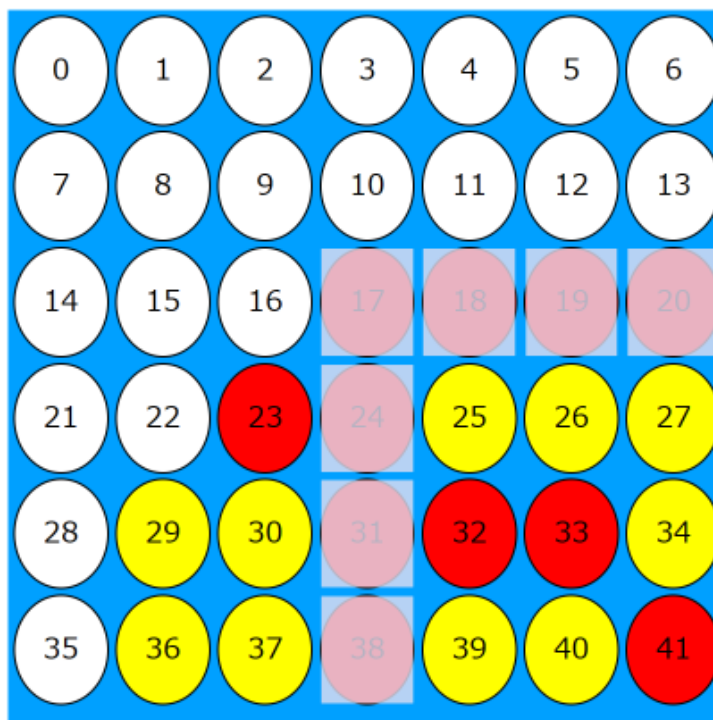
$$\text{fcount} = 1 \text{ If } F_g \cap R_g = \phi \text{ Else } 0 \quad (\text{A.1})$$

例外として $R_g = \phi$ (引き分け) の場合, F_g も空集合であるならば fcount は 1 となる。

A.3.2 fdcount

fdcount は fatal stone の精度を示す。fdcount の値域は $[0, 1]$ である。(Count() は集合の要素数を数える関数)

$$\text{fdcount} = \frac{\text{Count}(F_s \cap R_s)}{4} \quad (\text{A.2})$$



Fatal Stone: $\{17, 18, 19, 20, 24, 31, 38\}$

Fatal Group: $\{[17, 18, 19, 20], [17, 24, 31, 38]\}$

図 A.3: fatalStone と fatalGroup

例外として $R_s = \phi$ (引き分け) の場合, F_s も空集合であるならば fdcount は 1 となる。

A.4 データ実験における提案指標の計測

1. 比較手法: 付録?で述べた決定木の走査によりたどり着いた F_s, F_g によって fcount, fdcount を計算した
2. 提案手法: 第三章で述べたアルゴリズムによって収集された最終状態の集合 $S = s_{edge_0}, s_{edge_1}, \dots, s_{edge_{kl-1}}$ を指標ごとにグループ化する。

- fcount: fatal group によって S をグループ化してできた集合 $S_{g_0}, S_{g_1}, \dots, S_{g_n}$ (S_{g_i} は組み合わせ g_i が fatal group となっている盤面の集合) を要素が多い順に二つ取り出す。抽出された二つの集合 $S_{g_{m_1}}, S_{g_{m_2}}$ における g_{m_1}, g_{m_2} で構成される集合を F_g とし、fcount を計算した。
- fdcount: fatal stone によって S をグループ化してできた集合 S_0, S_1, \dots, S_{41} (S_i は番号 i の座標が fatal stone となっている盤面の集合) を要素数が多い順に4つ取り出す。抽出された4つの集合 $S_{m_1}, S_{m_2}, S_{m_3}, S_{m_4}$ における m_1, m_2, m_3, m_4 で構成される集合を F_s とし、fdcount を計算した

A.5 データ実験に使用したモデルの詳細

第4章に記載した表 A.4 の結果を求める際に用いたモデルのパラメータは以下である。

表 A.2: データ実験:使用モデルのパラメータ

time	対戦時の後番のモデルと同じ
C_{puct}	対戦時の後番のモデルと同じ
k (提案手法のみ)	4
l (提案手法のみ)	2

C.1 の通り、対戦データが生成された際のモデルのパラメータを使用している。そのため本手法はモデルの構造へのアクセスが可能な場合を想定したホワイトボックス的アプローチの実験であると言える。モデルの time, C_{puct} の値を固定して同様の手法を適用した場合の実験結果を次のセクションで示す。

A.6 グレーボックス的手法

モデルのパラメータを固定し、再び第四章のデータ実験を行った。パラメータの値を固定しているため、本実験はモデルの具体的なパラメータの値へのア

クセスが不可能な場合にも適用可能なグレーボックス的手法であると言える。

表 A.3: データ実験 (追加):使用モデルのパラメータ

time	5
C_{puct}	1
k (提案手法のみ)	4
l (提案手法のみ)	2

実験の結果は以下ようになった。第四章に記載したホワイトボックス的手法と同様に提案手法は `fcount` において比較手法より高い値を示した。

表 A.4: 実験結果:データ実験

手数 (盤面数, 補間の有無)	fcount		fdcount	
	提案手法	比較手法	提案手法	比較手法
19-24(9862, 無)	0.60	0.44	0.61	0.63
19-24(9862, 有)	0	0.44	0.61	0.63
13-24(21022, 無)	0.53	0.37	0.55	0.56
13-24(21022, 有)	0.55	0.37	0.55	0.56

付録 B

各種アルゴリズムの詳細

B.1 比較手法のアルゴリズム

比較手法の疑似コードを以下に示す。疑似コード中の `Traverse()` は第三章の疑似コードと同一である。

B.2 提案手法におけるニューロ補間



第四章におけるデータ実験、システム実験ではいずれも手法のニューラルネットワークによる補間を行っている。第四章で示した疑似コードでは未探索のノードにたどり着いた際は走査を終了する。ニューラルネットワークによる補間とはこの場合、方策 $P(s, a)$ で訪問回数 $N(s, a)$ を代用し走査を継続することである。以下にニューロ補間を行う場合の疑似コードを示す（変更部分に下線）。比較手法に対してニューロ補間を行う場合も同様に `Traverse()` のコードを変更する。

B.3 システム実験における提案手法の変更

システム実験では画面の左下に盤面 s の訪問回数 $N(s)$ と局面評価 $V(s, a)$ を表示する。このとき $V(s, a)$ の絶対値（どちらのプレイヤーの勝利に近いか）と予想図の勝敗が異なる場合、ユーザーの混乱を招く可能性がある。そのため、提案手法では勝敗が $V(s, a)$ の絶対値と一致する軌道を表示するように修正を施した。システム実験用に修正された疑似コードは以下となる。（下線が変更部分）

Algorithm 5 比較手法のアルゴリズム

```

1:  $t$ : 手法を適用する探索木
2:  $T$ : 状態遷移関数
3:  $\zeta(s, s')$ : ノード  $s$  から  $s'$  までの軌道
4: function COMPAREALGORITHM( $s, a$ )
5:    $s_n \leftarrow T(s, a)$ 
6:    $\zeta(s, s_n) \leftarrow s, a, s_n$ 
7:    $\zeta \leftarrow s_n, \zeta(s, s_n)$    $\zeta$ 
8: end function
9: function TRAVERSE( $s, \zeta(s_{start}, s)$ )
10:   $s_{now} \leftarrow s$ 
11:   $\zeta_r \leftarrow \zeta(s_{start}, s)$ 
12:  while  $s_{now}$  が探索済みかつ終了状態でない do
13:     $a_t \leftarrow \operatorname{argmax}_a N(s_{now}, a)$ 
14:     $s_n \leftarrow T(s_{now}, a_t)$ 
15:     $\zeta_r.append(a_t, s_n)$ 
16:     $s_{now} \leftarrow s_n$ 
17:  end while   $\zeta_r$ 
18: end function

```

Algorithm 6 提案手法のアルゴリズム (ニューロ補間あり)

```

1: function COLLECTBOARDS( $s, a, l, k$ )
2:    $s_{now} \leftarrow T(s, a)$ 
3:    $Z \leftarrow$  empty queue
4:   if  $s$  が終了状態のとき then  $\leftarrow Z$ 
5:   end if
6:   if  $s$  が未探索のとき then
7:     方策  $P(s)$  から上位  $k$  の行動  $a_0, a_1, \dots, a_{k-1}(= \alpha)$  を取り出す
8:   else
9:     訪問回数  $N(s)$  から上位  $k$  の行動  $a_0, a_1, \dots, a_{k-1}(= \alpha)$  を取り出す
10:  end if
11:  for each  $a_i$  in  $\alpha$  do
12:     $s_{next_i} \leftarrow T(s_{now}, a_i)$ 
13:     $\zeta(s_{now}, s_{next_i})(= s_{now}, a_i, s_{next_i})$  を  $Z$  の末尾に追加
14:  end for
15:  if  $l=1$  then  $\leftarrow Z$ 
16:  end if
17:   $i \leftarrow 1$ 
18:  while  $i < l$  do
19:    for each  $\zeta(s_{now}, s_j)$  in  $Z$  do
20:       $\zeta(s_{now}, s_j)$  を  $Z$  からポップ
21:      if  $s$  終了状態のとき then
22:         $\zeta(s_{now}, s_j)$  を  $k$  回  $Z$  の末尾に追加
23:      continue
24:    end if
25:    if  $s$  が未探索のとき then
26:      方策  $P(s)$  から上位  $k$  の行動  $a_0, a_1, \dots, a_{k-1}(= \alpha)$  を取り出す
27:    else
28:      訪問回数  $N(s)$  から上位  $k$  の行動  $a_0, a_1, \dots, a_{k-1}(= \alpha)$  を取り
        出す
29:    end if
30:    for each  $a_i$  in  $\alpha$  do
31:       $s_{next_j} \leftarrow T(s_j, a_i)$ 
32:       $\zeta(s_{now}, s_{next_i})(= \zeta(s_{now}, s_j).append(a_i, s_{next_i}))$  を  $Z$  の末尾に追
        加

```

Algorithm 7 提案手法のアルゴリズム (ニューロ補間あり)

```

1: function TRAVERSE( $s, \zeta(s_{start}, s)$ )
2:    $\zeta(s_{start}, s)$  から  $s_{start}$  の次の行動  $a$  を取り出す
3:    $v \leftarrow V(s, a)$ 
4:    $s_{now} \leftarrow s$ 
5:    $\zeta_r \leftarrow \zeta(s_{start}, s)$ 
6:   while  $s_{now}$  が終了状態でない do
7:     if  $s$  が未探索のとき then
8:        $a_t \leftarrow \operatorname{argmax}_a P(s_{now}, a)$ 
9:     else
10:       $a_t \leftarrow \operatorname{argmax}_a N(s_{now}, a)$ 
11:    end if
12:     $s_n \leftarrow T(s_{now}, a_t)$ 
13:     $\zeta_r.append(a_t, s_n)$ 
14:     $s_{now} \leftarrow s_n$ 
15:  end while
16:  if  $v$  と  $V(s_{now})$  の絶対値が異なるとき then  $\leftarrow \text{null}$ 
17:  end if  $\leftarrow \zeta_r$ 
18: end function

```

付録 C

alphazero_baseline

C.1 ニューラルネットワーク

モデルの構成は以下である。また、モデルの訓練過程は以下のステップの繰り返しによって構成されている。使用したテストデータ [34] は - 木探索による connect4 の解から生成されている [35]。

1. 500 ゲーム分の自己対戦を行う
2. 自己対戦によって収集した盤面を入力としてネットワークを訓練
3. ネットワークを教師データによって評価
4. 新しいネットワークを用いた AI と訓練前の最善のネットワークを用いた AI による対戦を 50 ゲーム分行い 6 割以上の勝率を記録した場合、新しいネットワーク最善のネットワークとして保存する

本論文の第 3 章におけるシステム実験で用いた「強い AI」のニューラルネットワークは上記をステップを 200 エポック分実行して訓練されたモデルを使用している。

また、上記の 2. におけるネットワーク訓練時のパラメータの値は以下を用いた。

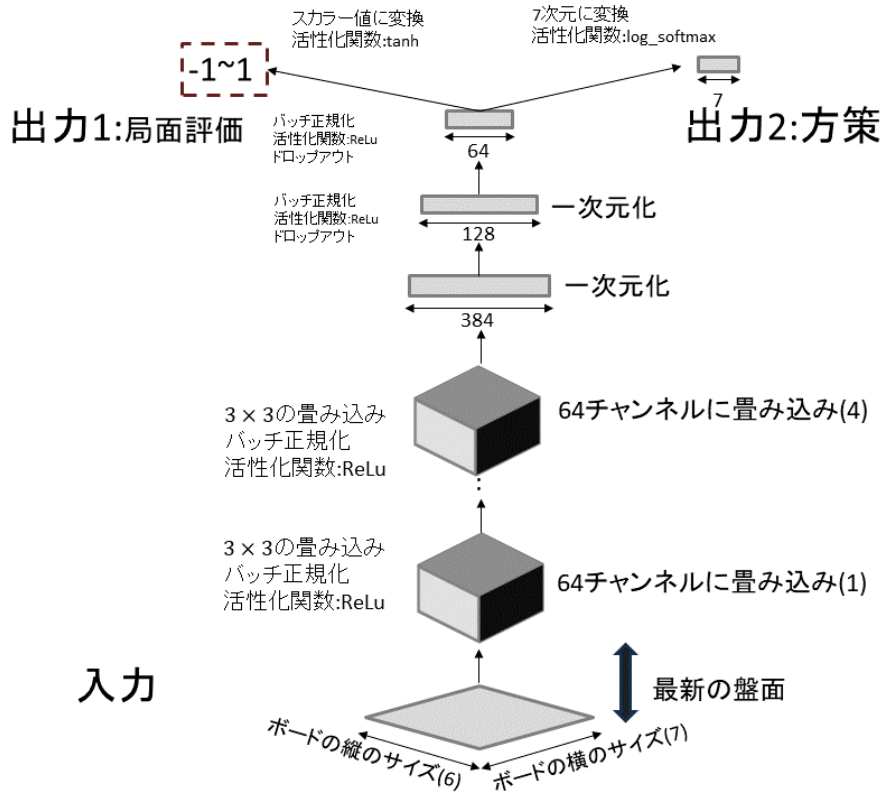


図 C.1: alphazero_baseline ネットワークの構成

表 C.1: ネットワーク訓練時のパラメータ名と値

学習率 (lr)	0.001
dropout	0.3
epochs	5
batch_size	64
num_channels	64

C.2 alphazero_baseline のパラメータ更新

alphazero_baseline のパラメータ更新は第二章で述べた手順とほぼ同一であるが PUCT スコア $U(s, a)$ の定義における C_{cpuct} はハイパーパラメータである。

($N(s)$, $N(s, a)$ はそれぞれ s , (s, a) に対して探索を行った回数)


$$U(s, a) = C_{\text{cpuct}} P(s, a) \frac{\sqrt{N(s)}}{1 + N(s, a)} \quad (\text{C.1})$$


$Q(s, a)$ は以下のように更新される。 $(s_c$ は現在の探索ノード s から見た最も *PUCT* スコア $U(s, a)$ と $Q(s, a)$ の和の高い子ノード)

$$Q(s, a) = \frac{N(s, a)Q(s, a) + V(s_c)}{N(s, a) + 1} \quad (\text{C.2})$$

Algorithm 8 PV-MCTS in alphazero-baseline (変更部分)

```

1:  $t$ : 決定木
2:  $T$ : 遷移関数
3:  $N(s, a)$ :  $(s, a)$  の組み合わせを探索した回数
4:  $Q(s, a)$ : 行動価値関数 (Explore( $s$ ) の平均)
5:  $W(s, a)$ : 行動価値の総和 ( $W(s, a) = Q(s, a)N(s, a)$ )
6:  $P(s, a)(= P(s_n), s_n = T(s, a))$ :
7: ニューラルネットワークから出力された方策
8:  $V(s, a)(= V(s_n), s_n = T(s, a))$ :
9: ニューラルネットワークから出力された局面評価
10: function TREEPOLICY( $s$ )
11:   if  $s$  が探索されていない子ノードを持つとき then
12:      $s_c \leftarrow T(s, a)$  ( $s_c$  は未探索のノード)
13:     INITNODE( $s_c$ )
14:       $a$ 
15:   else
16:     以下の PUCT スコア  $U(s, a)$  を計算
17:     
$$U(s, a) = C_{\text{puuct}} P(s, a) \frac{\sqrt{N(s)}}{1+N(s, a)}$$

18:     ( $N(s) = \Gamma N(s, a)$ )
19:     以下のように  $a$  を求める
20:      $a = \text{argmax}_a (Q(s, a) + U(s, a))$ 
21:       $a$ 
22:   end if
23: end function
24: function BACKPROPAGATE( $\zeta, G$ )
25:   for each node-action pair  $(s, a)$  in  $\zeta$  do
26:      $N(s, a) \leftarrow 0$ 
27:      $W(s, a) \leftarrow 0$ 
28:      $Q(s, a) \leftarrow 0$ 
29:   end for
30: end function
31: function INITNODE( $s$ )

```

```

32:   for each action  $a$  from  $s$  do 49

```

```

33:      $N(s, a) \leftarrow N(s, a) + 1$ 

```

```

34:      $W(s, a) \leftarrow W(s, a) + G$ 

```

```

35:      $Q(s, a) \leftarrow \frac{W(s, a)}{N(s, a)}$ 

```

付録 D

システム実験の詳細

システム実験では全三回にわたる実験を行い、被験者同士の対戦を行う第三回以外の二回は被験者が AI との対戦と AI を用いた振り返りを行う。

D.1 パラメータ

振り返りモードでは

D.2 異なる期間

図