Vol.2019-GI-41 No.9
2019/3/8

深層学習囲碁プログラムを用いた場合の手加減に関する研究

Shi Yuan [†1, a]　Fan Tianwen[†2, b]　Li Wanxiang[†3, c]　池田 心[†4, d]

概要：深層学習と強化学習の発展に伴い，コンピュータ囲碁の実力は人間のトッププレイヤを超えた．一方で，初中級者の教育やエンタテインメント方面での研究はまだ十分に行われていない．例えば初中級者と対戦して楽しませるためにはコンピュータ側に意図的に悪い手を打たせる必要があるが，評価関数にノイズを加えるなどの静的な方法，現在の勝率に応じて勝率の低い手を打たせるなどの動的な方法，それぞれに課題がある．
本稿では，AlphaGo Zero モデルに基づくプログラム Leela Zero と ELF OpenGo を採用したうえで，既存の勝率制御法を再現し，その有効性を検証する．そしてプログラムのモデルが変わったこと，より強くなったことによる新たな課題を発見し，その緩和法を提案し，評価することを試みる．

キーワード：手加減コンピュータ囲碁，エンタテインメント，教育，深層学習，AlphaGo Zero.

Research on Strength Control Go AI Using Deep Learning Methods

SHI YUAN [†1, a]　FAN TIANWEN[†2, b]　LI WANXIANG[†3, c]　KOKOLO IKEDA [†4, d]

Abstract: In recent year, Computer Go AI has exceeded the top-level human player by the advancement of Deep Learning and Reinforcement Learning techniques. However, the other approach for “Entertainment Go AI” or “Coaching Go AI” are still received less attention. But several approaches have been made to entertain the beginner or intermediate level of player. For example, to control strength by “static methods” that adding noise into evaluation function, or “Dynamic” approach which selects low winning ratio move according to current winning ratio, etc. However, it still appears that some task to be improved.
In this paper, we present the reproduction and validation of the existing strength control method using the "AlphaGo Zero"-based program Leela Zero and ELF OpenGo. Also, it is reasonable to try to find and evaluate some novelty research approaches and ideas using newer architecture and stronger methods.

Keywords: Strength Control Go AI, Entertainment, Coaching, Deep Learning, AlphaGo Zero.

1. はじめに

囲碁は非常に高度な判断が必要なゲームとして，長い間人工知能の目標であった．近年，深層学習技術の発展に伴い，AlphaGo[1]が初めてトッププロ棋士に勝利した．さらには改良版の AlphaGo Zero[2]の仕組みが公開され，そのモデルを使用した多数の強い囲碁プログラムが作成された．これらプログラムの強さは殆どの人間にとってもう十分あるいは過剰であり，強さに関する研究とは別に楽しませるプログラム・教えるプログラムの研究が重要度を増していると考える．囲碁を上手に指導することができる人やその機会は限られており，人間に指導することができるコンピュータ囲碁に関する研究は重要な課題であると考える．

楽しませたり指導したりする方法にはさまざまなものがあるが，実際に相手をすること（接待碁・指導碁）はその中心的なものになる．高段者ならばともかく，中級者では昨今のプログラムに九子のハンディキャップでも勝てないかもしれない．一般にハンデは少ないほうが好まれるため，強い側が“手加減”することが必要になる．しかし，単に悪い手を打つことで手加減すると，それは「変な手で手加減されたから勝てた」とネガティブな感情をプレイヤに与えてしまうだろう．

過去に，囲碁プログラムに自然な手加減を行わせる研究が提案されたが，これはアマチュア四段程度の囲碁プログラムを用いたものだった．本論文では，まずこれを新世代の囲碁プログラム Leela Zero[3]と ELF OpenGo[4]に実装し，うまく動作するか確認する．そして，用いられる技術が変わったことで生じる新たな課題を紹介し，どのような展望が考えられるかを考察する．

2. 関連研究

2.1 深層学習コンピュータ囲碁

コンピュータ囲碁では，状態評価関数を作成することの困難さから棋力が級位レベルで伸び悩む時代があった．それを解決するモンテカルロ木探索（MCTS）は大きなブレイクスルーであり，棋力はアマ高段レベルに達した．そして，計算機能力の増大と深層学習技術の発展によって，2016 年，AlphaGo[1]が最強棋士の一人 Lee Sedol に勝利するという金字塔が打ち立てられた．AlphaGo は深層畳込みニュー

†1†2†3†4　北陸先端科学技術大学院大学
Japan Institute of Science and Technology
a) shiyuan@jaist.ac.jp
b) fantianwen@jaist.ac.jp
c) liwanxiang@jaist.ac.jp
d) kokolo@jaist.ac.jp

ⓒ 2019 Information Processing Society of Japan

ラルネットワークを用いて人間の棋譜を学習して行動評価関数を作成し，これを強化学習で強めたうえで，自己棋譜生成の後に状態評価関数の作成に成功している．これらをMCTS と組み合わせることで強いプログラムができた．その後継の AlphaGo Zero[2]は，完全に人間の知識を用いず，AlphaGo の方策予測ネットワークと局面の価値予測ネットワークを組合せたものを自己対戦で強化学習し，AlphaGo よりも強いものを作成している．

また，AlphaGo Zero と同様な手法を用い，オープンソースの囲碁プログラムもいくつか開発・公開されている．その代表的なものが ELF OpenGo[4]と Leela Zero[3]である．これらは AlphaGo Zero と同じく，今の盤面の黒石と白石の配置，数手の履歴，そして現在の手番を深層ニューラルネットワークに入力して，次の各着手の選択確率（policy）と今の盤面の評価（value）を出力する仕組みとなっている．着手決定では，policy と今までの探索の局面価値によって評価値を計算し，MCTS での展開ノードを決める．探索木が末端のノードにたどり着いたら，value によって末端の盤面を評価し，親ノードまで全部の経路のノードの評価値を更新する．MCTS が終わったら，勝率や訪問回数に応じて好ましいものを選ぶ．

2.2 手加減方法

池田らは，人間プレイヤを楽しませる接待碁のためには，概ね 6 つの要素，（A）相手モデルの獲得，（B）形勢の誘導，（C）不自然な着手の抑制，（D）多様な戦略，（E）適切な投了タイミングと思考時間，（F）感想戦・検討・おしゃべり，などが必要であることを提唱した[5]．

手加減には一般に，a）常に一定の弱さを演出しようとする（例えば探索リソースを減らす）方法と，b）現在の形勢に応じて手加減の度合を決める（優勢なら悪い手を打つ）方法がある．それぞれ一長一短があり，前者はプレイヤに打ち勝つ目標を与えられる一方で，（A）相手モデルが必要になる．池田らは後者のアプローチをとり，（C）不自然な着手をできるだけ減らすことにも配慮しながら，弱いプレイヤに対しても形勢を制御して負けてあげるような方法を提案した．そのために，MCTS の結果である各手の“勝率”と，教師あり学習の結果である各手の“選択確率”を参考にしている．詳細は３章で述べる．

また仲道と伊藤はコンピュータ将棋における棋力の調整方法を提案している[6][7]. コンピュータ将棋の評価関数を調整し，評価値が互角に近ければ近いほど好ましいとなるように変換を行い，形勢を互角に誘導することに成功している．

どちらの研究でも，不自然な手加減が課題になり，それを解決する試みが行われている．池田らは「形が悪い手」，「流れにそぐわない手」，「明らかに損をする手」，「高度すぎる手」などが不自然さにつながると指摘し，このうちいくつかを定量化して手加減手法に組み込んだ．仲道と伊藤はさらに「強さが一貫性していない」,「意図性を感じない」ような手が不自然であるという発展を行っている．

池田らはさらに，MCTS のシミュレーション（プレイアウト）部分の結果に重みを付け，地合／中央志向，悲観／楽観などの多様な戦略が演出できることを示した. しかし，昨今のプログラムは終局までシミュレーションを行わないため，同じ手法を直接用いることはできない．

本稿では，池田らの“自然な手加減”の手法を，深層学習コンピュータ囲碁に実装する．手法がそのまま使えるのかを確認し，新しい課題が発見されればそれを解決するアイデアを出す．

3. 実験手法

池田ら[5]の手法では，主に MCTS の結果である**勝率**と，教師あり学習で得られた静的な着手**選択確率**を用いて着手を以下のように決めている．Elf OpenGo や Leela Zero を用いた場合にもこれにあたる値は利用可能なため，それらを用いて手加減着手を実装した．

1、 まず MCTS を行い，着手を勝率順にソートする．この際，一部の有望な手のみに探索が集中しすぎないように，MCTS の C 値を大きめにする．勝つためには良い手だけ探索すればよいが，手加減の場合は 5 位や 10 位の手もある程度詳細に調べる必要がある．

2、 1 位の手の勝率と 2 位の勝率差が T_{uniq} 以上の場合，“絶対の一手”があると判断し，それを着手する．そうでなければ，明らかに悪い手を打ってしまうことになる．

3、 1 位の手の勝率が T_{min} 未満の場合，劣勢である．無抵抗の負けを防ぐため，1 位の手を着手する．

4、 1 位の手の勝率が T_{min} 以上 T_{max} 未満の場合，適度な形勢である. 1 位の手との勝率差が T_{dif} 以下の手の中から最も選択確率が高い手を着手する．すなわち，“悪すぎない最も自然に見えそうな手”を打つ．

5、 1 位の手の勝率が T_{max} 以上の場合，優勢になってしまっており，手加減が必要である．この場合，“勝率差が大きすぎず同時に選択確率が小さすぎない手”の中で，勝率を下げるために最も勝率の悪い手を着手する．そういう手が存在しなければ 1 位の手を着手する．具体的には以下の手の中から最も悪い手を選ぶ．

- 勝率差 3%以下かつ選択確率 5%以上．
- 勝率差 4%以下かつ選択確率 10%以上．
- 勝率差 6%以下かつ選択確率 20%以上．
- 勝率差 8%以下かつ選択確率 40%以上．

ⓒ 2019 Information Processing Society of Japan

表 1：探索された手の勝率と選択確率の例

順位	着手	勝率	選択確率
1	A	62%	16%
2	B	59%	20%
3	C	57%	19%
4	D	56%	29%
5	E	48%	16%

表 1 に例を挙げる．1 位の手 A と 2 位の手 B の勝率差が 3%，ここでもし T_{uniq} が 3%以下の場合，1 位の手と 2 位の手の勝率差が T_{uniq} 以上になったので，1 位の手が着手される．もし T_{min} が 65%の場合も，劣勢と判断し 1 位の手が着手される．もし T_{uniq} が 3%以上、T_{min} が 35%，T_{max} が 65%かつ T_{dif} が 5%とすると，適度な勝率の範囲であるので，着手 A と勝率差が 5%以下の着手 A, B, C の内に選択確率が一番大きい B が着手される．もし T_{uniq} が 3%以上かつ T_{max} が 60%とすると，高勝率の場合に当たるので，高勝率の場合の条件を満たす着手 B と D の内の勝率が低い D が着手される．

4. 実験

4.1 用いるプログラム

実験には三つのコンピュータ囲碁プログラムを用いる．新世代のものとして ELF OpenGo，Leela Zero を用い，相手役として利用が容易で適度な強さを持つ Ray[8]を用いることにする．ELF OpenGo は 19 路盤での実験に使い，2018 年 7 月に公開された重みファイルを用いる．Leela Zero を 13 路盤での実験に使い, 13 路盤の公式重みファイルはない一方で自己学習機能が利用可能なため，1 台の GPU マシンを用いて 4 週間訓練したものを用いた．簡単のため，今後は Elf, Leela と省略することがある．

Elf では, tree_search_base.h の addAction 関数を変更した．getQSA 関数を探索後の勝率を取得するため, mcts.h ファイルの evaluate 関数を選択確率を取得するために用いる．Leela では, UCTSearch.cpp の sort_children 関数を変更した．勝率取得は UCTNodePointer.cpp の get_eval 関数を用い，選択確率取得には UCTNode.cpp に get_static_policy 関数を追加し，Network::get_output 関数から取得を行った．

手加減方法のパラメータ設定には，まずは既存手法と同様に，T_{uniq}=0.08c，T_{dif}=0.03 c，T_{min}=0.35，T_{max}=0.55 とする．条件 5 の優勢時の場合の勝率差条件をそれぞれ 0.03c, 0.04c, 0.06c, 0.08c とし, それぞれ下限選択確率は 5%, 10%, 20%, 40%とした．ここの c は手加減の程度を一括で定めるパラメータであり，軽度の手加減では c=0.8，中度では c=1.5，重度では c=2.5 とした．

4.2 勝率制御に関する実験

我々の目的は“自然な手加減”である．自然さの確認の前に，非常に強いプログラムがちゃんと負けてあげることができているのかを調べる．まず 13 路盤の互先で，Leela と Ray を用いて 4 つの実験を行った．手加減の度合いを前節の c パラメータで調整し，(a) 手加減なし，(b) 軽度手加減，(c) 中度手加減，(d) 重度手加減それぞれを 30 戦した，Leela は各着手 5 秒探索，Ray は各着手 15 秒の探索とした．結果を表 2 にまとめる．

表 2：Ray 対 Leela．手加減の度合いが違う

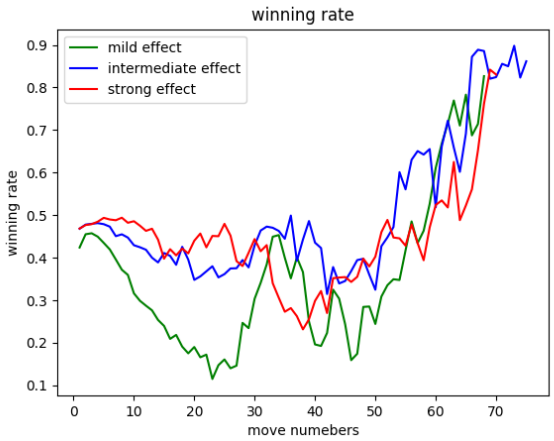
Leela の手加減	Ray から見た勝敗	Ray の勝率
(a) 手加減なし	0-30	0.00%
(b) 軽度手加減	7-23	23.33%
(c) 中度手加減	12-18	40.00%
(d) 重度手加減	22-8	73.33%

手加減をしない場合，Leela の 30 戦全勝となり，我々が少ないリソースで学習した重みファイルでも十分に Ray よりも強いことが分かった．

そして (b), (c), (d) を見ると，手加減程度を強めるに従い，Ray の勝率が上がっていくことが分かった．特に重度手加減の場合は“勝率だけ見れば”既存手法そのままで十分に Ray に対して手加減が出来ていると言える．

次いで，図 1 には，(b) から (d) の実験それぞれについて，Ray が勝った対局のみについて，「Leela が判断した，Ray 側の勝率」の平均値の推移を示す．

図 1：平均勝率変化



対局数が多くはないので一般的なことは言いにくいかもしれないが，軽度手加減の場合，まず一旦 Ray 側の勝率がかなり落ちていることが目につく．これは Ray が悪い手を打っているということと同時に，Leela の勝率判定が（ある意味）極端だからとも言える．囲碁は厳密にはどの局面も

ⓒ 2019 Information Processing Society of Japan

勝ちか負けか引分であり，プログラムの強さが向上すれば相手プレイヤが初手を打った時点で勝率 90%になってもおかしくはない．言いかえれば，ここでは勝率が 20%程度に下がってしまっているものの，普通の人間プレイヤにとっては“すごく不利だ，つらい”と思うほどの形勢ではない可能性はおおいにある．

一方で中度手加減や高度手加減の場合は中盤以降まで（強いプログラムから見ても）形勢を互角付近に維持できていることが多いと予想できる．これは好ましいことではあるが，それが不自然な手による維持であれば意味がないため，また別の分析が必要ではある．

図 2：同じ盤面の Ray（横軸）と Leela（縦軸）の予測勝率

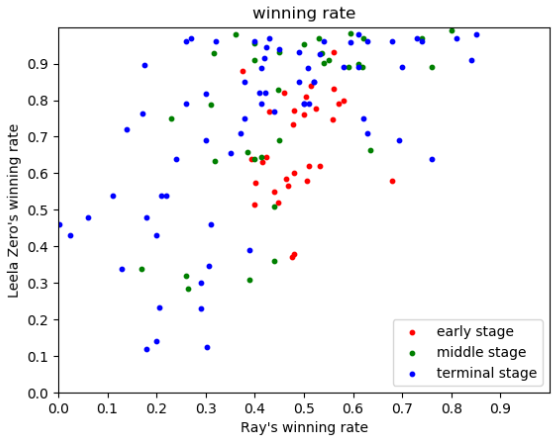


図 2 はランダムに選んだ 60 局について，序盤・中盤・終盤の局面を 2 つのプログラムに与え，探索の後に勝率を推定したものである．序盤中盤について Ray は概ね 0.4〜0.6 と判定しているのに比べ Leela は 0.9 以上と判定することもあることが分かる．これは Leela の勝率推定精度が高いという見方もできるかもしれないが，少し前のプログラムにおいて勝率差等のパラメータを調整した既存手法が，近代的なプログラムでは適切でない可能性を示している．

4.3 着手の自然さに関する課題

前節までは勝率に着目したが，本節からは不自然な着手について観察してみる．本来ならばいくつかの手法を比較するために被験者実験を行うことが望ましいが，その前段階として著者らによる少数の対局の観察を行うことにした．

前節までの Leela と Ray による 13 路盤棋譜に加え，本節では 19 路盤で Elf を用い，人間プレイヤと対戦をさせた．Elf は各着手 10 秒探索，人間側は特に時間制限を設けない．人間側の思考中に探索を進める pondering は用いていない．

総譜で少し見にくいが，付録図 8 に良い対局の一例を示す．黒番が中度手加減の Elf，白石はアマチュア 3 段程度のプレイヤである．このプレイヤからは，「特に不自然な着手は感じなかった」「漫然と互角の形勢を保つわけではなく，自分が悪い手を打つとそれには（適度に）咎める手を打ってくれた（例えば 155 手目など）」という肯定的な意見が得られた．高段者レベルから見れば不自然に弱い手はあるかもしれないが，当該プレイヤに感じさせないことが重要であるのでその意味では十分良い結果であると言える．なお最後は黒番（コンピュータ側）の 1 目半勝ちとなっている．ヨセの段階に入ると，1 目半勝ち＝勝率 98%，半目負け＝勝率 2%などと勝率が 2 分化し，「勝率差がありすぎる手加減はしない」というポリシーから，これを負けに持っていくような手は打てないという課題が見つかった．

一方で少し極端な対局も行ってみた．付録図 9 は同様に黒が手加減 Elf，白石はアマチュア 3 段程度のプレイヤの対局（途中まで）である．このとき，手加減側は，着手の選択確率（一見した良さ）を考慮せず，機械的に「最も勝率が 50%に近い」手を打つことにした．この結果，選択確率の低そうな，“普通に見えない手”というのが散見された．例えば，41 の後の 43 手抜きは一貫性がなく，81〜85 なども一貫性を感じず，89 以降 101 まで何手も手抜きしたことも目立ったという感想である．

以降は，他の対局で見られた，具体的かつ典型的な課題について紹介する．

(1) シチョウ

シチョウはアタリの連続で斜め方向に石を追いかけ最後には取るような手筋のことである．追いかけ先に相手の石があるとシチョウが成立しないこともあり，読みが必要である．単純なシチョウでも 40 手前後の読みを必要とするが，追い方はほぼ一通りしかないため，級位者でも読み切れることの多い手筋である．一方シンプルな囲碁プログラムでは 40 手の読みが難しいこともあり，苦手の一つとされていた．深層学習以前の強豪プログラムの多くはシチョウの専用ルーチンを作成して判定を行っていたが，標準の Elf ではそのような工夫はなされていないようである．

図 3 に問題のある着手を示す．黒が G14（×）とマガり，白 3 子を E13 からシチョウで取るぞと脅した場面である．ここで白は A や B に着手すればシチョウを防いで自然であるが，実戦では Elf は E18（1）に着手してしまった．人間プレイヤは E13 に着手してシチョウを成立させ，数手後に Elf の投了となった．

白が 1 に着手してしまったのは手加減手法が原因なのか Elf そのものの問題なのかをはっきりさせるため，探索時間を 40 秒に延ばした．表 3 はそのときの各着手の勝率上位と探索回数である．A や B の着手は含まれるが，C，D，E などシチョウを防がない手をより高勝率と判定してしまっている．従って，この問題は手加減云々の問題ではなく，Elf が（家庭にあるような PC と待てる程度の探索時間では）シチョウを苦手にするということであると判断する．

ⓒ 2019 Information Processing Society of Japan

図 3：不自然な着手（白 1）シチョウ

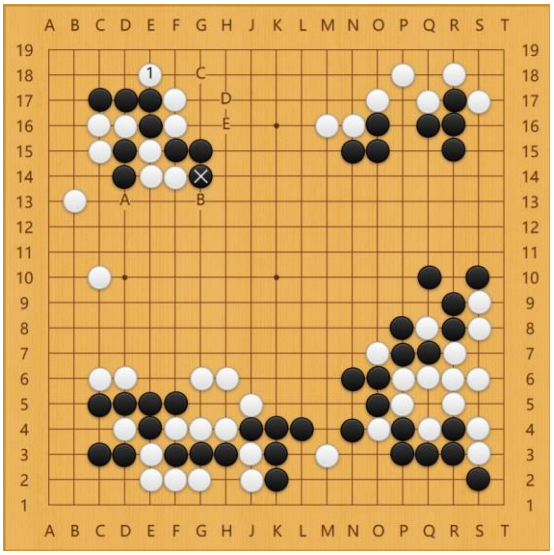


表 3：図 3（白番）持ち時間を 40 秒に上げた探索結果一部

順位	座標	勝率	探索回数	選択確率
1	H16(E)	82.15%	6680	7.55%
2	H17(D)	81.33%	4348	8.96%
3	G13(B)	79.78%	2730	12.16%
4	G18(C)	78.33%	961	6.00%
10	D13(A)	75.32%	1428	15.46%
14	E18(1)	69.25%	1809	35.11%

(2) 局所的な応対をしない（手抜き）

囲碁では，自分の石に対して相手が攻撃や侵略や様子見をしてきた場合には直接対応するのが自然に見えることが多い．これは特に初級者には顕著な傾向であり，上級になるほど相手の言いなりにならず（手についてゆかず），本当に大事な箇所に打てるようになる．

従って，「対応が必要な部分で対応しない」ことは悪いだけでなく不自然さの要因になる．一方「対応が不要な（小さい）部分で対応しない」ことは，本来は最善であったとしても，初級者中級者を相手にする場合には不自然に映るかもしれないことに注意しなければならない．

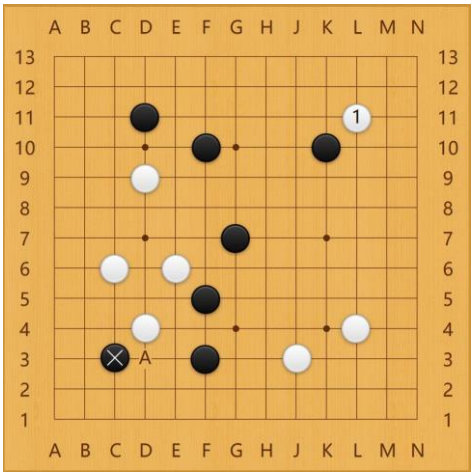
図 4 は後者の一例である．黒が C3（×）に着手したところであり，これは白地・白石への働きかけの強い手である．この場合，中級者くらいまでは，D3（A）などで対応をすることが自然に見えるだろう．しかし，本研究の手加減プログラムは D3 ではなく L11（1）に着手した．これは決して悪い手ではないが，手加減時に適切な手とは言えないかもしれない．

表 4 は，この局面における白番の探索結果のうち C3 と L11 のみを示したものである．勝率はほぼ同じであり，その意味ではどちらも不自然な手とは言えない．現在 Leela から見た白番の勝率は優勢レベルにあり，少しでも勝率を落とそうとして L11 が選ばれたのである．しかし，選択確率を見ると D3 が 3 倍近く「もっともらしい手」であり，Leela から見ても D3 のほうが自然な手であることが分かる．既存の手加減手法では，このように“たかだか 0.7%勝率を落とすために，3 倍近く自然な手があっても打たない”ということが行われてしまっていることがわかった．これは分かりやすい課題であり，論文後半で一つのアイデアを提示する．

表 4：図 4（白番）の探索リストの一部

順位	座標	勝率	探索回数	選択確率
1	D3(A)	69.75%	1295	40.00%
2	L11(1)	69.03%	255	13.89%

図 4：不自然な着手（白 1）手抜き



(3) 序盤

旧来の MCTS 手法では，序盤に 80%であるとか 20%といった極端な勝率になることは滅多にない．これは終局までのランダムシミュレーションによって，どんなに有利な局面不利な局面であっても運不運によって最終的な勝率が 50%に近づいてしまうためである．新世代のプログラムでは終局までの探索は行わず状態評価関数を用いるため，序盤であってもどちらかが有利になればかなり劇的な評価値を与えることが多い（図 2）．

この事実は，序盤でも手加減プログラムが優勢であると判断して，手加減を頻繁に行おうとすることを意味する．図 5 はその例である．黒が自然な流れで R9（×）着手時，表 5 を見るとすでに白の勝率は 58%と計算されている．そして，自然な O17（A）ではなく，勝率を互角に近づけることができる Q4 が選ばれているのである．Q4 ツケは決してない手ではない（実際，選択確率も 16.8%で低いとは言えない）が，この時点で初級者中級者に対して打つのが適切な手とは言い難い．これ以外にも序盤に不自然な手加減をしてしまうケースは多く見受けられた．序盤は見慣れた盤面が多く不自然な手も目立ちやすいため，序盤はパラメ

ⓒ 2019 Information Processing Society of Japan

ータを変えるとか何かの工夫は必要かもしれない.

図 5：序盤の不自然な着手（白 1）

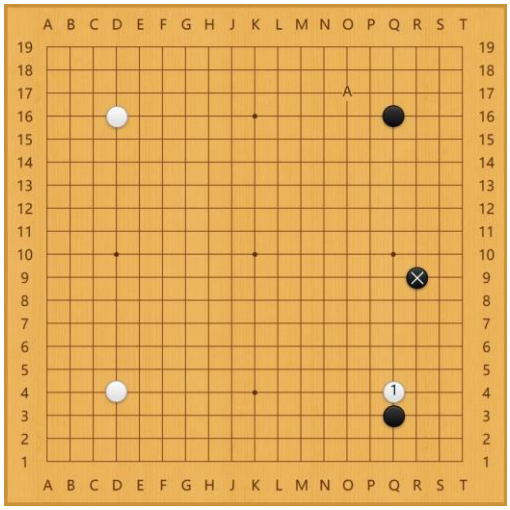


表 5：序盤の例の探索リスト

順位	座標	勝率	探索回数	選択確率
1	O17(A)	58.06%	3770	71.59%
2	Q5	57.54%	285	5.69%
3	Q4(1)	52.95%	502	16.756%

(4) 終盤

微差の終盤になると，通常は少し勝つ手が高い勝率を，少し負ける手が低い勝率を持つようになる．これが完全に二極化すれば，今のシステムであれば良くも悪くももう勝敗は逆転しない(4.3 節 3 段落後半の例など)．しかし稀に，好ましくない逆転が起きる場合がある．

図 6 は人間との対局の最終盤であり，黒が少し勝つ局面で，白が L18（×）に打った局面である．ここで黒は C10 の小ヨセや，O9 や N6 に傷を守るなどが自然な手である．表 6 の探索リストを見てもそのような手が上位に来ている．しかし実際には黒は 65 位の手である Q7 を着手してしまった．これは自殺手の一種であり，白 O9 キリ，黒ツギ，白 N6 キリでどちらかの黒石が取られてしまう．これでは興ざめである．

表 6 より Q7 は勝率 93.26%と判定されているが，これは実際には負け（5%など）と判定してほしいのに過大評価が起きていることになる．多少の読みは必要な部分なので，164 回しか訪問していないのではこの推定エラーは仕方ないのかもしれない．本来「勝つため」の探索であれば，93%であろうが 5%であろうが 65 位の手など考慮もされないわけで，これは手加減特有の課題であると言える．対策のアイデアとしては，一旦探索して「手加減するならこの手」と決めたものに対し，再度確認の探索を行うということはありうるかもしれない．この場合であれば，Q7 に打つことを考えたあと，164 回の探索に例えば 1000 回を上積みして再確認をするということである．

図 6：終盤の不自然な着手（黒 1）

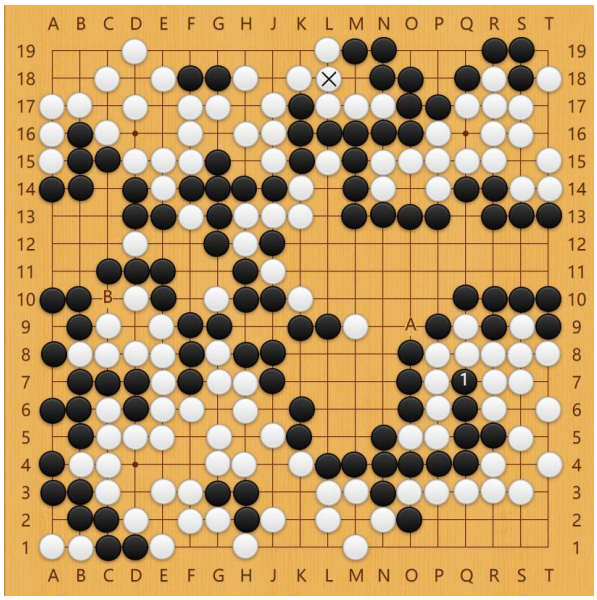


表 6：終盤の探索リスト，一部

順位	座標	勝率	探索回数	選択確率
1	O9(A)	99.34%	566	11.63%
2	N6	99.33%	277	5.67%
3	C10(B)	99.16%	1407	30.59%
65	Q7(1)	93.26%	164	6.28%

5. 新しい試み

実験のデータを分析すると，勝率が T_{max} 以上の場合つまり優勢局面で不自然な着手が出る例が多く見られる．これは採用したプログラムが強く，また序盤から高めの勝率を出力することが原因である．優勢局面の着手決定法は述べてきたようにまだ課題が多く，本章では幾つかの試みを紹介することにする．

5.1 前の一手との距離の評価関数

4.3 節（2）あるいは図 4 で見たように，局所的な対応をしないことは，特に初級者中級者にとっては不自然に映ることが多い．そこで，優勢局面において，勝率を落とすことよりも自然さを重視し，またプログラムが持つ本来の選択確率を「前の着手からの距離」で補正する試みを紹介する．このような距離による確率補正自体は，旧世代の MCTS プログラムでも使われていることが多い．

3 章手順 5 においては，“勝率差が大きすぎず選択確率も低すぎない”手の中で，最も勝率の悪い手が着手された．本節では，同候補の中で，(1-d/D) ×選択確率が最大のものを着手することにする．D は盤上の最大距離，d は前の

着手と候補着手の距離である．特に勝率低下は考えずに，初級者中級者に自然に見える手を打とうということである．

また 4.2 節と同じ実験設定で，重度手加減 Leela と Ray に 30 対局をさせた結果は表 7（上側）である．

表 7：自然さを重視した場合の勝率

手法	Ray の勝敗	勝率
重度手加減＋提案手法	6-24	20.00%
重度手加減	22-8	73.33%

通常の重度手加減をした場合には Ray に 7 割以上勝たせることができていたが，この手法では 2 割しか勝たせることができなかった．勝率を下げることを明示的に行えていないので，ある意味当然の結果である．

図 7：提案手法の例（白 1）

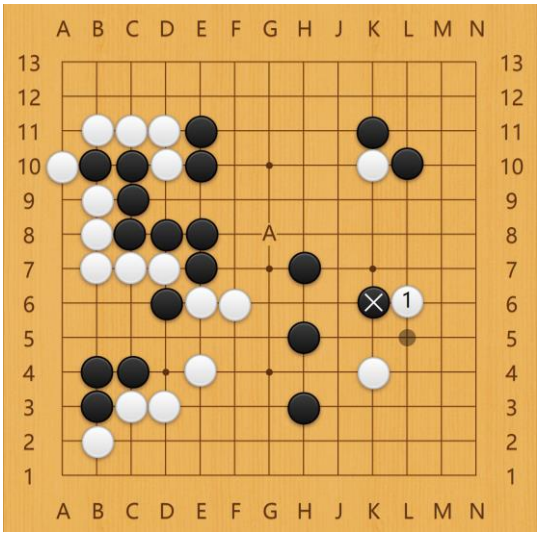


表 8：図 7（白番）の探索リストの一部

順位	座標	勝率	探索回数	選択確率	補正後
1	L11	91.43%	468	5.16%	0.04
10	L6(1)	88.12%	165	17.35%	0.16
11	G8(A)	87.57%	69	9.32%	0.07

一方で，着手の自然さはたしかに改善しているように思われた．既存手法で「ああ，こんなところを手抜きされてしまった，わざとっぽいな」と思えるような場所で新手法を用いたところ，手抜きはされずに付き合ってもらえることが多くなった．

図 7 はこの提案手法を用いた例である．黒が K6 にカカって隅の白石を圧迫した場面である．表 8 がその探索リストの一部であり，従来手法であれば G8 が選択される．これは比較的難しい手であるうえ，右下の白を助けていない手抜きの手である．提案手法では，（悪過ぎない中で）補正後の選択確率が最も高くなる L6（1）が選ばれる．これは直前の黒の手に対応する自然な手である．この提案手法はやや極端に自然さを求めているので，従来法との丁度よい組み合わせで自然な手加減が実現できるかもしれない．

5.2 勝率区間の変化

これまで述べたように優勢局面での着手選択法が不自然さを招く傾向があることが分かっている．その選択法を変えることの他に，優勢局面の定義自体も考慮の余地がある．新しいプログラムは少しの有利でも勝率をかなり高く見積もる傾向がある（図 5，表 5 など）ため，すぐに優勢局面であるとみなして強い手加減を行ってしまう．本節では，微妙ではあるが手加減方法の勝率区間を決めた T_{min} を 0.35 から 0.4 へ，T_{max} を 0.55 から 0.6 に変更させてみた．例えば図 5 の局面は「適度な勝率の範囲内」と判定されるため，自然な着手すなわち O17 が着手されるようになる．

表 9：定義域をずらした場合の勝率

Ray	勝敗	勝率
軽度手加減＋提案手法	12-18	40.00%
中度手加減＋提案手法	15-15	50.00%
重度手加減＋提案手法	16-14	53.33%

このパラメータで 4.2 節と同様の実験を行った結果が表 9 である．表 2 と比べると全体に軽度～重度の差が小さくなっており，これはあまり良い傾向とは言えない．今後，パラメータの調整などを行い，序盤での自然さと，最終的にうまく負けることを両立していきたい．

6. 未来の展望

6.1 動的な手加減と静的な手加減

既存手法およびここまで述べた手法はすべて，形勢に合わせてそれを互角に近づけようとする動的な手法である．従って，「自分が良い手を打つほどコンピュータも強い手を打って勝ちにくく」「自分が悪い手を打つほどコンピュータも悪い手を打って勝ちやすく」なるという皮肉な結果を導くことがある．これは，良い手を打とうとすることへの負のインセンティブになりかねない．良い手を打てば勝ち，悪い手をある程度以上打てば負けるというほうが自然であろう．そのためには，コンピュータの強さを（概ね）固定して，ただしそれを何段階にも分けるということが自然である．現在の軽度～重度手加減でも似たような調整は可能だが，より明示的に，動的でない手加減が行える手法も開発していきたい．

ⓒ 2019 Information Processing Society of Japan

6.2 多様な戦略の演出

池田らは，MCTS のプレイアウト結果を単なる勝ち負けで集計せずに，「隅や辺に地が多ければボーナス（地合い派）」「10 目くらい過小評価（悲観派）」「大勝や大敗ならボーナス（好戦派）」などと補正して集計することで，多様な戦略が演出できることを示した[5].

しかし，この手法はただちに現代的な深層学習コンピュータ囲碁プログラムに適用することはできない. なぜなら，AlphaGo Zero をはじめ Elf，Leela などではリーフノードの評価値を“終局までのシミュレーション”ではなくニューラルネットワークの value で求めるからである.

これを解決する方法は概ね 2 通りある．一つは，そもそもニューラルネットワークを学習するときに，勝ったか負けたかを学習データとせずに，「どのような勝ち（負け）になったか」を学習データとすることである．value の出力を 1 つではなく複数にし，使い分けることができればそのようなことも可能かもしれない．もう一つは，探索中にリーフノードの値を補正することであるが，これは従来法ほど単純には行かないと思われる.

6.3 局面成熟度による手加減の調整

4.3 節（3）（4）で述べたように，序盤や終盤では特有の課題が出てきやすい．序盤ではいきなり優勢と意識して自然でない手で手加減しようとする．これは序盤は少し選択確率を重視するなどが必要だろう．終盤では，1.5 目勝ちの局面から 1.5 目負けの局面に持っていくことができない（勝率が二極化し，差が大きくなるため）ことも課題であるし，逆に自殺手のようなことも課題である．これにもいくつかの対応がありえるが，いずれにしても「今は序盤なのか終盤なのか」を判断することが必要になる．もっと言えば,「今は落ち着いた局面なのか，どちらが取られるかのような局面なのか」の判断も将来的には必要であろう.

7. おわりに

本稿ではモンテカルロ囲碁プログラムが初級者中級者に手加減する既存手法について，深層学習を用いた新世代のプログラムでも有効なのか，新しい課題があるのかを検証することを目的とした．そして，手加減そのものはそれなりに有効に働き，不自然な手も思ったよりは多くないものの, 非常に都合の悪いこともしばしば起こることを確認し，その理由と対応案を考察した.

謝辞 本研究は JSPS 科研費（基盤研究 C）17K00506 の助成を受けたものである.

参考文献

[1] Silver. D., Huang. A., Maddison. C., et al: Mastering the game of Go with deep neural networks and tree search, Nature, Vol.529, No.7587, pp.484–489 (2016).
[2] Silver. D., Schrittwieser. J., Simonyan. K., et al: Mastering the game of go without human knowledge, Nature, Vol.550, No.7676, pp.354 (2017).
[3] https://github.com/leela-zero/leela-zero アクセス時間 2019.02.10
[4] Tian, Y., Ma, J., Gong, Q., Sengupta, S., Chen, Z., & Zitnick, C. L. (2018). Elf opengo.
[5] 池田 心, Viennot Simon, モンテカルロ碁における多様な戦略の演出と形勢の制御: 接待碁 AI に向けて，情報処理学会，ゲームプログラミングワークショップ 2012 論文集, Vol.2012, No.6, pp. 47-54, 2012-11-09.
[6] 仲道隆史, 伊藤毅志 : 人を楽しませる接待将棋システム、2014 年度人工知能学会全国大会,1E5-OS-23b-5in,(2014).
[7] 仲道隆史，伊藤毅志：将棋 AI における棋力の調整が不自然さを与える影響、ゲームプログラミングワークショップ 2014, P-16 (2014).
[8] http://computer-go-ray.com アクセス時間 2019.02.10

付録

図 8：良い手加減の一局（黒がコンピュータ）

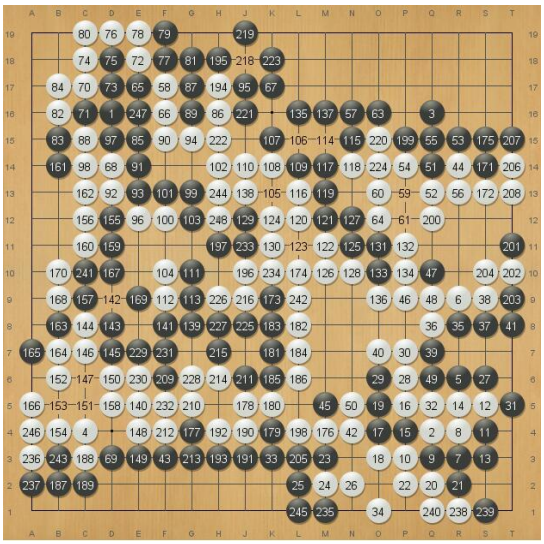
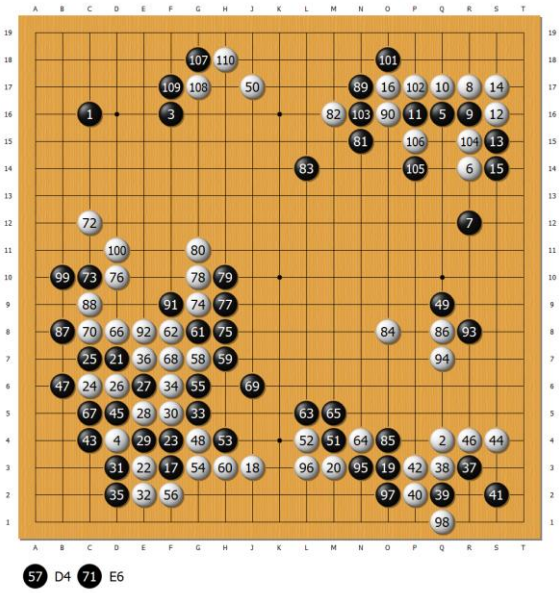


図 9：単なる手加減をした例（黒がコンピュータ）



ⓒ 2019 Information Processing Society of Japan