第 35 回世界コンピュータ将棋選手権
dlshogi with HEROZ アピール文章

山岡忠夫
加納邦彦
大森悠平
2025/3/20

1　dlshogi のアピールポイント

1.1　モデル構造

ResNet と Transformer を組み合わせた構造を採用する。

Transformer は、局面全体を見たグローバルな特徴を捉えることができる。しかし、局所的な特徴を学習するには比較的多くのデータ量が必要になり、ResNet と比べて学習の効率が悪い。

一方、ResNet は、構造上、入力に近いブロックで局所的な特徴を捉えやすい帰納バイアスを持っているため、局所的な特徴を学習する効率が良い。

そこで、Transformer を ResNet の後ろに接続することで、ResNet が効率的に捉えた局所的な特徴量を使って、Transformer で効率良くグローバルな特徴を捉えることができる。

このような ResNet と Transformer を組み合わせた構造は、画像認識の分野では、「Bottleneck Transformers for Visual Recognition[1]」などで提案されている。

1.2　ラージカーネル

ResNet は、複数の層を重ねることで、グローバルな特徴を捉えることができる。入力に近い層では局所的な特徴を捉えて、後続の層で、徐々にグローバルな特徴を捉えられるようになる。

以前より、カーネルサイズを大きくすることで、少ない層で、グローバルな特徴を捉えられることが知られている。しかし、現在の GPU は 3x3 のカーネルサイズに最適化されており、計算速度の観点から大きなカーネルサイズは採用されていない。

「A ConvNet for the 2020s[2]」では、カーネルサイズを大きくすることで、最先端の Vision Transformer と同等の性能が出せることが報告されている。

「InceptionNeXt[3]」では、大きなカーネルを、縦と横に分割することで、計算コストを抑えて、同等の性能が出せることが報告されている。

第 33 回世界コンピュータ将棋選手権の、Ryfamate が採用した「Ryfamate Cross Network[4]」は、9x9 のカーネルを 9x1 と 1x9 のカーネルに分割する構造を採用している。将棋の盤面が

[1] https://arxiv.org/abs/2101.11605
[2] https://arxiv.org/abs/2201.03545
[3] https://arxiv.org/abs/2303.16900
[4] https://www.apply.computer-shogi.org/wcsc33/appeal/Ryfamate/appeal_ryfamate_20230421.pdf

9x9 固定であるため、ストライドが不要になり、大幅に計算コストを下げられる利点がある。

実験[5]したところ、Ryfamate Cross Network は、7x7 のカーネルよりも、効率的に高い精度が出せることが分かった。しかし、ResNet から置き換える層を増やし過ぎると、逆に精度が下がる現象も確認できた。これは、9x1 と 1x9 のカーネルの出力がそれぞれ、1x9 と 9x1 になるため、元のサイズに戻すためにブロードキャストを行う過程で、位置情報が失われているためと考えた。そこで、1x1 のカーネルを並列で加えて実験したところ、層を増やしても精度が劣化しないことが確認できた[6]。

以上の結果を元に、局所的な特徴を捉える特性を残して、グローバルな特徴も位置情報を残しながら伝えられるように、通常の ResNet ブロックを 5 ブロック間隔で 9x1、1x9、1x1 のカーネルに置き換える構造を採用する。

1.3 入玉特徴量

コンピュータ将棋の大会で採用されている 27 点法では、先手と後手で宣言勝ちの条件が異なるため、モデルに手番の情報を入力しない限り、正しく局面を評価できない。

しかし、ほとんどの局面では、手番を対称に捉えても問題ないため、後手の場合、反転した局面を学習して、手番を無視して評価する方が効率が良い。

最近の大会では、入玉宣言で勝敗が決まる対局が増える傾向にあり、入玉の精度が勝敗に影響してきている。

そこで、より正確に入玉の状況を評価できるように、入力特徴量を追加することを検討した。

前述の通り、手番そのものを入力特徴量に加えるのは、入玉に無関係の局面での対称性が失われるため、あとどれくらいで入玉宣言勝ちできるかを逆算した特徴量として与えることにする。これにより、先手、後手の違いを吸収し、対称性を維持できる。

具体的には、以下の特徴量を追加した。

- 入玉しているか
- 敵陣の玉を除く枚数(10 枚までの残り枚数。ワンホットで与える。)
- 残り点数(先手は 28 点、後手は 27 点までの残り点数。19 を上限として残り点数をワンホットで与える)

同じ訓練データを入玉特徴量の有無で学習して、強さを計測したところ、入玉特徴量を加えた方が強くなることが確認できた[7]。入玉宣言勝ちの回数は増えなかったため、入玉するかの見極めが上手くなったと考えている。

1.4 モデルサイズ

これまでの dlshogi では、モデルのパラメータ数を増やすほど、同一持ち時間でも強くな

[5] https://tadaoyamaoka.hatenablog.com/entry/2024/07/20/160442
[6] https://tadaoyamaoka.hatenablog.com/entry/2024/07/26/224144
[7] https://tadaoyamaoka.hatenablog.com/entry/2024/12/29/141617

ることが確認できている。

前大会の 30 ブロック 384 フィルタのモデルのパラメータをさらに増やして、40 ブロック 512 フィルタの約 2 億パラメータのモデルを学習した。

訓練データは同一ではないが、40 ブロック 512 フィルタのモデルが、elo レーティングで 37.9 だけ強くなった[8]。

また、50 ブロック 640 フィルタの約 4 億パラメータのモデルの学習も行ったが、floodgate の棋譜に対する方策、価値の精度はともに大きく向上したが、探索速度が 53%程度となり、同一持ち時間では弱くなることが確認できた（GPU に H100 PCIe を使用）。

現状の GPU では、50 ブロック 640 フィルタのモデルは推論速度が大きく低下するため、探索が浅くなり過ぎて、同一持ち時間で弱くなったと考える。なお、同一探索数では強くなる。

パラメータ数を増やすほど同一持ち時間でも強くなるというスケーリング則は、50 ブロック 640 フィルタあたりで、成り立たなくなることがわかった。

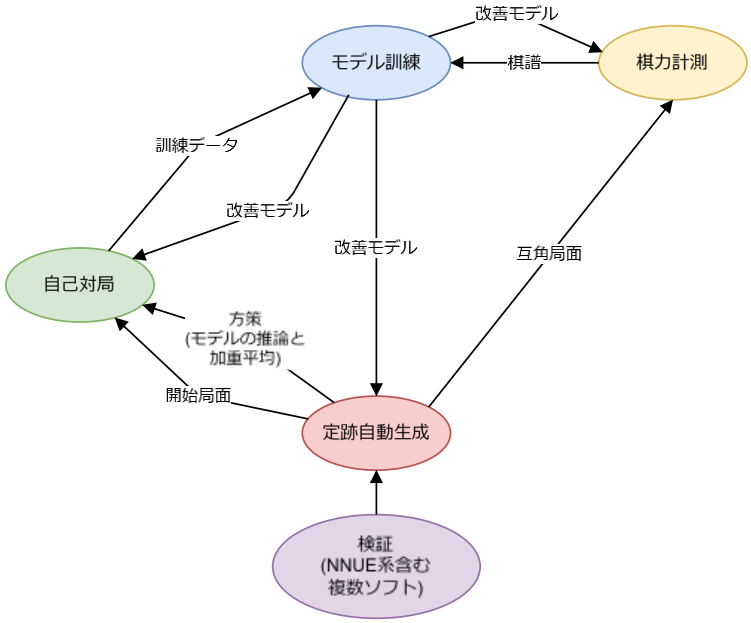
ただし、モデル精度は高いため、知識蒸留することで、活用することはできる。

1.5　定跡自動生成

前大会の定跡自動生成の手法[9]で、継続して定跡の自動生成を行った。

約 300 万局面が登録されている。

また、自動生成した定跡を、訓練データ生成、棋力測定にも活用している[10]。



[8] https://tadaoyamaoka.hatenablog.com/entry/2024/06/07/224348

[9] https://www.apply.computer-shogi.org/wcsc34/appeal/dlshogi_with_HEROZ/dlshogi_with_HEROZ_appeal_detail_wcsc34_v2.pdf

[10] https://tadaoyamaoka.hatenablog.com/entry/2024/12/22/194641

2　dlshogi の特徴

以下に、これまでの dlshogi の累積的な特徴を示す。

※下線部分は、第 34 回世界コンピュータ将棋選手権からの差分を示す。

- ディープラーニングを使用
- 指し手を予測する Policy Network
- 局面の勝率を予測する Value Network
- 入力特徴にドメイン知識を積極的に活用
- モンテカルロ木探索
- 未探索のノードの価値に親ノードの価値を使用
- GPU によるバッチ処理に適した並列化
- 自己対局による強化学習
- 詰み探索結果を報酬とした強化学習
- 既存プログラムを加えたリーグ戦による強化学習
- 既存将棋プログラムの自己対局データを混ぜて学習
- 既存将棋プログラムの自己対局データを使った事前学習
- ブートストラップ法による Value Network の学習
- 引き分けも含めた学習
- 指し手の確率分布を学習
- 同一局面を平均化して学習
- 評価値の補正
- SWA(Stochastic Weight Averaging)
- 末端ノードでの短手順の詰み探索
- ルートノードでの df-pn による長手順の詰み探索
- 勝敗が確定したノードのゲーム木への確実な伝播
- PV 上の局面に対する長手数の詰み探索
- 序盤局面の事前探索（定跡化）
- 定跡作成時に floodgate の棋譜の統計を利用した確率分布を方策に利用
- マルチ GPU 対応（NVIDIA A100×8 を使用予定）
- TensorRT を使用
- Optuna による探索パラメータの最適化
- 確率的な Ponder
- ノードのガベージコレクションとノード再利用処理の改良
- 飛車と角の利きのビット演算
- 2 値の入力特徴量を 1 ビットで転送することで推論のスループットを向上
- Stochastic Multi Ponder

3　使用ライブラリ

- Apery[11]（WCSC28）
  →局面管理、合法手生成のために使用

3.1　ライブラリの選定理由

本プログラムは、将棋におけるディープラーニングの適用を検証することを目的としており、学習局面生成、局面管理、合法手生成については、使用可能なオープンソースがあれば使用する方針である。そのため、学習局面を圧縮形式(hcpe)で生成する機能を備えていて、合法手生成を高速に行える Apery を選定した。

4　各特長の具体的な詳細（独自性のアピール）

4.1　ディープラーニングを使用

DNN(Deep Neural Network)と MCTS を使用して指し手を生成する。
従来の探索アルゴリズム(α β法)、評価関数(3 駒関係)は使用していない。

4.2　Policy Network

局面の遷移確率を Policy Network を使用して計算する。
Policy Network の構成には、Residual Network を使用した。
入力の畳み込み 1 層と、ResNet 36 ブロック(畳み込み 2 層で構成)と Transformer 2 ブロック(Transformer は Attention ブロックと FFN ブロックがあるため ResNet2 ブロック換算)で構成した。通常のカーネルサイズは 3x3（入力層の持ち駒のチャンネルのみ 1)、5 ブロック間隔で 9x1 と 1x9 と 1x1 を並列にしたカーネルに置き換える。フィルタ数は 512 とした。

4.3　Value Network

局面の勝率を Value Network を使用して計算する。
Value Network は、Policy Network と出力層以外同じ構成で、出力層に全結合層をつなげ、シグモイド関数で勝率を出力する。

4.4　入力特徴にドメイン知識を積極的に活用

Alpha Zero では、入力特徴に呼吸点のような囲碁の知識を用いずに盤面の石の配置と履歴局面のみを入力特徴とすることで、ドメイン知識なしでも人間を上回ることが示された。し

[11] https://github.com/HiraokaTakuya/apery

かし、その代償として、入力特徴にドメイン知識を活用した AlphaGo Lee/Master に比べて倍のネットワークの層数が必要になっている。AlphaGo Zero の論文の Figure 3 によると、ネットワーク層数が同一のバージョンでは Master を上回る前にレーティングが飽和している。

強い将棋ソフトを作るという目的であれば、積極的にドメイン知識を活用した方が計算リソースを省力化できると考えられる。

そのため、本ソフトでは、入力特徴に盤面の駒の配置の他に、利き数と王手がかかっているかという情報を加えている。それらの特徴量が学習時間を短縮する上で、有効であることは実験によって確かめている。

4.5　モンテカルロ木探索

対局時の指し手生成には、Policy Network と Value Network を活用したモンテカルロ木探索を使用する。

ノードを選択する方策に、Policy Network による遷移確率をボーナス項に使用した PUCT アルゴリズムを使用する。PUCT アルゴリズムは、AlphaZero の論文[12]の疑似コードに記述された式を使用した。

また、末端ノードでの価値の評価に、Value Network で計算した勝率を使用する。

通常のモンテカルロ木探索では、末端ノードから複数回終局までプレイアウトを行った結果（勝率）を報酬とするが、将棋でランダムなプレイアウトは有効ではないため、プレイアウトを行わず Value Network の値を使用する。

4.6　未探索のノードの価値に親ノードの価値を使用

モンテカルロ木探索の UCB の計算時に、未探索の子ノードがある場合、そのノードの価値に何らかの初期値を与える必要がある。子ノードの価値は親ノードの価値に近いだろうという将棋のドメイン知識を利用し、それまでの探索で見積もった親のノードの価値を動的に初期値として使用する。ただし、ノードの訪問回数が増えるに従い、その価値の減衰を行い、幅より深さを優先した探索を行う(FPU reduction)。

4.7　GPU によるバッチ処理に適した並列化

複数回のシミュレーションを順番に実行した後、それぞれのシミュレーションの末端ノードの評価をまとめて GPU でバッチ処理する。その後、評価結果をそれぞれのシミュレーションが辿ったノードにバックアップする。以上を一つのスレッドで行うことで、マルチスレッドによる実装で課題となる GPU の計算後にスレッドが再開する際にリソース競合が起き

[12] http://science.sciencemag.org/content/362/6419/1140

る問題（大群の問題）を回避する。

GPU で計算中は、CPU が空くため、同じ処理を行うスレッドをもう一つ並列で実行する。2 つのスレッドが相互に CPU と GPU を利用するため、利用効率が高い処理が可能となる。

4.8　自己対局による強化学習

事前学習を行ったモデルから開始して、AlphaZero[13]と同様の方式で強化学習を行う。自己対局により教師局面を生成し、その教師局面を学習したモデルで、再び教師局面を生成するというサイクルを繰り返すことでモデルを成長させる。

2018 年の大会で使用した elmo で生成した教師局面で収束するまで学習したモデルに比べて、自己対局による強化学習によって有意に強くすることができた。

4.9　詰み探索結果を報酬とした強化学習

自己対局時に終局まで対局を行うと、モンテカルロ木探索の特性上、詰むまでの手順が長くなる傾向がある。勝率予測に一定の閾値を設けることで、終局する前に勝敗を判定することで対局時間を短縮できるが、モデルの精度が低いうちは誤差が大きいため、学習精度に影響する。

この課題の対策として、df-pn による高速な長手数の詰み探索の結果を報酬とした。単純にすべての局面で詰み探索を行うと、自己対局の実行速度が大幅に落ちてしまう。自己対局は複数エージェントに並列で対局を行わせ、各エージェントからの詰み探索の要求をキューに溜めて、詰み探索専用スレッドで処理するようにした。エージェントが GPU の計算待ちの間に詰み探索が完了する。エージェントが探索している局面は別々のため、時間のかかる詰み探索の要求が集中することは少ない。これにより自己対局の速度を大幅に落とすことなく長手数の詰み探索を行えるようになった。

4.10 既存プログラムを加えたリーグ戦による強化学習

自分自身のプログラムのみで強化学習を行うと戦略に弱点が生まれる可能性がある。弱点をふさぐには多様なプログラムによるリーグ戦が有効だが、複数のエージェントを学習するにはエージェント数の分だけ余分に計算資源が必要になる。

計算資源を省力化して、リーグ戦の効果を得るために、オープンソースで公開されている既存の将棋プログラムを 1/8 の割合でリーグに加えて強化学習を行うようにした。

4.11 既存将棋プログラムの自己対局データを使った事前学習

本プログラムを使用して、Alpha Zero と同様に、ランダムに初期化されたモデルから強化

[13] https://arxiv.org/abs/1712.01815

学習を行うことも可能だが、使用可能なマシンリソースが足りないため、スクラッチからの学習は行わず、既存将棋プログラムの自己対局データを教師データとして、教師あり学習でモデルの事前学習を行う。

教師データには、elmo で生成した自己対局データを使用した。

4.12 既存将棋プログラムの自己対局データを混ぜて学習

以前の dlshogi は、入玉宣言勝ちできる局面でなかなか入玉宣言勝ちを目指さないという課題があった。

自己対局では入玉宣言勝ちの棋譜が少ないため、それを補うため既存将棋プログラム(水匠)の自己対局で、入玉宣言勝ちの棋譜を生成し、dlshogi の自己対局のデータに混ぜて学習した。

4.13 ブートストラップ法による Value Network の学習

Value Network の学習の損失関数は、勝敗を教師データとした交差エントロピーと、探索結果の評価値を教師データとした交差エントロピーの和とした。

このように、本来の報酬（勝敗）とは別の推定量（探索結果の評価値）を用いてパラメータを更新する手法をブートストラップという。

経験的にブートストラップ手法は、非ブートストラップ手法より性能が良いことが知られている。

4.14 引き分けも含めた学習

将棋はルールに引き分けがあるゲームであるため、引き分けも正しく学習できる方が望ましい。そのため、自己対局で引き分けとなった対局も学習データに含めて学習した。

4.15 指し手の確率分布を学習

以前の dlshogi では、指し手のみを学習していたが、AlphaZero と同様に、自己対局時で MCTS で探索した際のルート局面の子ノードの訪問回数に従った確率分布を学習するように変更した。確率分布を学習することで、最善手と次善手の行動価値が近い場合に、次善手の行動価値を正しく学習できるようになる。

確率分布を学習することで、floodgate の棋譜に対する一致率が向上することが確認できたが、対局して強さを計測すると弱くなるという現象が確認できた。原因は、モデルの方策の出力の性質が変わるため、探索パラメータの調整が必要なためであった。Optuna を使用して探索パラメータを最適化(4.27 参照)することで、指し手のみを学習したモデルよりも強くすることができた。

4.16 同一局面を平均化して学習

自己対局では、序盤の同一の局面の教師データが多く生成される。それらの重複した局面を別のサンプルとして学習すると、モデルの学習に偏りが起きる。

局面の偏りをなくするために、同一の局面を集約し、指し手の確率分布と勝敗を平均化し、1 サンプルとして学習した。

4.17 評価値の補正

自己対局で生成するデータには、MCTS で探索して得られた勝率(最善手の価値)を局面の評価値を記録し、学習時にブートストラップ項(4.13 参照)として使用している。記録した評価値(勝率)が、実際の対局の結果から算出した勝率と一致しているか調べたところ、乖離しているという現象が確認できた。そのため、評価値を実際の自己対局での勝率に合うように、補正を行った。

4.18 SWA(Stochastic Weight Averaging)

画像認識の分野でエラー率の低減が報告されている手法である、SWA(Stochastic Weight Averaging)をニューラルネットワークの学習に取り入れた。一般的なアンサンブル手法では、推論結果の結果を平均化するが、SWA では学習時に一定間隔で重みを平均化することでアンサンブルの効果を実現する。

4.19 末端ノードでの短手順の詰み探索

モンテカルロ木探索の末端ノードで、5 手の詰み探索を行い、詰みの局面を正しく評価できるようする。並列化の方式により、GPU で計算中の CPU が空いた時間に詰み探索を行うため、探索速度が落ちることはない。

4.20 ルートノードでの df-pn による長手数の詰み探索

モンテカルロ木探索は最善手よりも安全な手を選ぶ傾向があるため詰みのある局面で駒得になるような手を選ぶことがある。

対策として、詰み探索を専用スレッドで行い、詰みが見つかった場合はその手を指すようにする。

詰み探索は、df-pn アルゴリズムを使って実装した。優越関係、証明駒、反証駒、先端ノードでの 3 手詰めルーチンにより高速化を行っている。

4.21 勝敗が確定したノードのゲーム木への確実な伝播

モンテカルロ木探索で構築したゲーム木の末端ノードで詰みが見つかった場合、その結果をゲーム木に伝播して利用する。つまり、モンテカルロ木探索に、AND/OR 木の探索を組み

合わせて、詰みの結果を確実にゲーム木に伝播するようにする。

4.22 PV 上の局面に対する長手数の詰み探索

ディープラーニング系の将棋 AI は、選択的に探索を行うために、終盤の局面で読み抜けがあると、頓死することある。

頓死を防ぐため、PV 上の局面に対して、df-pn による長手数の詰み探索を行い、詰みが見つかった場合、局面の価値を更新するようにする。

4.23 序盤局面の事前探索（定跡化）

出現頻度の高い序盤局面は、対局時に探索しなくても、事前に探索を行い定跡化しておくことができる。また、事前に探索することで、対局時よりも探索に時間をかけることができる。

ゲーム木は指数関数的に広がるため、固定の手数までの定跡を作成するよりも、有望な手順を選択的に定跡に追加する方が良い。自分が指す手は、1 つ局面につき最善手を 1 手（または数手）登録し、それに対する応手は、公開されている定跡や棋譜の統計情報を使って確率的に選択する。その手に対して、また最善手を 1 手（または数手）登録する。この手順により、頻度の高い局面については深い手順まで、頻度の低い局面については短い手順の定跡を作成することができる。

4.24 定跡作成時に floodgate の棋譜の統計を利用した確率分布を方策に利用

定跡を自分自身の探索のみで作成した場合、読み抜けがあった場合に定跡を抜けた後に不利な局面になる恐れがある。そのため、モンテカルロ木探索の PUCT の計算で、方策ネットワークの確率分布と floodgate の棋譜の統計を利用した確率分布を平均化した確率分布を利用し、致命的な読み抜けを防止する。

4.25 マルチ GPU 対応

複数枚の GPU を使いニューラルネットワークの推論を分散処理する。

「4.7 GPU によるバッチ処理に適した並列化」の方式により、GPU ごとに 2 つの探索スレッドを割り当てることで、GPU を増やすことでスケールアウトすることができる。ノードの情報は、すべてのスレッドで共有する。

確認できている範囲で 4GPU までほぼ線形で探索速度を上げることができている。

4.26 TensorRT を使用

モデルの学習にはディープラーニングフレームワークとして PyTorch を使用しているが、

対局プログラムには、推論用ライブラリの TensorRT を使用する。

TensorRT を使うことで、事前にレイヤー融合などのニューラルネットワークの最適化を行うことで、推論を高速化することができる。TensorCore に最適化されており、TensorCore を搭載した GPU では CUDA+cuDNN で推論を行う場合より、約 1.33 倍の高速化が可能になる[14]。

また、対局の実行環境にディープラーニングフレームワークの環境構築を不要とすることを目的とする。

4.27 Optuna による探索パラメータの最適化

PFN により公開された Optuna[15]を使用して、モンテカルロ木探索の探索パラメータ（PUCT の定数、方策の温度パラメータ）を最適化した。

Optuna は、主にニューラネットワークの学習のハイパーパラメータを最適化する目的で利用されるが、将棋エンジン同士の連続対局の勝率を目的関数として、探索パラメータの最適化に使えるようにするスクリプト[16]を開発した。Optuna の枝刈り機能により、少ない対局数で収束させることができる。

4.28 確率的な Ponder

モンテカルロ木探索は確率的にゲーム木を成長させる。その特性を活かして、相手が思考中に、相手局面からモンテカルロ木探索を行うことで、確率的に相手の手を予測して探索を行うことができる。予測手 1 手のみを Ponder の対象とするよりも、効率のよい Ponder が実現できる。

4.29 ノードのガベージコレクションとノード再利用処理の改良

世界コンピュータ将棋オンライン大会でノード再利用に 10 秒以上かかる場合があるこがわかったため、ノード再利用の方式の見直しを行った。

以前は、オープンアドレス法でハッシュ管理を行っており、ルートノードから辿ることができないノードをすべてのハッシュエントリに対して線形探索してノードの削除をおこなっていた。

これを、Leela Chess Zero のゲーム木の管理方法[17]を参考に、ゲーム木をツリーで管理を行うようにし、ルートの兄弟ノードをガベージコレクションする方式に変更した。ノードの

---

[14] https://tadaoyamaoka.hatenablog.com/entry/2020/04/19/120726

[15] https://optuna.org/

[16] https://github.com/TadaoYamaoka/DeepLearningShogi/blob/master/utils/mcts_params_optimizer.py

[17] https://tadaoyamaoka.hatenablog.com/entry/2020/05/05/181849

合流の処理が行えなくなるという欠点があるが、ノード再利用を短い時間で行えるようになった。

4.30 飛車と角の利きのビット演算

第 31 回世界コンピュータ将棋選手権の Qugiy のアピール文章[18]による、飛車、角の利きをビット演算により求める方法を実装した（実装はやねうら王のソースコードを参考にした)。ZEN2 の CPU で NPS が約 1% 向上した。

4.31 2 値の入力特徴量を 1 ビットで転送することで推論のスループットを向上

マルチ GPU を使用した場合、4GPU 以上では CPU と GPU 間の帯域がボトルネックになるため、2 値の入力特徴量を float の代わりに、1bit で表現し、GPU にビットで転送後、GPU 側で CUDA のプログラムでバッチ単位に並列に float に戻す処理を実装した。こうすることで、転送量が削減でき、NPS が 36.6%向上した。

4.32 Stochastic Multi Ponder

相手番に、相手番の局面から探索を行う Stochastic Ponder（4.28 参照）と、次の相手の指し手を複数予測し、並列に分散して探索を行う Multi Ponder を組み合わせて使う。

shotgun で実装されていた Multi Ponder[19]では、技巧 2 の Multi PV の結果を利用しているが、dlshogi の Stochastic Ponder では、ほとんどの場合、相手局面でのゲーム木が展開済みであり、ルートの子ノードの訪問回数を参照することで、有望な予測手を N 手取得することができる（ゲーム木が未展開の場合は、方策ネットワークの推論結果を使用する)。

また、予測した N 手以外の手が指された場合、Stochastic Ponder でも並列に探索を行っているため、Multi Ponder を使用しない場合と遜色のない手を指すことができる。

ponderhit した場合、次の局面の指し手予測の第一候補をその ponderhit したエンジンに割り当てることで、前回の探索結果を再利用する。

[18] https://www.apply.computer-shogi.org/wcsc31/appeal/Qugiy/appeal_210518.pdf
[19] http://id.nii.ac.jp/1001/00199872/

Serenade アピール文書

谷合廣紀

2025 年 3 月 30 日

1　独自の工夫

- Stockfish の NNUE を参考にした、評価関数アーキテクチャと探索部
- 教師データのデータベース管理による、教師データの選別と定跡作成
- LLM を主体とした、複数エンジンのオーケストレーション

2　使用ライブラリ・使用データ

- ライブラリ: やねうら王, cshogi, stockfish
- 教師データ: nodchip/tanuki-.nnue-pytorch-2024-07-30.1, nodchip/shogi_hao_depth9

INUGAMI
アピール文書

ザイオソフト コンピュータ将棋サークル
野田久順 岡部淳 鈴木崇啓
河野明男 伊苅久裕

目次

- INUGAMI
- 改良点
- 使用ライブラリ

INUGAMI

- 『松山騒動八百八狸物語』に登場する化け狸が元ネタです。

『隠神刑部』をイメージして AI で生成した
『INUGAMI』のイメージ画像

改良点（1）

- 評価関数の学習に知識蒸留を用いています。
  - 知識蒸留は教師モデルの知識を生徒モデルに伝達する手法です。
  - Response-based Knowledge のオフライン知識蒸留を行っています。

改良点（2）

- 知識蒸留の具体的な手順は以下の通りです。
  1. CPU エンジン同士で自己対戦し、学習データを生成する。
     - 学習サンプルには局面と評価値、勝敗等を含む。
  2. 学習データの局面を GPU エンジンの評価関数に推論させ、勝率を出力させる。
  3. 勝率を評価値に変換する。
  4. 学習サンプルの評価値を 3. で置き換える。
  5. 4. を用いて NNUE 評価関数を学習する。

使用ライブラリ

- やねうら王
  - やねうら王を元に改造した思考部を使用しています。
    - 独自の工夫を加えるにあたり、改造しやすく、レーティングも高いためです。
- 水匠
  - DL 水匠を使用しています。
    - GPU エンジンの評価関数の中で知識蒸留をした際に最もレーティングが高かったためです。
- tanuki-
  - 棋譜の生成に使用ています。
    - 過去に開発した資産の再利用のためです。
- nnue-pytorch
  - 評価関数の学習に使用しています。
    - レーティング向上と学習時間の高速化のためです。

『松山騒動八百八狸物語』のあらすじを読み、
『隠神刑部』の境遇が可哀そうすぎて悲しくなりました。

[English]

In the history of infectious diseases, a single drug has been easily resisted by mutations in the pathogen. In contrast, when multiple drugs are used simultaneously, each helps prevent the emergence of resistance to the others. Ryfamate derives from Rifamate which is the combination of two medicines for tuberculosis to retard the development of drug resistance. As with this method, Ryfamate aims to combine an NNUE alpha-beta search with deep learning (DL) Monte Carlo tree search to compensate for each other's shortcomings.

Ryfamate employs a modified majority voting system integrating a deep-learning-based evaluation engine and an NNUE-based evaluation engine [1]. By combining a deep-learning engine optimized primarily for GPU computation and an NNUE engine designed for CPU computation, Ryfamate effectively leverages the complementary strengths of both approaches. This hybrid configuration enables efficient utilization of computing resources within a single PC, significantly enhancing overall performance.

The author also proposes the Ryfamate Cross Network (RyfcNet) [2], an advanced deep-learning architecture specifically designed to enhance Shogi evaluation functions by effectively integrating the following distinctive components:

1. **S-Layer**: Conventional convolutions employing standard square-shaped kernels.
2. **C-Layer**: Extended convolutions utilizing kernels that span the full length of the input space across two or more dimensions, enabling selective dimensional shifts.
3. **A-Layer** / **F-Layer**: Spatial self-attention layer / channel-wise fully connected layers.

Standard deep-learning-based Shogi evaluation models [3] typically experience substantial computational challenges during training, particularly as the depth of the network increases. To address these limitations, RyfcNet introduces several strategic innovations:

1. Integration of multiple activation functions coupled with modified skip-connection architectures to alleviate gradient degradation and enhance feature propagation.
2. Implementation of ensemble learning and knowledge distillation techniques, leveraging multiple types of evaluation functions as teacher models to enrich learned representations.
3. Adoption of a hybrid optimization approach that selectively applies RAdam or LAMB optimizers alongside Stochastic Gradient Descent (SGD) at the layer level, significantly improving training efficiency and stability.

In recent years, large-scale neural models, exemplified by large language models (LLMs), have attracted considerable attention due to their exceptional representational power. Analogously, the innovations embodied within RyfcNet promise efficient scaling of Shogi evaluation models to accommodate greater parameter complexity, thereby improving predictive accuracy across diverse gameplay scenarios.

[Library]

YaneuraOu : https://github.com/yaneurao/YaneuraOu

dlshogi : https://github.com/TadaoYamaoka/DeepLearningShogi

* Ryfamate also uses a large amount of data that Kano-san, Yamaoka-san, Tayayan-san, nodchip-san have published.

[Reference]

[1] 水無瀬香澄, “Ryfamate 開発記と変則多数決合議,” コンピュータ将棋協会誌 Vol.36, 2025.

[2] Komafont, “Ryfamate Cross Network,” 第 33 回世界コンピュータ将棋選手権, 2023.

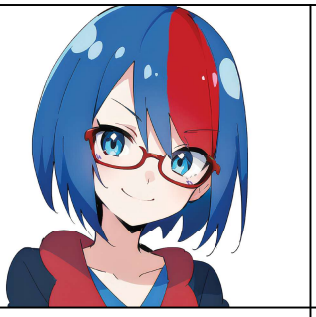
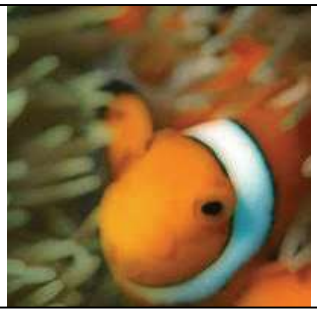
[3] 山岡忠夫, 加納邦彦, “強い将棋ソフトの創りかた Pythonで実装するディープラーニング将棋AI,” マイナビ出版, 2021.

第35回世界コンピュータ将棋選手権

キーワードは「わかりやすさ」
大規模言語モデルを用いた
将棋初心者支援システムの提案

水匠チーム 創造研究開発部 熊田ゴウ(主任)/たややん/アイシア＝ソリッド/shimojolno

プロジェクトメンバー

			
熊田ゴウ	たややん	shimojolno	アイシア＝ソリッド
• リーダーとしてプロジェクト立上げから統括、開発を担う。 • 普段は主にユーザー支援のための研究開発に従事。 • VTuberでアマチュア四段。	• 水匠を強くする係＆DL系AIを用いて本プロジェクトに貢献する担当。 • 普段は弁護士！	• 先行研究調査を行い、本プロジェクトの位置づけを明確化。手法検討のサポートなど。 • 普段は企業のデータサイエンティスト。 • 3手詰めが精一杯。	• 将棋初心者を担当。最新のAI動向を共有しつつ、局面類似検索で爪痕を残す予定 • 普段はデータサイエンス VTuber として統計や AI の動画を投稿 • 棋力はウォーズ5級
X：@kumada_goh Youtube：@kumada_goh	X：@tayayan_ts YouTube：@tayayan_ts	X：@shimojolno reserchmap：asaya	X：@AIcia_Solid Youtube：@AIcia_Solid

研究サマリ

背景と課題

- 将棋人気が高まっているが、初心者向けの支援は不十分。
- AI評価値は初心者の理解を助ける一方、評価値以上の情報を与えられない。
- 自分で調べようと思っても名称不明で調べられない上、学習を支援する人員は不足しているため、初心者が取り残されている。

手法と結果

- 観戦者向けの理解支援：将棋知識DBとLLM(ChatGPT 4o)の活用で、人間に理解できる・納得感のある解説を生成。
- 初心者向けの学習支援：段位者AIと級位者AIを戦わせ、級位者によくある失敗と理由を解説。
- 多言語対応：上記システムは即座に多言語化可能。
- 結果、将棋初心者の将棋学習を支援し、継続的に将棋を楽しむためのシステムを構築した。

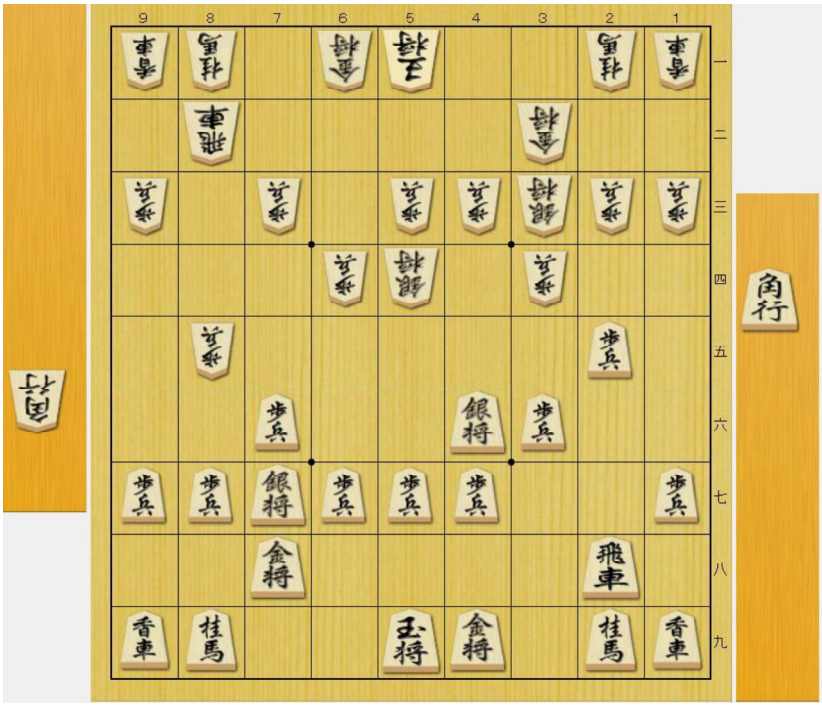
今後の展望

- プロの将棋の中継に対してもAIによる自動解説を行うなどし、広く将棋初心者に向けて価値を提供すること。
- システム内部の知識DBの改善やLLMのチューニング等により、プロが使えるようなシステムを開発すること。

抜粋

1.2.3 現代の将棋AIの限界：難しすぎる

ユーザー局面



システム出力

将棋AIによる出力：

評価値

-32

読み筋

▲６九玉(59) △４四歩(43) ▲５六歩(57) △５二金(61) ▲９六歩(97) △４二玉(51) ▲７九玉(69) △９四歩(93) ▲８八玉(79) △１四歩(13) ▲１六歩(17) △７四歩(73) ▲３五歩(36) △４三銀(54) ▲３四歩(35) △同　銀(43) ▲５五歩(56) △３一玉(42) ▲６六歩(67) △２二玉(31) ▲５八金(49) △６五歩(64) ▲同　歩(66) △３九角打 ▲３八飛(28) △８四角成(39) ▲６七金(58) …

抜粋

2.1 開発方針

将棋をわかりやすく。

問題

- 😇 難しすぎる
- 🤷‍♂️ 説明なし
- 😓 疑問が残る

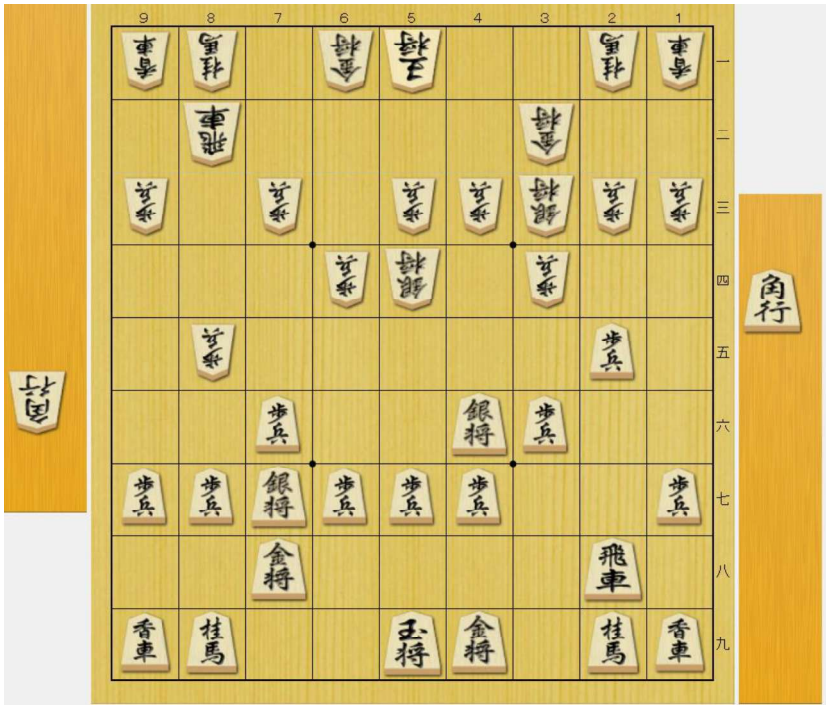
解決

- 🐣 初心者にもわかる！
- 💬 言葉でわかる！
- 🙋‍♀️ 疑問に寄り添う！

抜粋

3.1 開発結果1（手法A）

ユーザー局面



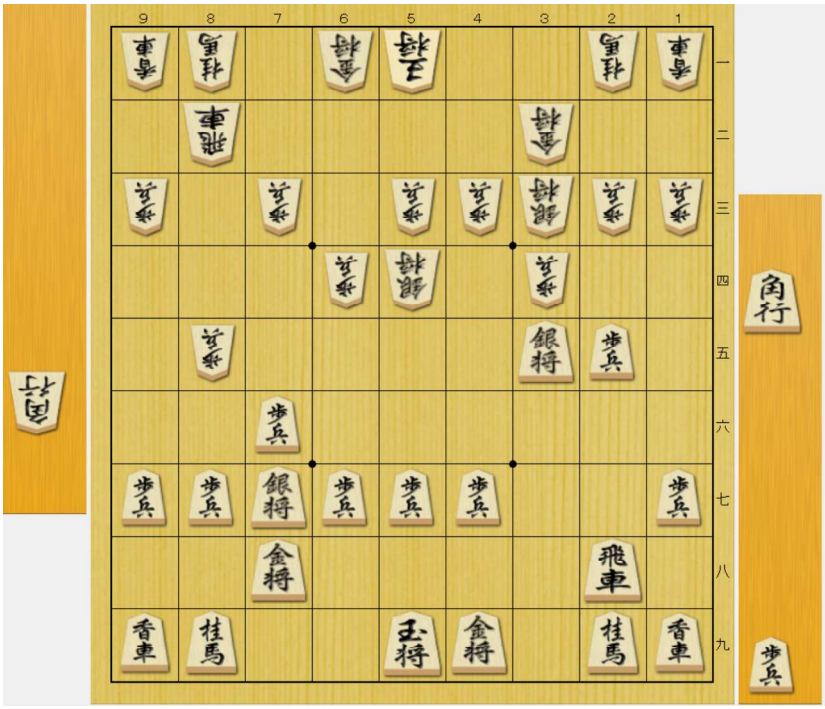
システム出力

知識DB × ChatGPTで
解説文を生成：
この局面は角換わり早繰り銀対腰掛け銀ですね。
このまま仕掛けると王手飛車になるので、玉を上がるか端歩を突いてください！

抜粋

3.2 開発結果2（手法B）

ユーザー局面



システム出力

級位者AIが王手飛車を喰らうその筋を解説し学びを提供：

この局面で、相手は△３五歩という手を狙っていると思われます。この手で、**銀を取ることを狙っているので注意しましょう。**

この局面で▲２四歩という手が見えるかもしれません。
しかし、▲２四歩、△同歩、▲同銀、△同銀、▲同飛、△１五角、▲６八玉、△２四角のように進めてしまうと、飛車を取られてしまうため、注意が必要です。

この局面では▲４六銀を指すことが考えられます。
▲４六銀、△４四歩、▲６九玉、△４五歩、▲３七銀という進行が一例で、相手の狙い（△３五歩）に対応することができます。

目次

1. 背景と目的

1.1 将棋ブームによる好機と課題

取り残された課題

- 藤井聡太八冠の誕生で将棋が注目を集めている一方、
  競技人口は減少傾向。*1

- スマホアプリの普及で「始める環境」は整ったが、
  「続けるための支援」が不足。

- **AI を活用した初心者支援**が拡充できれば、
  より**将棋界を盛り上げられる可能性**がある！

*1 公益財団法人 日本生産性本部 レジャー白書2024

1.2 初心者が抱える問題と解決できない構造

やり始めても続けられない

①初心者を待ち受ける将棋学習の壁

駒の動きや基礎の囲いを覚えた後、200超の未知の戦法が立ちはだかる。*1

将棋AIの普及は進んだものの、出力が高度すぎて初心者の学習には向かない。

将棋の難しさに魅力は感じるものの、学ぶための取っ掛かりが少なすぎる。

②初心者を支援する人が足りない。

人に教われば上記の悩みは解消するが、プロと普及指導員は約1200名。*2

将棋をやめてしまう理由に「指導者や教室がないこと」が約30%。*3

*1 本研究で列挙した結果。局面を見ても名前がわからず、勉強したくても調べることすらできない

*2 日本将棋連盟HPより

*3 文化庁 平成２９年度 生活文化等実態把握調査事業報告書 p33

1.2.1 指す将にとっての学習環境の例

AIにより学びやすくなっているが、まだ大きな壁がある。

- **アクティブな学びを得るのが難しい環境**

  棋書や詰将棋などでパッシブな学びはできるが、根気が必要で継続が難しい。
  対局〜検討等アクティブな学びができれば楽しいが、オンライン中心では難しい。

- **今のAIで学べること、学べないこと**

  評価値を見れば、悪手と最善手はわかる。
  しかし、それぞれの手の意味が分からなければ、やはり実戦には活かしづらい。

  ※例外的に、短手数の詰みが生じている局面で指し手の意味がわかることもある。
  しかし初心者にとっては、それ以外ほぼすべての局面で最善手の意味はわからない。

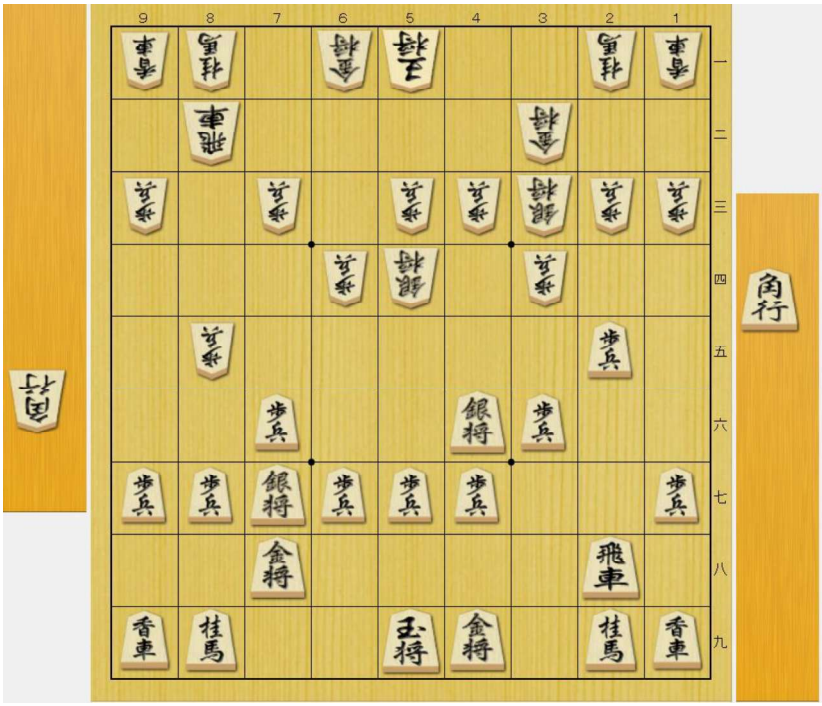
1.2.2 戦法の例：相居飛車の定跡

これでもまだ一部。多すぎる！

矢倉	急戦矢倉	角換わり	相掛かり	横歩取り	雁木	その他
3七桂4七銀 棒銀 早繰り銀 四手角 雀刺し 森下システム 脇システム	米長流急戦 阿久津流急戦 矢倉中飛車 右四間飛車 カニカニ銀 居角左美濃 4二飛戦法 3五歩早仕掛け	腰掛銀 58金型 木村定跡 48金29飛型 早繰り銀 棒銀 ４五桂戦法 右玉 後手一手損	原始棒銀 相掛かり棒銀 UFO銀 早繰り銀 腰掛銀 ひねり飛車 中原流 鎖鎌銀 塚田スペシャル	△3三角 △3三桂 △2三歩 △4五角 △8五飛 ▲3三飛成 相横歩取り 青野流 △４一玉戦法 勇気流 竹部スペシャル	ツノ銀雁木 旧型雁木	右玉 袖飛車 陽動振り飛車

1.2.3 現代の将棋AIの限界：難しすぎる

ユーザー局面



システム出力

将棋AIによる出力：

評価値

-32

読み筋

▲６九玉(59) △４四歩(43) ▲５六歩(57) △５二金(61) ▲９六歩(97) △４二玉(51) ▲７九玉(69) △９四歩(93) ▲８八玉(79) △１四歩(13) ▲１六歩(17) △７四歩(73) ▲３五歩(36) △４三銀(54) ▲３四歩(35) △同　銀(43) ▲５五歩(56) △３一玉(42) ▲６六歩(67) △２二玉(31) ▲５八金(49) △６五歩(64) ▲同　歩(66) △３九角打 ▲３八飛(28) △８四角成(39) ▲６七金(58) …

1.3 将棋AIの進化と課題

強さの追求と取り残される初心者

🤖将棋AIの発展	☹️既存の支援の限界
• 将棋AIは強すぎる！ • プロがAIの将棋を研究する時代 • 「初心者のためのAI」が未発達	• 既存の教材は一方通行 • 初心者が感じた実践的な疑問を解消できない • 人に聞こうにも教えられる人間が少ない

⇒ 初心者への新たな支援が必要！

1.4 研究の目的

初心者の「わからない」「不安」を「わかる」「楽しい」へ

1.人間らしいフィードバックやガイドで
初心者の疑問解消をサポートする

2.誰でも簡単に利用できるローコストな支援手法を探求

3.楽しみながら学び、続けられる環境を作る

1.5 局面解説AIに関する先行研究

局面の言語化を行ってきた研究たち。

	技術アプローチ	出力形式	評価方法	メリット	デメリット
亀甲ら (2017)	解説木を用いた手順の予測	局面の遷移を含む解説文	指し手予測モデルの精度評価	局面遷移を考慮	解説木が大きくなりすぎる
佐々木・関 (2023)	コーパス構築と機械学習モデルのFine-tuning	構成要素を分類した上での解説文	BLEU-2スコア	解説文の構成要素を体系的に分類	指し手の読み筋や戦型のみに特化
中村・小田 (2024)	局面価値と着手価値評価の数値データを言語化	囲碁特有の表現を用いた解説文	（未実施）	着手の特性（好手/悪手）の表現	評価実験が未実施
山内・河原 (2024)	手順のテキスト化と言語モデルのマルチタスク学習	対局の流れを考慮した解説文	ROUGE-L、BLEU、BERTScore	時系列の流れを考慮した入力形式	着手の意味理解が不十分

1.6 先行研究の課題と本研究のゴール

先行研究の課題：

解説文生成のための表現や言語モデルの選択がメインなため、

「初心者にとってのわかりやすさ」を十分に考慮した
実用的な支援システム構築には至っていない。

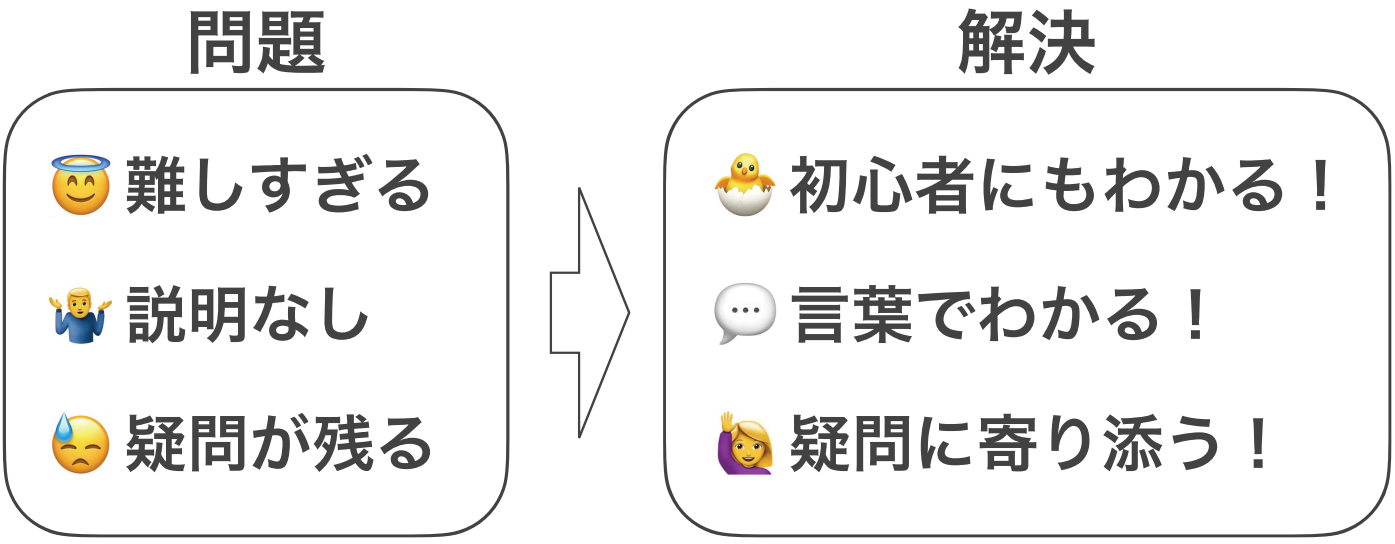
本研究のゴール：

局面の戦型認識と最適な次手推薦を100字程度の簡潔な自然言語で提示することで、初心者将棋への局面理解を促進する実用的な支援システムを構築すること。

2. 手法

2.1 開発方針

将棋をわかりやすく。



2.2 今回用いる2つの手法

将棋の勉強に重要な「知識」と「読み」を2つの手法でサポート

- **手法A：知識DBとLLMを活用して解説を生成する（知識のサポート）**

  将棋の局面（部分図）と、その狙いや注意点等の解説データを搭載したDBを作成。
  与えられた局面に類似する知識をDBから呼び出し、LLMを用いて解説を生成。
  ⇒ 現局面の意味合いや戦型の名前を示しつつ、今後の方針をわかりやすく解説する。

- **手法B：棋力差のあるAI同士を戦わせ、学びのある手筋と解説を生み出す（読みのサポート）**

  段位者と級位者の棋譜で追加学習し、それぞれ段位者・級位者の指し筋を再現したAIを作成。
  与えられた局面から段位者AIと級位者AIに対局させ、成り駒や駒の損得が生じる手筋を発見。
  ⇒ 級位者が陥りがちなミスを具体的な手順で示すことで、学びのある解説を生み出す。

2.3【手法A】人間の言語化の仕組み

有段者は局面をどのように言語化しているか。

①思い出す/読む

角換わり…
△1五角…
早繰り銀…

②言語化

「角換わり早繰り銀対腰掛け銀の形だね。」
「このまま仕掛けると王手飛車になるよ。」

⇒ AIを使って実装すればいい！

2.4【手法A】「わかりやすさ」を届ける仕組み

局面の取得	ユーザーが不明な局面で**ヒントボタン**を押す (その時点の局面情報を取得)
DBを検索	データベース上で**最もマッチする知識**が検索される
AIが言語化	マッチした知識を踏まえて、 **AIが言語化**する
画面に出力	マッチした知識と言語化された内容が **画面に出力**される

2.5【手法A】知識データベースと検索の仕組み

将棋の基本的な知識を蓄積、検索できる。

📚知識データベース	👩‍💻検索の仕組み
• 初心者向けの基本的な形とその形に関する説明を蓄積 • 戦法は200種、囲いは40種程度収録 (みんなで頑張って作成)	• ユーザーの局面図と知識DBの部分図を比較 ＝駒の並びが最も類似する知識を抽出 (完全一致じゃなくてOK)

2.6【手法A】知識検索の例：戦法の判定

ユーザーの現局面

マッチした知識(赤枠)と付随情報

```
Match 1:
Size: 9×5
Window:
[['k' '.' '.' 'n' 'l']
 ['.' '.' 'g' '.' '.']
 ['p' 'p' 's' 'p' 'p']
 ['s' '.' 'p' '.' '.']
 ['.' '.' '.' 'P' '.']
 ['.' 'S' 'P' '.' '.']
 ['P' 'P' '.' '.' 'P']
 ['.' '.' '.' 'R' '.']
 ['K' 'G' '.' 'N' 'L']]
Matched pattern: pattern_0
Similarity: 1.00

Description: 角換わり早繰り銀の形の例。このまま仕掛けると
△1五角で王手飛車になるので注意！玉を上がるか、端歩を突こう。
```

⇒ 情報を整形後、画面へ表示。

2.7【手法A】言語化の仕組み：大規模言語モデル(LLM)

人間らしく言語化してくれる。

- Open AI 社の「**ChatGPT 4o**」を利用した。
- 人間との会話が行える**言語処理AI**。
- ファインチューニング*などは行わず、
  公式のモデルをAPIで呼び出す形で利用した。

*特定の文章などを学習させ、専門的な出力を可能にする手法

2.8【手法A】言語化の仕組み：LLMへのプロンプト(命令)例

前提共有

付与情報

出力方針

ルール：
あなたは**将棋の局面図を言語化するスペシャリスト**です。
①の情報を踏まえ、②の方針に従って出力してください。

①あなたに与えられる情報は以下のとおりです。
- 現在の局面図:
- 先手視点の検索結果:
- 後手視点の検索結果:

②以下のルールに従って局面についての方針を示してください。
- **簡潔な日本語で**ユーザーに指針を示す
- **将棋に詳しくない人でもわかるように**話す
- **最大100字程度で**出力する

2.9 【手法B】庶民的な棋力のAIの作り方

級位者・段位者のデータでFine-Tuning（微調整）

①まず将棋AIを作ります

③級位者っぽい手を指す将棋AIが完成！

②級位者の棋譜データを与え、級位者の着手を再現するよう学習

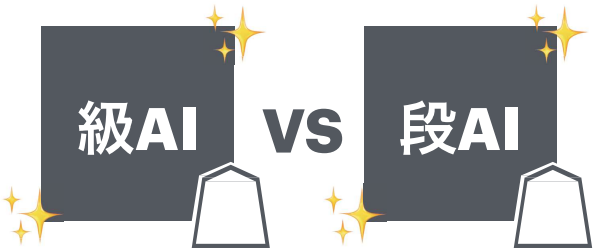
※一連の流れは、近年のLLMの学習方法と類似
まず一般的なタスクを学習させ (Pre-Training)、その後個別のタスクに調整 (Fine-Tuning)
今回の場合、まず将棋AIを作って将棋を理解させた後、級位者の考え方を学習させるイメージ

2.10【手法B】庶民的AIを用いた解説文生成

級位者 vs 段位者を通して庶民的でわかりやすい学びを得る

- **以下のステップで指し手と解説文を生成**

1. 指定局面から**級位者AIと段位者AI**に対局させる
2. 段位者AIが**駒得**や**成り駒作り等**に成功したら終了
3. 着手の推定選択率*を加味しながら解説文を生成



- **狙いと得られる効果**

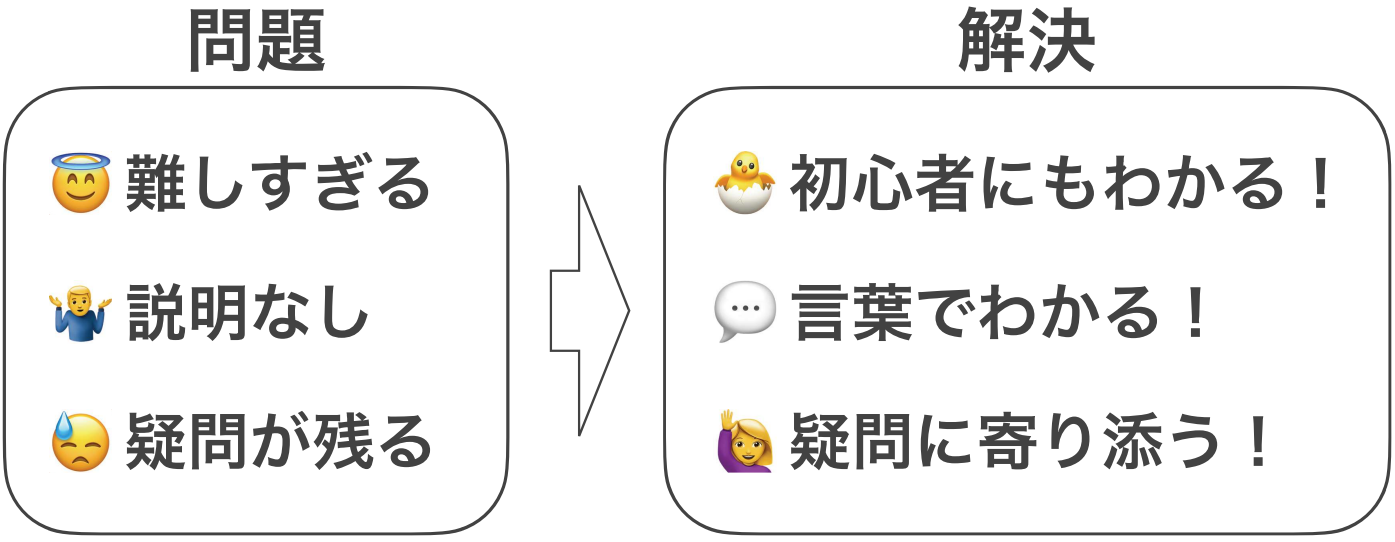
級位者AIが**初心者にありがちなミスを発見**し、段位者AIが**人間に理解可能な手筋で咎める。**
これにより、初心者でも理解可能かつ、深い学びにつながる解説を提示できる。
また、推定選択率を絡めることで「どの程度自分でも指せないといけなかったか」を
分かりながら振り返りを行える

*Policyという言葉をたややん氏が日本語化したもの。AIによる、次の一手でどの手を選ぶべきかを示す確率分布のこと。

3. 結果

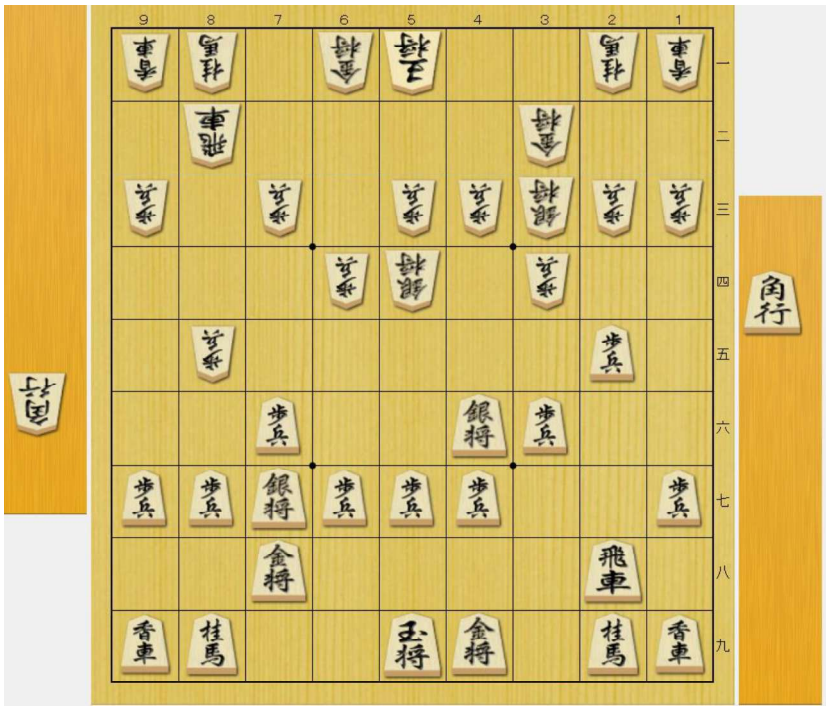
2.1 開発方針（再掲）

将棋をわかりやすく。



3.1 開発結果1（手法A）

ユーザー局面



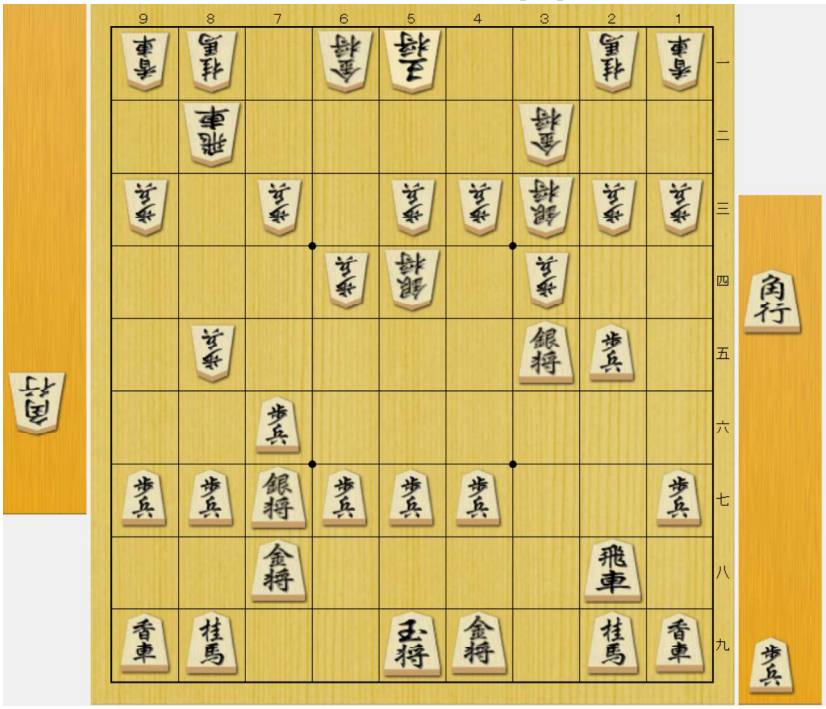
システム出力

知識DB × ChatGPTで解説文を生成：

この局面は**角換わり早繰り銀対腰掛け銀**ですね。
このまま仕掛けると**王手飛車になる**ので、**玉を上がるか端歩を突いて**ください！

3.2 開発結果2（手法B）

ユーザー局面



システム出力

級位者AIが王手飛車を喰らうその筋を解説し学びを提供：

この局面で、相手は△３五歩という手を狙っていると思われます。この手で、**銀を取ることを狙っているので注意しましょう。**

この局面で▲２四歩という手が見えるかもしれません。
しかし、▲２四歩、△同歩、▲同銀、△同銀、▲同飛、△１五角、▲６八玉、△２四角のように進めてしまうと、飛車を取られてしまうため、注意が必要です。

この局面では▲４六銀を指すことが考えられます。**▲４六銀、△４四歩、▲６九玉、△４五歩、▲３七銀という進行が一例**で、相手の狙い（△３五歩）に対応することができます。

3.3 開発結果：将棋がさらにわかりやすくなった。

まるで人間に指導してもらえているような体験

- **初心者にもわかりやすい**
  手順や評価値だけでなく**やさしい言葉で説明する**ため、**初心者の疑問を解消！**
- **指導してもらえているような体験**
  言葉によるガイドにより、**人間に教えてもらっているような安心感**を得られる！
  知識がなくても次の一手を考えやすく、**ストレスなく純粋に将棋を楽しめる！**
- **離脱防止&継続促進**
  指した将棋に関連する知識が自動で提供されるため、**効率的に学習を行える！**
  「わからない→やめる」ではなく、「わかる→続けたくなる」

3.3.1 AI活用例①：ぼこぼこに負けて途方に暮れたとき

次こそは勝てるように。

- **初心者はまず序盤で劣勢になり、悔しい思いをする。**
  「なぜか駒を全部取られた……」「よくわからない何かが起きて負けた……」
  なので学んで対策し、次は勝ちたい。
- **しかし、形の名前がわからないため検索しようがなく、学べない。**
  そもそも、**「定跡を知らなかったから負けたのか？」「ただ力負けしたのか？」**
  という疑問が残り、解消されない。

⇒AIのサポートで、悔しい時すぐに身のある学習ができる！

3.3.2 AI活用例②：対局相手の事前対策をしたいとき

本番で力を出し切れるように。

- **対局相手が事前にわかっていて、相手の棋譜を持っている場合もある。**
  相手の棋譜からは、相手の得意としている戦法を学べるはず。
  さらには、相手が負けている棋譜からは、**苦手な戦法を学べるはず**である。
- **万全の状態で迎えたいのに、対策できないもどかしさ。**
  しかし、**相手の棋譜を見ても、居飛車か振り飛車かくらいしか分からない。**
  なので、対策もできない。（←本当にできない。なかなかもどかしい）

⇒AIのサポートで、事前にしっかり対策できる！

3.3.3 AI活用例③：プロの将棋がわからないとき

解説をお願いします。

- **プロの将棋は難しい。**
  そもそも将棋が難しいのに、プロはさらに複雑なことをやっている(らしい)
  解説してくれるプロがいるときは、理解できて楽しい。
- **解説してもらえないと何もわからない。**
  しかし、中継の間ずっと解説してくれるわけではない。
  解説がない間は静かに盤面だけが映されていて、**初心者は取り残される。**

⇒AIが基礎知識や手順を解説し、安心して楽しめる！

4. 今後の展望

4.1 観る将にとっての将棋AI

「難しいけど、観ていてとても楽しい」(級位者代表：アイシア)

- **早指し戦や最終盤は今のAIだけでもかなり楽しい。**

  指し手も早いし、評価値が大きく動く。**「最善手以外悪手」だと手に汗握って観戦できる。**

  （第91期ヒューリック杯棋聖戦五番勝負第1局藤井竜王vs渡辺棋聖（当時）の連続王手など）

  現代の将棋観戦は、数字の大小を見るだけでも楽しめる（超簡単！）のに、

  奥に底知れぬ深みと人間ドラマがあり、歴史もある。

  おそらく、**観戦される娯楽の中で最も良い特性を持っている！**

- **しかし、序盤中盤の観戦は楽しみにくい。**

  初心者にとっては、AIの示す手の意味はわからないし、評価値の違いもわずか。

  プロの解説や、勝負飯等の話題がないと、観続けるのは難しい。

4.2 広く将棋を普及するために

- **西尾明七段（将棋連盟常務理事）「将棋ChatGPTの開発が進めば人間の学習機能向上だけでなく、さまざまな将棋サービスの向上が期待できる」** *1
- **もし、プロの対局中継に本システムを導入したら……**

  例1：プロの解説がない時間に、システムが解説。

  ⇒視聴者の理解を助け、将棋をもっと学べて、もっと楽しめる！

  例2：中継記者に本システムを活用いただき、業務時間を短縮。

  ⇒人間にしかできない取材という仕事が、さらに高品質になる！

  例3：LLMによる柔軟な多言語対応で、誰にでも「わかりやすく」。

  ⇒生まれた場所に関係なく、将棋を楽しめる！

*1 “「なぜその手を選んだか」　将棋AIが教える時代到来へ” 2024.03.25 NIKKEI BizGate

4.3 さらに高度なシステムについて

- **もし、プロのために本システムを改善したら……**

  今回は最も多くのユーザーに価値を提供できるよう、対象を初心者とし、基礎的な出力のみを目標とした。

  今後は、システム内部のデータを精細化、LLMのチューニング等、探索システムの開発等を行うことで、プロにとっても有用な出力を行える可能性がある。

- **プロに焦点を当てたシステム開発。**

  プロにとって当たり前の知識、当たり前の言葉遣い、第一感。
  それらの模倣や、強いAIとの組み合わせなど、研究の余地がある。

5. 引用文献

引用文献

[1] 亀甲博貴, 森信介, 鶴岡慶雅: "将棋解説文生成のための解説すべき手順の予測", 情報処理学会論文誌, Vol.58, No.12, pp.2070-2079 (2017).

[2] 公益財団法人 日本生産性本部 レジャー白書 (2024)

[3] 佐々木謙人, 関洋平: "将棋解説文の構成要素を考慮した解説文生成手法の検討", DEIM Forum 2023, 1a-7-5 (2023).

[4] 中村貞吾, 小田直輝: "着手価値評価を考慮した囲碁棋譜からの解説文生成", 2024年度電気・情報関係学会九州支部連合大会, 09-1P-04 (2024).

[5] “「なぜその手を選んだか」　将棋AIが教える時代到来へ” 2024.03.25 NIKKEI BizGate

[6]文化庁 平成２９年度 生活文化等実態把握調査事業報告書

[7] 山内悠輔, 河原大輔: "手順のテキスト化による将棋解説文生成", 言語処理学会第30回年次大会発表論文集, pp.3197-3202 (2024).

6. 今回の水匠の工夫

今回の水匠について

- **将棋初心者支援システム以外の部分（水匠の棋力の改善点）**
  NNUE（HalfKP512x2_8_64）を使用。
  nodchip氏の工夫（DL水匠で教師データの評価値を付け直して学習させる）を採用。
  参考：https://nodchip.hatenablog.com/entry/2024/12/25/000000

- **上記工夫に加えて...**
  Lambda = 1.0（勝敗項を考慮しない）で学習することで棋力向上。
  局面データをuniqueなものとする（重複を排除する）ことで棋力向上。
  Ryfamateモデル（感謝：Komafont氏）を活用することで棋力向上。

7. 謝辞

謝辞

知識DB作成に尽力してくださった皆さん。どうもありがとう！！

写真	名前	X	Youtube
	美野辺沙羅@怨霊VTuber	@Sara_Minobe	怪奇放送局野辺チャンネル
	甘党あずを🍵🍡Vtuber	@amatou_azuwo	Azuwo Ch. 甘党あずを
	一梨透🍮将棋系VTuber	@ichilitre_V	一梨透
	⛩鷺宮ローラン【プロe棋士Vtuber】プロゲーマー👽	@SaginomiyaL	鷺宮ローラン（バーチャル将棋星人Vtuber）
	五反田えぬ	@Gotanda_N	五反田えぬ
	四宮式@小説系VTuber	@YotsumiyaS	四宮 式 -Yotsumiya Shiki-
	篠崎マコト	@thekeymaple_1	篠崎マコト

やねうら王 PR 文書

■ 定跡について

やねうら王では、今回『新ペタショック定跡』を搭載している。

これは、1 局面につき 2 億ノード以上探索して生成した定跡である。探索したあと定跡をペタショック化している。2025 年 3 月 23 日時点で 80 万局面程度ある。大会当日までには 150 万局面程度になると見込まれる。

ペタショック化とは、定跡ツリーを minimax 化したものが局面数 N の定数倍の時間で得られる非常に優れたアルゴリズムである。このアルゴリズムの詳細についてはどこかに論文を書く。この実装は、やねうら王の GitHub のコードにある。

やねうら王の GitHub
https://github.com/yaneurao/YaneuraOu

■ 探索について

あとで書く。

■ 評価関数について

あとで書く。

AobaZero の 2025 年のアピール文書

山下 宏
yss@bd.mbn.or.jp

1　AlphaZero の追試が最初の目的

AobaZero は Bonanza、LeelaZero のコードをベースに AlphaZero の追試をするべく MCTS ＋ディープラーニングで実装されてます。ネットワークは 3x3 のフィルタが 256 個の 20 block の ResNet でパラメータの個数は 2340 万個。棋譜生成をユーザの皆様と協力して行う分散強化学習です。オープンソースです[*1]。

2　AlphaZero の追試は 2021 年 4 月に終了

AlphaZero の将棋の追試は、2019 年 3 月から開始し、2021 年 4 月に 3900 万棋譜を作成して終了しました[*2]。2025 年 3 月 31 日現在、7109 万棋譜を作成しています。

3　追試終了後から +260 ELO、去年の選手権から +0 ELO

追試終了後からは +260 ELO、去年の選手権からは +0 ELO で、ほぼ同じです。追試終了時では AlphaZero より +150 ELO 弱い、という推定でしたが現在は +260 なので AlphaZero を +110 ELO 程度、超えた棋力かもしれません。

4　昨年の選手権からの主な変更点

昨年は振り飛車のゼロからの強化学習「Aoba 振り飛車」[*3]を主に開発していたので目立った進展はありません。ただ、生成された振り飛車の棋譜を AobaZero の棋譜に 20 %から 25 %ほど混ぜると 128x10 block の小さいネットワークだと +50 ELO ほど強くなりました[*4]。

AobaZero は振り飛車の棋譜が 2 %しか含まれていないので、もう少し多様な戦法を学習させた方が強くなるのかもしれません。ただ 256x20 block の通常のサイズのネットワークだと 25 %混ぜてもほぼ同等に強さでした。

5　対振りと宣言勝ち対策をした dlshogi 互換モデルで出場予定

選手権では dlshogi の互換モデルで出る予定です。ネットワークのサイズは 384x30b。NN の入力には手番、手数を追加する予定です。手数は 320 手を 8 分割して 40 手ごとに全面 1 にします (手数で 8 面)。学習棋譜は Aoba 振り飛車の棋譜を 25 %混ぜます。これで相手が振った時に極端に勝率が上がるのを防げると思います。また持将棋で敵陣に駒が入るように 400 手以降で敵玉に入ってる枚数が 0 枚なら勝率を 0.50 に、9 枚で 0.95 に、する補正をしています。300 手と 250 手以上も勝率が上がりすぎないように小さく補正を。探索勝率を無理やり下げて、dlshogi の学習で勝敗結果と混ぜる割合を 0.333 から 0.500 にしています。あと時間があれば KataGo の NN 構造を試してみるつもりです。

6　定跡は floodgate の AobaZero の棋譜から

定跡は floodgate で走らせている AobaZero の棋譜の局面を 300 万局面ほど探索させて作成する予定です。

7　棋力の推移

図 1 が ELO の推移です。floodgate での測定レートの方が若干高めなのは棋力測定に用いている互角局面集 (24 手まで) には AobaZero が指さない穴熊や振飛車が多数含まれているせいです。局面集を使わずに初手から指させた方が +100 ELO ほど強くなります。

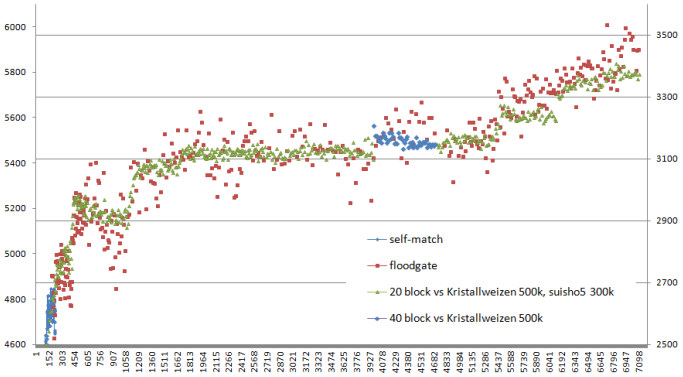


図 1　棋力の推移。右軸が floodgate のレート、横軸は棋譜数 (万)

8　今後

選手権後は 320x30b に移行して学習開始局面集を利用して多様性のある棋譜を作ってみようと考えています。

[*1] https://github.com/kobanium/aobazero
[*2] https://github.com/kobanium/aobazero/issues/54
[*3] http://www.yss-aya.com/furibisha/
[*4] http://www.yss-aya.com/bbs/patio.cgi?read=158&ukey=0

9　6 年で 7100 万棋譜

7100 万棋譜、という膨大な棋譜を 6 年間で生成してきました。棋譜生成に協力していただいてる皆様に感謝いたします。

Kanade アピール文書

山口　奏
2025 年 3 月 31 日

Kanade は一般的に DL 系と呼ばれる類の将棋 AI です。
主に評価関数と定跡を独自で作成しており、探索部はふかうら王を使用させていただいております。

評価関数

Resnet25x320 の標準的なネットワークを使用しています。
序盤中盤に比べ、終盤の評価精度が著しく悪い傾向があります。
教師データは、公開されているデータのみを使用しており、独自で生成した教師データは使用していません。どの教師データが学習に有効的かを研究し、厳選したデータを使用しました。教師データを水増しするために、局面を左右反転させたものも教師データに使用しています。
また、正規化やバッチノーマライゼーションに関する技術などの学習テクニックを多数実験し、最も実験結果が良かった数値やテクニックを採用しています。

定跡生成

4 つの NNUE 系のソフトを使用して対局を行い、対局結果を min-max で親ノードに伝播させる手法で定跡を生成しています。
4 種類のソフトを使用することによって、定跡の穴をより発見しやすくしています。
定期的に人間が定跡の精度を確認し、間違った評価をしていると思われる局面が見つかった場合は、その局面を対局開始局面に設定することで正しい結論に導きやすくしています。