#28 世界コンピュータ将棋選手権(WCSC28)
elmo

瀧澤 誠 May.12 2018

はじめに

- 最初はマジメに作っていたのですが、挫折しました
- なんか、ごめんなさい。

✓音付き(動画)

https://youtu.be/jiYmJRpiVZM

では、本編どうぞ。

elmo

- 公開されているコンピュータ将棋ライブラリを利用して開発。
  - 主に評価関数部に変更を加えています。
  - 最先端ライブラリからどうやったらより強くなるのか、を探っています。

- elmoの評価関数、定跡など (google driveにて公開)
  - https://drive.google.com/drive/folders/0B0XpI3oPiCmFOG1RX1FidlpqM00

WCSC#27 (2017)

elmo = Ponanza + 激指

elmo(第27回世界コンピュータ将棋選手権版(2017/05))

- ライブラリ:Apery(浮かむ瀬)からの実質的な変更は1点のみ。
  - 激指(WCSC26※): 自己対局での勝ち負けを教師とした強化学習…①
  - Ponanza(浮かむ瀬 他色々): 自己対局での深い読みの評価値を教師とした強化学習…②
  - elmo(WCSC27※): ①と②両方を教師として用いる

※WCSC:世界コンピュータ将棋選手権の略称。WCSC26→2016年の世界コンピュータ将棋選手権

式など詳細は以下の説明が詳しいです。
http://tadaoyamaoka.hatenablog.com/entry/2017/05/06/130935

①の方がより良い手法と思っていたのですが、単体では②の方が強いです(少なくともKPPでは)。

評価関数の生成は上記の通り。対局時にはやねうら王を利用しました。

なので、実装上は、elmo = Apery + やねうら王 （+ 激指） となります。

WCSC#28 (2018)

elmo = Ponanza + 激指

+ Bonanza(NineDayFever)

elmo(第28回世界コンピュータ将棋選手権版(2018/05))

- elmo(wcsc27)①②＋Bonanzaメソッド③
  - 激指(WCSC26)： 自己対局での勝ち負けを教師とした強化学習…①
  - Ponanza(浮かむ瀬 他色々): 自己対局での深い読みの評価値を教師とした強化学習…②
  - Bonanza(NDF): 任意局面での兄弟手の優劣を教師とした学習…③

Bonanzaメソッドの適用：

強化学習の自己対局時の指し手(と評価値)を教師として、
- 任意局面A→(教師の指し手) →(静止探索)→局面B
- 任意局面A→(その他の指し手)→(静止探索)→局面C

とすると、局面Bの評価値 ＞局面Cの評価値 となるように要請(ついでにBは教師の評価値に近付ける)。

効果：

(兄弟局面の数だけ教師が増えるので)
事前に作成が必要な教師局面数が数分の1に。

ってのは嘘で

弱かったので破棄

(見所はあったし理屈上も良く出来ていた)

こういうのはよくある

(これでうまくいくなら誰かもうやってる系)

→ と思ったんですがやった人いなかったっぽいです

KPPの学習むずい

KPPの学習むずい

- 最適化/ノイズとの闘い
  - 最適化出来ていない状態
    - KPPの特徴空間が広く(4億パラメータ)、教師データが一部の局面に偏っており、満足に学習出来ていない特徴量が多い。
    - 学習する際に与える情報として、「勝ち負け」「深く読んだ時の評価値」とあるが、1局面でKPPだけでも1406パラメータあり、1局で200局面程度あるのでどの手が効果的だったかはよく分からない(大量のデータでなんとかしてるのが現状)

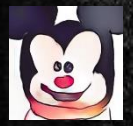
    → この辺があってボナメソ入れてみた。
    - 最適化出来ていないのであれば、「学習率を下げる」「バッチサイズを大きくする」「更新を一部パラメータに限定する」辺りで改善出来そうだが…(大会版では大体この辺やってます)
  - 評価関数マージ(評価関数を足して2で割る)の破壊力
    - 大変都合の良い手法
      - 最適化出来ていない部分同士が打ち消しあい、最適解に近付く
      - 逆に言うと、マージ後の評価関数は「うまく出来ている状態」なので、ここからの改善はより難しい

解いている問題はある程度見えてきたが、根本的な解決策が見えない(そもそもKPPであるべきか？)

悔しいけど止めとく

凄くやったけど

もういいじゃんKPPとか。
誰かが何とかしてくれるに違いない



さあ定跡だ


WCSC
さあ
elmo
定跡だ

将棋は良いらしい

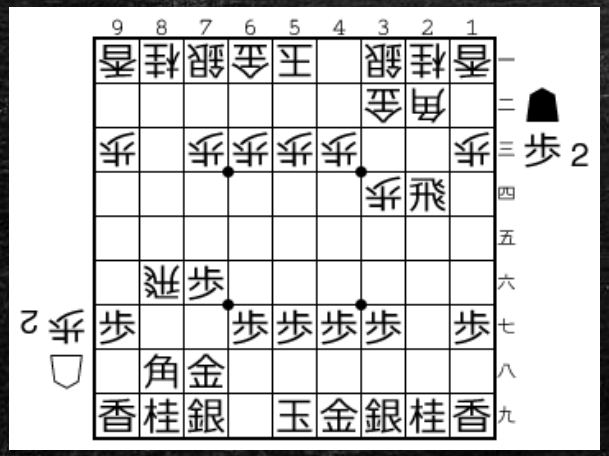
- チェスは定跡勝負を避けるために、定跡が指定されるとか。
  - 定跡勝負は面白く無いけど、勝てるようになってルール上問題無いならやらないと駄目。
  - 割と真面目に考えられていない分野なので作ってて面白い

もし、わしのみかたになればせ　　んぶんをおまえにやろう。

※図はイメージです

コンピュータ将棋における定跡

- 特定の局面が現れたら思考することなく、事前に決められた着手を応答する。



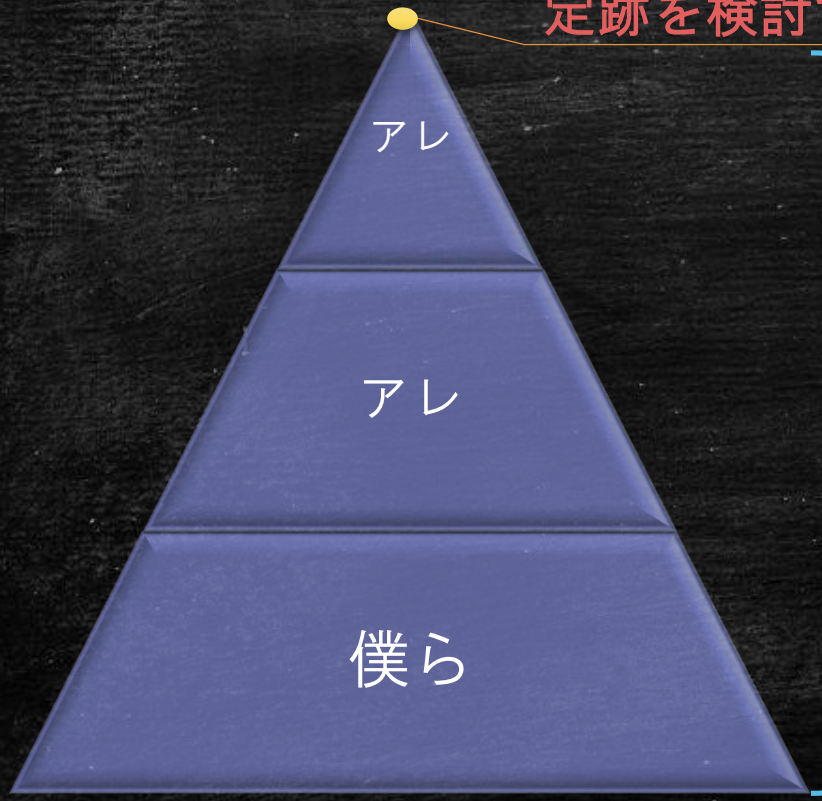
通常の思考：自分がこう指して、相手がこう指して、
20手先の局面がこうなって評価値が+100で
３四飛！　←時間がかかる

定跡　　：この局面は３四飛！　←考えないので早い

定跡手は予め時間をかけて考えた着手の場合、対局時の持ち時間を削減したり、
よく検討された着手だと質の面でも向上が期待できる。準備が面倒だけど。

コンピュータ将棋における定跡

- 定跡の検討が必要な局面は氷山の一角に過ぎない　→なんかやれそう



※図中の文字はつい書いてしまった文字列で説明とは関係ありません。

定跡生成と評価関数最適化で解いてる問題は同じ

- 機械学習での識別と回帰くらいの違いでしかない
  - “良さ”を実数で与えるのが評価関数、”良さ”が最大となる着手を与えるのが定跡

  → 要は評価関数の手法はほとんどそのまま使える。

  → じゃあelmoの手法をそのまま適用しよう！
- 違うとこ
  - 定跡は局面そのものが特徴量。言うなれば(KKPPP…→KKP38)
    - 全局面を定跡で表現できるようにすると、異次元な量のメモリが必要。地球の砂粒の数より遥かに多い
    - でもそんなに必要無い。

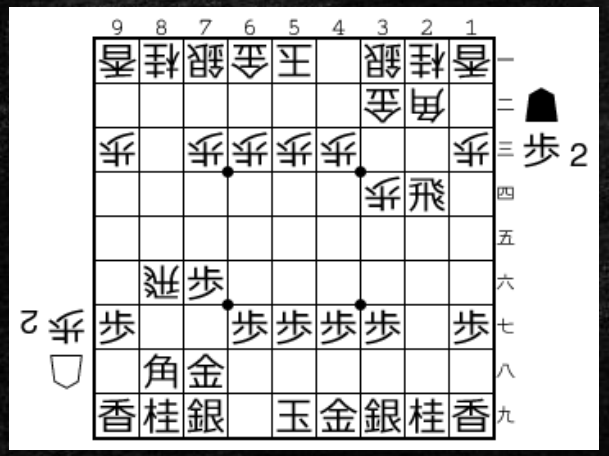
    → 100万局面も網羅出来ればおなかいっぱい。1手100バイトとしても100MB程度。

強化学習での定跡生成

- 定跡生成でも自己対局。勝てば重用、負ければ優先度を下げる。
  - 対局毎(※)に更新。(※効率の問題で12局毎としている)
  - 勝ち負けだけでは、サンプルが少ない場合や対戦相手の癖で安定しない

  → 探索時の評価値を用いて安定化。(思考の順序は違えど、評価関数と同じ)

  - 1局面を構成する特徴量が1406個あるKPPと比べて定跡では1局面1個であり、特徴量に対する勝ち負けの報酬がより適切に行えるメリットも。



3四飛は100点
2八飛は97点
→定跡は3四飛

3四飛で勝った！
→ 3四飛は105点
3四飛で負けた！
→ 3四飛は95点
→今度から2八飛(97点)にしよう！

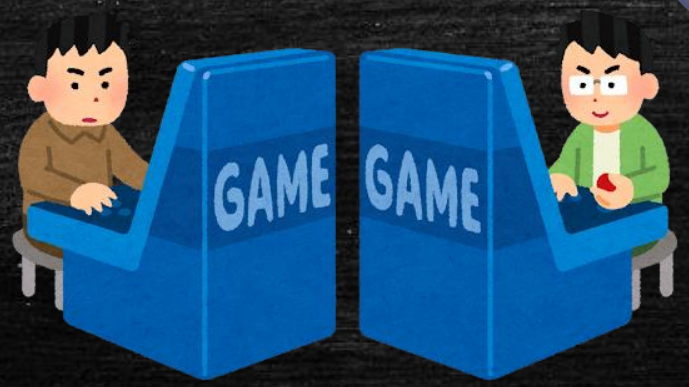
局面の偏り対策

- 自己対局の場合、概ね同じような展開となる
  → 同じ局面で過剰に勝ったり負けたりするため、評価も過剰となる。
  → 複数の評価関数、定跡を対戦相手として利用する。

elmo
・評価関数(固定)
・elmo定跡(都度更新)

×

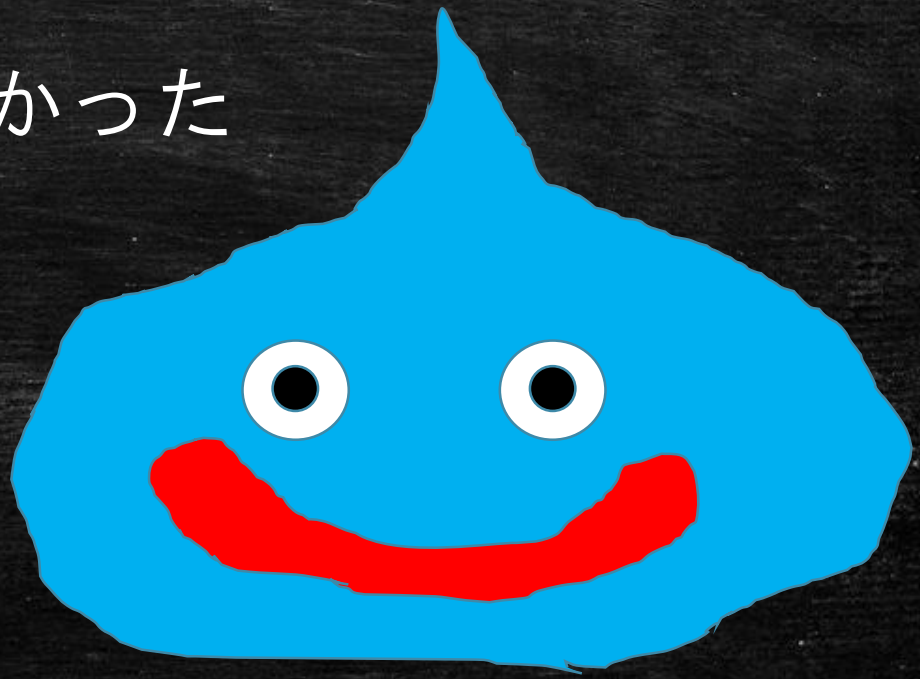
色々な評価関数
色々な定跡



で

こっちも破棄

正直、すまんかった



- ある程度勝率が悪くなると定跡から外れ、自分で考えるのですが、探索で発生するランダム要素だけだと、同じような手ばかり指すようです。
  → 意図的に他の手を探す必要がありそうです。

結局、多腕バンディット問題の手法になるのか...

結局定跡は、
✓ 従来の対局による生成(狭く深い定跡)
✓ やねうら王の定跡生成(広く浅い定跡)
を組み合わせて生成しました。

※後者は主に定跡外し対策(序盤が広い形)

こんな感じで

よろしく！