爆裂駒拾太郎
アピール文書

作成日：2024年3月31日

最近の将棋AIは
入玉が苦手らしいですね？

つまり爆裂駒拾太郎が
最強ってコト！？

目次

1. 爆裂駒拾太郎について
2. WCSC33からの変更差分まとめ
3. 主な使用言語・ライブラリ
4. 定跡部
5. 探索部
6. 評価関数部

爆裂駒拾太郎について

- 開発方針
  - 詰将棋が意味ないことを証明する
  - 詰みではなく基本的に入玉宣言による勝ちを目指す
  - 思考に大きな影響を与える他者の創作物は使用しないフロムスクラッチ将棋ソフトとして開発する
    ※指し手生成はライブラリ使用

- GitHub
  - URL:https://github.com/burokoron/Yolts

目次

1. 爆裂駒拾太郎について
2. WCSC33からの変更差分まとめ
3. 主な使用言語・ライブラリ
4. 定跡部
5. 探索部
6. 評価関数部

WCSC33からの変更差分まとめ

- 探索部
  - Null Move Pruningを実装しました
  - Late Move Reductionsを実装しました
  - 置換表を連想配列から動的配列に変更し、エンジン起動直後にメモリ確保するように変更しました
  - 指し手並び替え用配列を動的配列から静的配列に変更しました
  - 置換表を世代管理し、前回の探索結果を活用するように変更しました
- 評価関数部
  - KKP型評価関数(第7世代)からPP型評価関数(第2世代)に変更しました
  - 局面の評価を差分計算するように変更しました
- 評価関数学習部
  - 学習に使用するライブラリをTensorFlow 2.5.0からPyTorch 2.1.1に変更しました
  - 学習データ生成部をJITコンパイルするように変更しました
  - スパーステンソルを用いて学習するように変更しました
- 学習棋譜生成部
  - 千日手手順は選択しにくくなるように変更しました
- その他
  - 評価関数ファイルが読み込まれていない場合のみ、isreadyコマンドで読み込むように変更しました

目次

1. 爆裂駒拾太郎について
2. WCSC33からの変更差分まとめ
3. 主な使用言語・ライブラリ
4. 定跡部
5. 探索部
6. 評価関数部

主な使用ライブラリ(対局時に使用)

- プログラミング言語
  - Rust 1.73.0
    - 対局エンジンの作成に使用

- ライブラリ
  - yasai 0.5.0 [1] ベース
    - Rustで利用できる将棋ライブラリ
    - 指し手生成、局面管理に使用
    - 使いやすいように一部改変

[1] yasaiのURL : https://github.com/sugyan/yasai

主な使用ライブラリ(開発時に追加使用)

- プログラミング言語
  - Python 3.11.5
    - 学習データおよび評価関数の作成に使用
- ライブラリ
  - cshogi 0.7.5 [2]
    - Pythonで使用できる将棋ライブラリ
    - 学習データおよび評価関数の作成における局面管理に使用

[2] cshogiのURL : https://github.com/TadaoYamaoka/cshogi

目次

1. 爆裂駒拾太郎について
2. WCSC33からの変更差分まとめ
3. 主な使用言語・ライブラリ
4. 定跡部
5. 探索部
6. 評価関数部

定跡部

- 学習データの生成時の副産物として作成
- 学習データの棋譜から各局面の勝ち数、負け数、引き分け数を算出し、Thompson Samplingアルゴリズムに基づいて指し手を選択する
- ただし、訪問数が10未満の局面となるような指し手は選択されない

R	時間	深さ	ノード数	評価値	読み筋
11				48	▲5八金(69)
					6i7h move_rate=48.83% black_wins=250 white_wins=278 draw=22
10				48	▲7八金(69)
					5i6h move_rate=48.92% black_wins=583 white_wins=564 draw=44
9				48	▲6八玉(59)
					7i7h move_rate=48.98% black_wins=191 white_wins=215 draw=17
8				48	▲7八銀(79)
					1i1h move_rate=49.26% black_wins=323 white_wins=335 draw=29
7				49	▲1八香(19)
					9g9f move_rate=50.02% black_wins=325 white_wins=342 draw=32
6				50	▲9六歩(97)
					2h4h move_rate=50.25% black_wins=792 white_wins=747 draw=72
5				50	▲4八飛(28)
					5i5h move_rate=50.86% black_wins=816 white_wins=782 draw=60
4				50	▲5八玉(59)
					4i3h move_rate=51.43% black_wins=242 white_wins=271 draw=24
3				51	▲3八金(49)
					4g4f move_rate=51.45% black_wins=517 white_wins=508 draw=38
2				51	▲4六歩(47)
					2g2f move_rate=55.06% black_wins=189809 white_wins=146995 draw=46527
1				55	▲2六歩(27)

目次

1. 爆裂駒拾太郎について
2. WCSC33からの変更差分まとめ
3. 主な使用言語・ライブラリ
4. 定跡部
5. 探索部
6. 評価関数部

探索部

- 探索アルゴリズム
  - αβ探索
  - 反復深化探索
  - 静止探索(駒取り、王手回避のみ最大3手延長)

- ムーブオーダリング
  - Hash Move (前回の探索結果も活用)
  - Killer Heuristic
  - MVV-LVA
  - History Heuristic (piece-to)

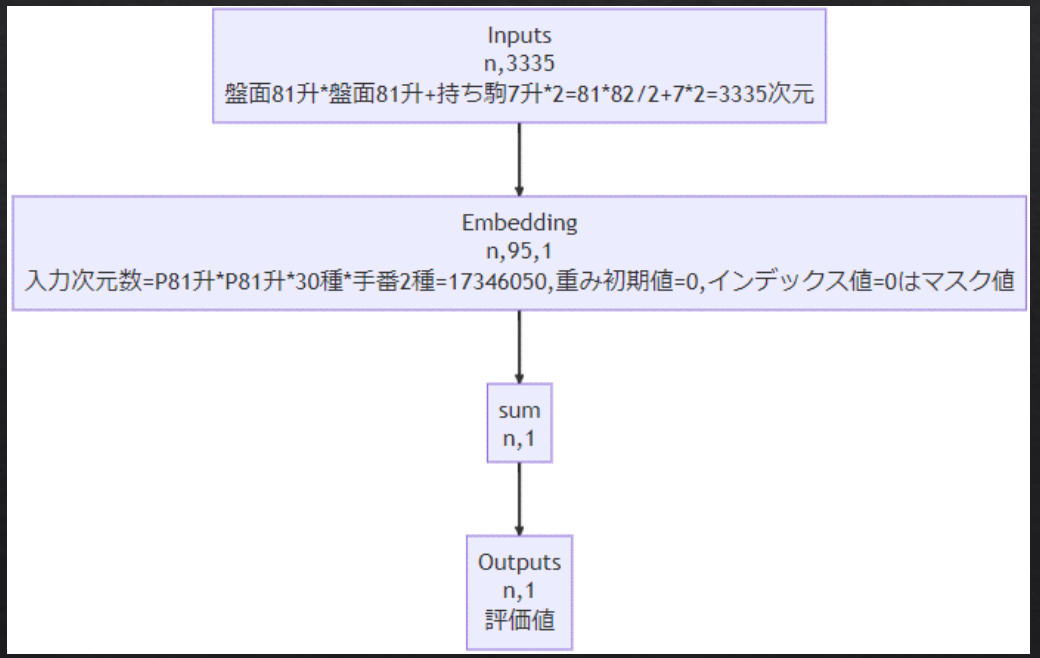
- 前向き枝刈り
  - Mate Distance Pruning
  - Null Move Pruning
  - Late Move Reductions

目次

1. 爆裂駒拾太郎について
2. WCSC33からの変更差分まとめ
3. 主な使用言語・ライブラリ
4. 定跡部
5. 探索部
6. 評価関数部

評価関数部(1/2)

- 方針
  - 詰みではなく入玉を目指すような評価関数を作成する
- 学習時と推論時の違い
  - 推論時は1手進んでも局面があまり変化しないため、変化した特徴量のみ差分計算する
- アーキテクチャ
  1. 入力層(2駒関係特徴量)
     - 3335次元
  2. 埋め込み層(2駒関係特徴量)
     - 17,346,050次元⇒1次元
  3. 出力層
     - 合計



評価関数部(2/2)

- パラメータの調整
  1. 探索部により強制的に相手を詰ますと負けになるようにし、
     入玉でなければ勝てないようにした爆裂駒拾太郎を作成する
  2. 通常の爆裂駒拾太郎と対局を行い棋譜を作成する。
     棋譜をばらけさせるために1度以上出現した局面については、
     勝ち数、負け数、引き分け数を算出しておき、
     Thompson Samplingアルゴリズムに基づいて指し手を選択する
     ※ただし、引き分け数が出現回数の半分以上を占めている場合は選択率を下げる
  3. 棋譜のうち1で作成した爆裂駒拾太郎の手番の局面と評価値のみで
     評価関数の学習を行う
  4. 3で作成した評価関数を新たな爆裂駒拾太郎とし、1へ戻る。
     ただし、2では過去の評価関数との対局も行う。
     このときの対局割合は勝率の高い評価関数を小さく、
     勝率の低い評価関数を大きくする
- 初期値は小駒100点、大駒500点とし、敵陣に近いほど1点加点(最大9点加点)
  する評価関数とする