# みざうら王 with お多福ラボ PR 文書

みざうら王チーム

## 特長

・dlshogi 互換エンジンを作った(dlshogi の model ファイルがそのまま使える)
　本家 dlshogi より優れている点もいくつかある。

・詰将棋ルーチンとして df-pn を改良したものを実装した。メモリと時間が無限にあるなら、どんな長編でも間違わずに解ける。(実際にはメモリと時間は無限にないので実際は 50 手を超えるものは現実的には解けないこともあるが…)

・ソースコードは GitHub で公開しているのでそれを見て欲しい。
　　https://github.com/yaneurao/YaneuraOu

## 学習手法

学習はいわゆる強化学習なのだが、どのような教師をどういう配分で混ぜれば良いのかは知られていない。
・序盤と中終盤の局面の割合は？
・戦型はどれくらいバラけているのが好ましい？
・序盤はどれくらいバラけているのが好ましい？
・入玉将棋の割合はどれくらいが好ましい？
・どれくらいの po(playout)で対戦させた棋譜が好ましい？

そこで、いま公開されている AobaZero、dslhogi の教師局面、それから floodgate の R3600 以上の棋譜、水匠の入玉絡みの棋譜、自己対局で生成した棋譜(800po,1600po,2400po,…)など複数の教師データを用意して、それぞれを 1 本の stream とみなし、1epoch 分の学習に用いる局面数は 500 万と固定化して学習させることにした。

つまり、AobaZero : dlshogi : 水匠 ＝4:5:1 のように書けば、その配分から成る 500 万局面分の一時ファイルが生成される。(無論、前回用いた続きの箇所から)　この一時ファイルから学習する。(ようにした)

epoch が進んだときに、最適な配分は変化していくものと思われる。

しかし、それを試すにも 10 epoch ぐらいは学習させて、対局させるなり loss(損失関数の値)や accuracy(検証用データとの指し手一致率)などを比較しないといけない。そのような比較をしていると手戻りするので時間がもったいない。

かと言ってそのような検証なしだと、悪い配分のまま学習させてしまうことになるので、本来その DNN の architecture の持つポテンシャルまで至らない。

そこで我々が考えたのは、**天国と地獄メソッド**である。

PC は複数台所有しているので、条件(教師の配分)をそれぞれ変えて同時に学習を進める。

そのなかで一番成績の良かったものを生かして(**天国**)、残りの PC の学習結果は捨てる(**地獄**)。

一番成績の良かった PC のデータを残りの PC にもコピーする。そしてまた条件をそれぞれ変えて同時に学習を進める。

それを繰り返す。

こうすることにより、手戻りなしで学習ができる…はずである。

これを書いているのは 3 月 23 日。正直、当日までに学習が終わるかどうかわからない。時間が足りなさそうなら、GCP で A100 を借りて学習を回す予定である。

// この PR 文書は選手権後に差し替える(かも知れない)

チームBarrel houseのアピール文書＠第31回世界コンピュータ将棋選手権

Barrel houseとは岡山の駅前にあるビアバーです．元々チームメイトを求めさすらって行きついたところです．マスターに許可頂いたので名前をお借りしました．その後，メンバーが変わって当初の方向性とは全く違って来ましたが，まぁ一度出した名前を変更するのもアレなのでそのままです．

プログラム名：白ビール
第28回のHefeweizenが濁った白ビールでしたが，第29回ではKristallweizenとフィルターでろ過した透き通った白ビールでした．その後Hefeweizenに戻すという形を取りました．しかしながら，欧州のチェスサイトでは命名由来から説明頂いているにも関わらず国内では白ビールとしか読んで頂けないので，もう白ビールでいいやってことです．

チームの特徴
フレッシュなチームを自認してますがどうやらおっさんチームのようです．本大会は体力勝負ではないので各方面に様々な知識や技術・勘などを働かせて今年も決勝に残れればいいかなぁと思っています。諸事情でメンバーが多忙なため具体的な策はこれから詳細を詰めていくところです．（前回も前々回もそんなアピール文だった気もしますが）

評価関数
昨年度の計測でもトップチームと劣らないものであることを確認しました．（詳しくはblogにて）
今後の調整等は未定です．

使用マシン
今年もノートパソコンを中継にクラウドの力をお借りする予定です．一昨年と昨年は同じ仕様のものを用いましたが，クラウドの方も今年は若干様子が変わっているようでベンチマーク等も未だなので具体的なことは全く決まっていません．正直あまり予算がないのですが，マシンパワーだけで負けるのもアレなので予算内で一番速いマシンを借りようと考えています．

クラスタリングについて

第５回電王トーナメントで御披露目をしたshotgunシステムの進化版であるMulti Ponderのクラスタリングを用います.

制作メンバーが本年は外れておりますが，既に実装は文書化され多くの類似実装が選手権でも見られますので改良して用いるのは問題ないと考えます.

# たぬきち
# 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 安達瞭

# 目次

- たぬきち
- 評価関数
- 定跡
- 使用ライブラリ

# たぬきち

- 元ネタは『どうぶつの森』に登場するキャラクターです
- 富豪的プログラミングを意識しています

# 評価関数

- NNUE 評価関数 halfkp_256x2-32-32 を使用しています
- ランダムパラメーターからの学習時、および強化学習時に、進捗度を用いた Weighted Loss を使用しています
  - 評価値のスケールを過去の評価関数より小さくしています

# 定跡

- floodgate の棋譜と自己対戦の棋譜から生成しています
- 勝率と評価値の両方を考慮し、取捨選択しています

# 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る。
    - 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。
- 水匠2
  - 評価関数をランダムパラメーターから学習させる際の、学習データの生成に使用している。
    - レーティングが十分に高いため。
- tanuki-
  - 学習データの生成に使用している。
    - 過去に開発した資産の再利用のため。

よろしくお願いします

elmo アピール文書 2021/3/29

１．サマリ

コンピュータ将棋開発 → 「高ノード対局用の定跡生成」と位置付ける。
定跡を抜けた後はある程度の棋力と、残り時間の差(と定跡による優位)で押し切れる
と期待して止まない。

今後、評価関数以外の開発ポイントとしてあっても良いと考えている。

２．戦略

実際のところ、時間をかけて深く局面を読むよりも、
ある程度の棋力で20手30手進めて優位となる基準で指し手を選択した方が
消費時間だけでなく、(経験的に)質の面でも優位と考えている。

定跡として必要となる膨大な局面数についても
棋力が上がるにつれて序盤進行は限定されつつあり、
何とかなるのではと思える程度にはなっている。

50手近くまで定跡進行で進められれば、それだけで自分としては成功である。

３．定跡の生成

基本的には従来のelmoの定跡生成法と同様で、あまり開発者らしいことはしていない。
・高ノードで対局し、不利無く勝ち越したものを採用する
・負けた場合は検討モードで修正し、採用する
　その際、既存の定跡が不適と判断されればそこも修正する

手動部分が大きいことを除けば、1局だけで採用するため
対コンピュータ将棋としては効率的な作成手法である。
(比較的少ない計算量で実戦的な(的中率の高い)定跡が作成可能)

※ 評価関数は前回のelmoを利用して生成

定跡の生成時に評価値や勝率等は利用していないため、
NNUEではなくディープラーニング等の手法を利用することも可能である。

４．その他改善(評価関数等)

残念ながら強くなっていない。

・学習方法の変更
評価値と勝率の関係を見ていると、評価関数によっては
勝率の高い部分/低い部分で一定程度外れ値を取るものが確認できる。
(学習用の棋譜から確認)

仮定となるが、
「学習棋譜での勝率と、評価値から算出される勝率との間にモデル上のギャップがあり、
　評価値→勝率の変換係数が不適切」なのではないかと考えた。
変換係数を学習棋譜から算出されるものと一致させることで
強制的にモデルに合うようにした。

とはいえ、既存のNNUEのパラメータを流用するために他のパラメータも修正しており
結果的に既存とほぼ等価な変換となった。が、一致率等の指標は改善した。
何か効果があったかもしれないが、残念ながら気休め程度である。

・学習棋譜生成速度の改善
やねうら王は棋譜生成用のバイナリを作る際には、専用のコンパイルオプションを用いるが、
棋譜生成速度が遅いためtournament(対戦専用)で棋譜生成をするようにした。
約50%生成速度が向上した。

・定跡ponder
ponder中に定跡にヒットしても相手時間中に思考するようにした。
定跡として不成を指せるようにした。

・stockfishの探索部の更新は取り込み予定

５．やるき(低下中)

12月くらいまで真面目に定跡作っていたが、評価関数側が全く良くならずテンションが下がる。

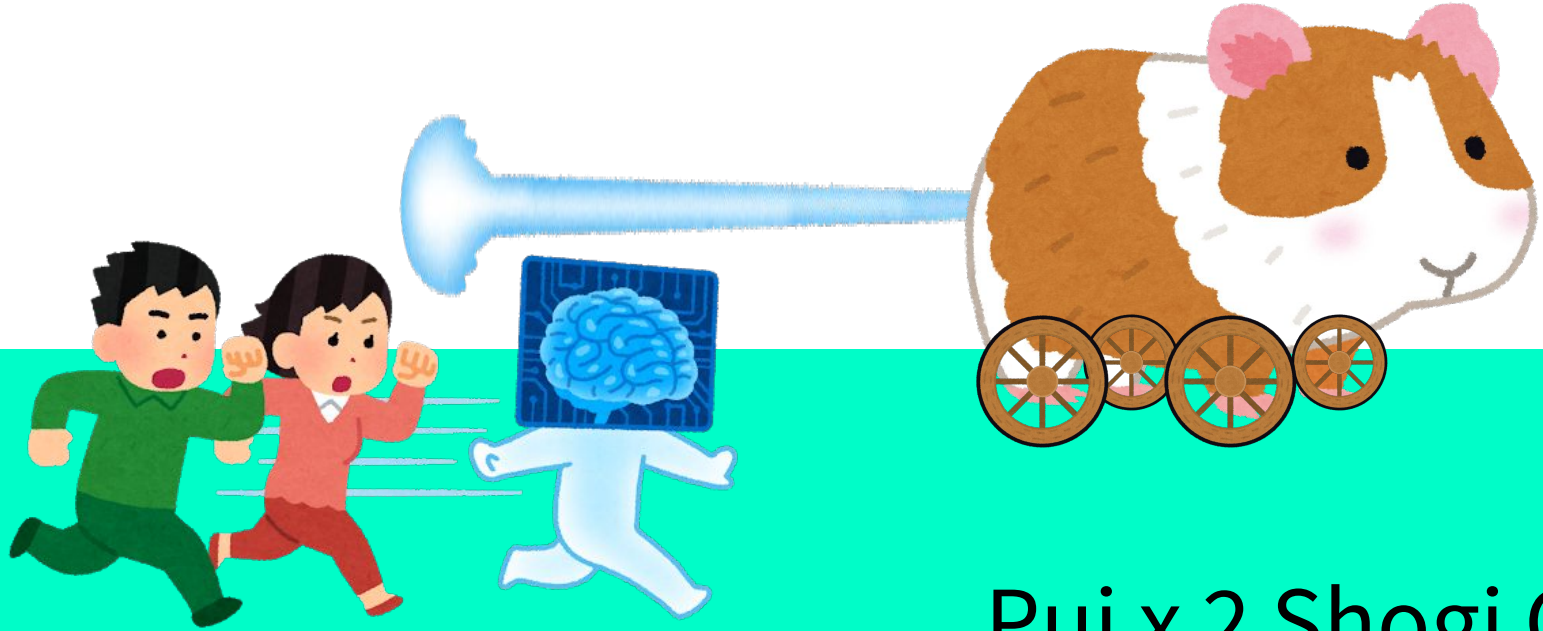
ちなみに先手番定跡は殆ど作っていなかったりします。

６．使用ライブラリ

・やねうら王：主に利用(一部改修して利用)
・水匠シリーズ(2～3改)：評価関数のベース＆学習棋譜生成に利用予定

MolQha- のPR文章

# 変態将棋で勝ちたい



Pui x 2 Shogi Club

本研究で目指すもの

# できるだけ簡単に

# 変態将棋を指して

# (できるだけ)勝つ

# 人間にとっての変態将棋

## プロの対局では滅多に出てこないもの


鬼殺し


嬉野流


一間飛車

画像はwikipediaより引用

# AIにとっての変態将棋

AIが普段指さないもの……

＝角換わり、矢倉、雁木、相掛かり以外


一手損角換わり


横歩取り


振り飛車

画像はwikipediaより引用

# 特定の戦型を指させる方法は2つある



## 定跡　　評価関数

# AIの定跡作戦


「この局面ではこの手を指す」
という情報が集まった巨大なカンペ

戦型調整、時間節約、
嵌め手誘導に活用！

# 定跡の長所 1/2：（最初は）楽


お気に入りの棋譜
を入れるだけでも
最低限動く

# 定跡の長所 2/2 :（ハマれば）強い


第五回将棋電王トーナメント
画像はねとらぼ様より引用

## 定跡の力で勝った試合も多々ある

【定跡を抜けたら既に不利だった被害者の会】
第五回将棋電王トーナメント
ponanza，elmo，やねうら王

第四回将棋電王トーナメント
技巧

第26回世界コンピュータ将棋選手権
nozomi
・・・・・・他たくさん

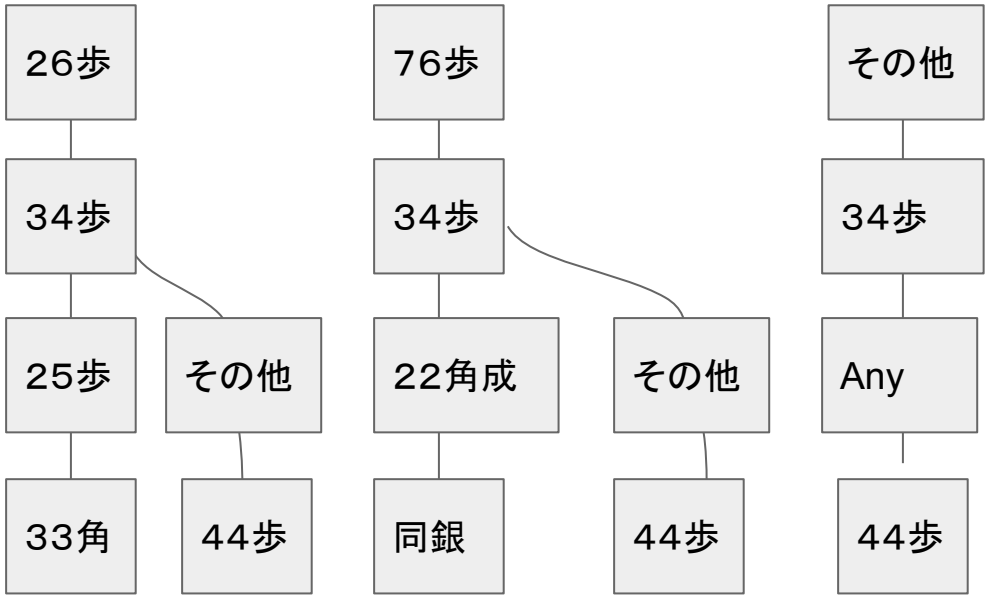
# 定跡の短所 1/2：予想外の手に弱い


例：定跡を外された振り飛車

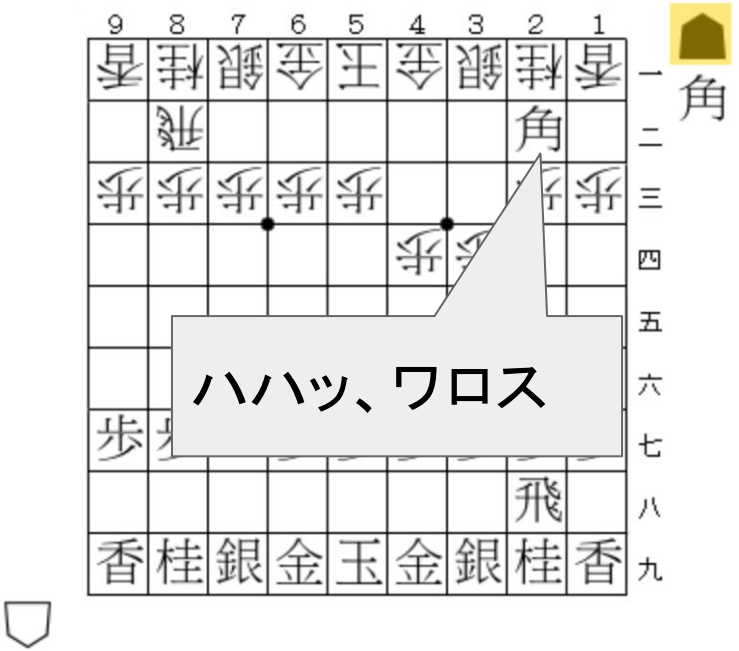
広い指し手に対応した定跡が欲しい

# 定跡の短所 2/2：複雑な戦型が組めない


## 例：後手四間飛車



# 定跡の短所 2/2：複雑な戦型が組めない




例：後手四間飛車

▲76歩△34歩▲22角不成△44歩

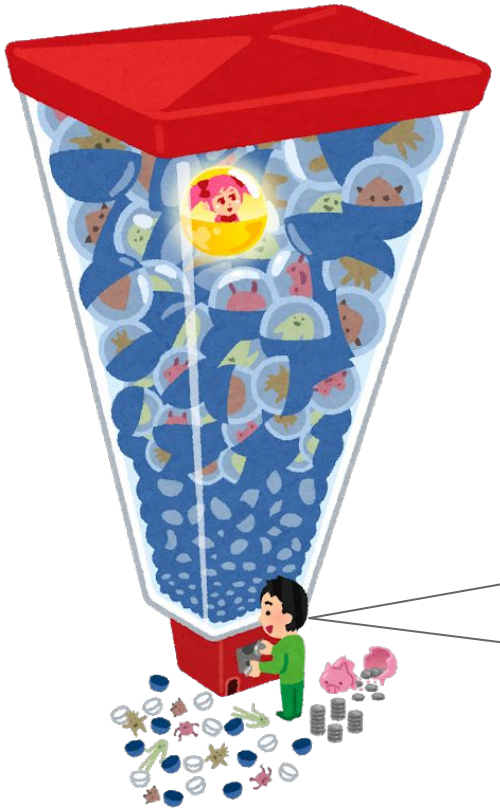
死

# 評価関数の調整方法(特定戦型向け)

Q:振り飛車の勝率は？


勝率を都合よく書き換えた上で学習させる

# 評価関数の調整の短所



大体の場合、元の関数より弱くなる

illqha4.0をオリジナルの評価関数として、プロの振り飛車の棋譜の勝率が7割になるように調整して、学習率は0.01で1 iteration回したものを混合比1000:1で線形結合して...

# MolQha-の作戦


定跡と評価関数のいいとこどりをしたい

# 序列つき定跡


定跡で可能な限りカバー

定跡が外れたら追加学習した関数（弱くても良い）を利用

所望の戦型になったら適当な強いAIに指し継ぎ

# 序列つき定跡のキモ


ここにできるだけ充実させる

追加学習関数にはできるだけ仕事をさせない

変態定跡の自動育成

# 変態定跡の自動育成


追加学習関数と強い関数とで戦わせる（2割ちょいは勝てる）


勝った棋譜を定跡に組み込む

# MolQha-の構成まとめ


棋譜を集める

追加学習をする


作った関数を戦わせる


勝局で定跡を作る

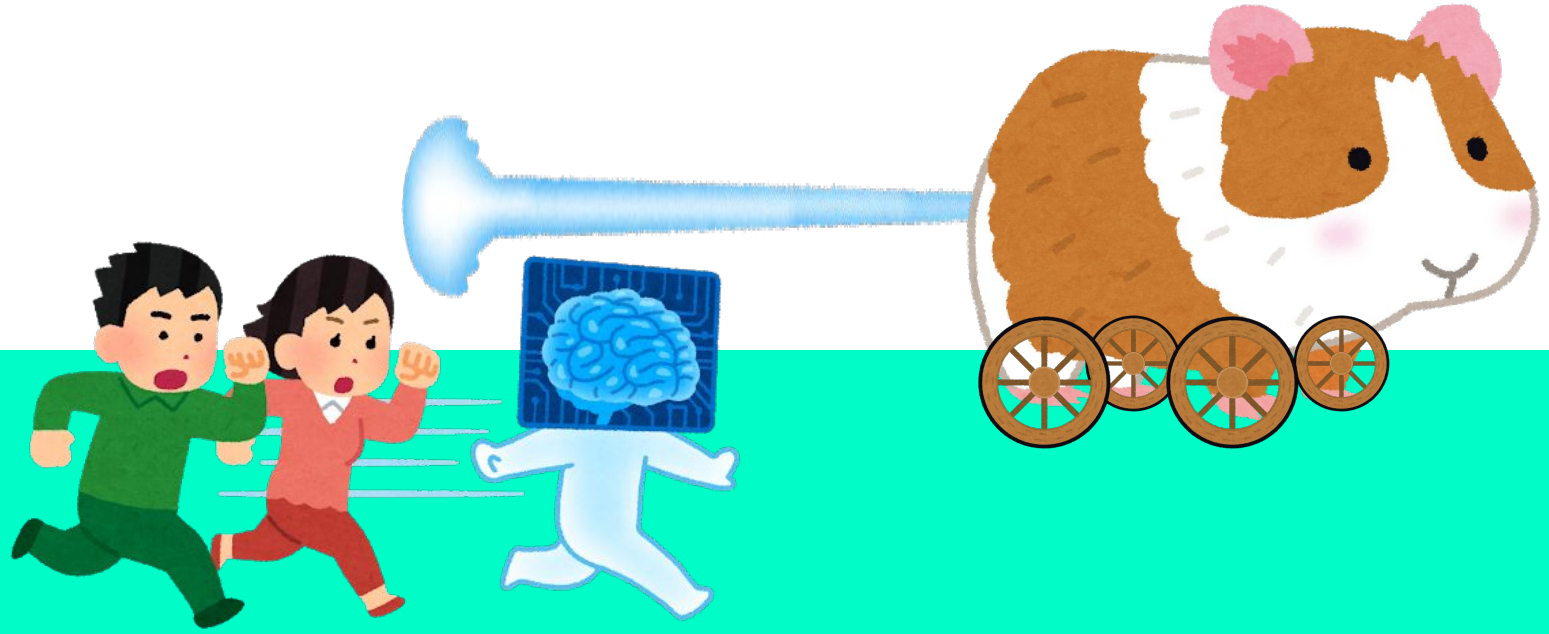

序列つき定跡でエンジンを構築

# MolQha-の工夫


追加学習ルーチンの品質確保（10局程度の棋譜で動作するようにする）

徹底的な自動化（棋譜を入れただけで定跡生成から勝率測定まで行う）

多分youtubeにpr動画がでますの
でそちらもよろしくお願いします



第31回 世界コンピュータ将棋選手権

# PAL

## アピール文書

山口 祐

# PALの概要

- pal/ﾊﾟﾙ/【名】<話> 仲間、友だち （例）a pen ―
- 前回大会時の工夫
  - マルチエージェント（Population Based Training）
  - 評価/探索/定跡を同時に学習、MCTS探索の実装と検証

# 開発者

- 山口 祐（@ymg_aq）
- 囲碁・将棋プログラム等を開発 (Github)

## PALの特徴

- 2021年度より**NHK杯将棋トーナメント**の評価用AIとして採用
- **Deep Learning**を使う
- 前年のPALで生成済の評価値・候補手データを事前学習
- 探索部を書き始めたので、大会までには動くよう間に合わせたい

## 使用ライブラリ

- **技巧**: 探索部以外の大部分で使用

# 名人コブラ アピール文書

松山洋章

# 概要

dlshogiをベースに、評価関数の学習と
定跡作成に工夫を加えています。

# 評価関数

Prioritized Experience Replay を簡易的に実装して学習します。苦手な局面をより頻繁に学習する方法です。

参考:
https://tadaoyamaoka.hatenablog.com/entry/2021/02/25/220351

# 定跡

DL系の探索方法であるMCTSと他の方法を組み合わせて定跡を作成します。

定跡で積極的に勝ちにいくわけではなく、序盤で不利にならなければ良いという考え方です。

# 使用ライブラリ

- **dlshogi**

  対局エンジン、評価関数学習用

  ビッグウェーブに乗りたいため

- **やねうら王**

  定跡作成用

  解説等が充実しているため

- **Apery**

  学習データの生成用

  OSSとしての歴史が長いため

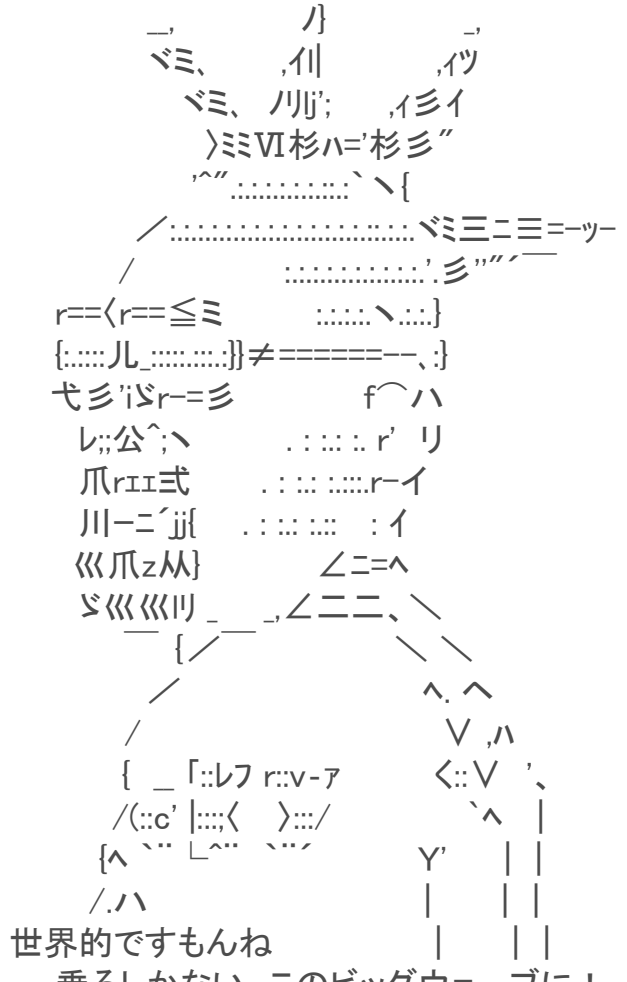
# ソフト名の由来

劇団鋼鉄村松

「二手目8七飛車成り戦法」

登場人物より

参考:
https://stage.corich.jp/stage_main/23036


世界的ですもんね
乗るしかない　このビッグウェーブに！

# Team Novice 2021

# About

- 2015年から開発されている将棋プログラム
- 2018年からインテル®コラボレーションセンターの協力を得て、チーム開発に移行
- 第29回世界コンピュータ将棋選手権で独創賞を受賞
- 2020年からAlphaZeroを参考に開発を行う

# Members

- 熊谷啓孝　　【チーム代表】
- 中屋敷太一　　【エンジニア・主任開発者】
- 矢内洋祐　　【ハードウェア担当】
- 堀越将司　　【最適化・プロファイル担当】
- 笹井雄貴　　【棋力担当】
- 幅野莞佑　　【機械学習等担当】

# Implementation

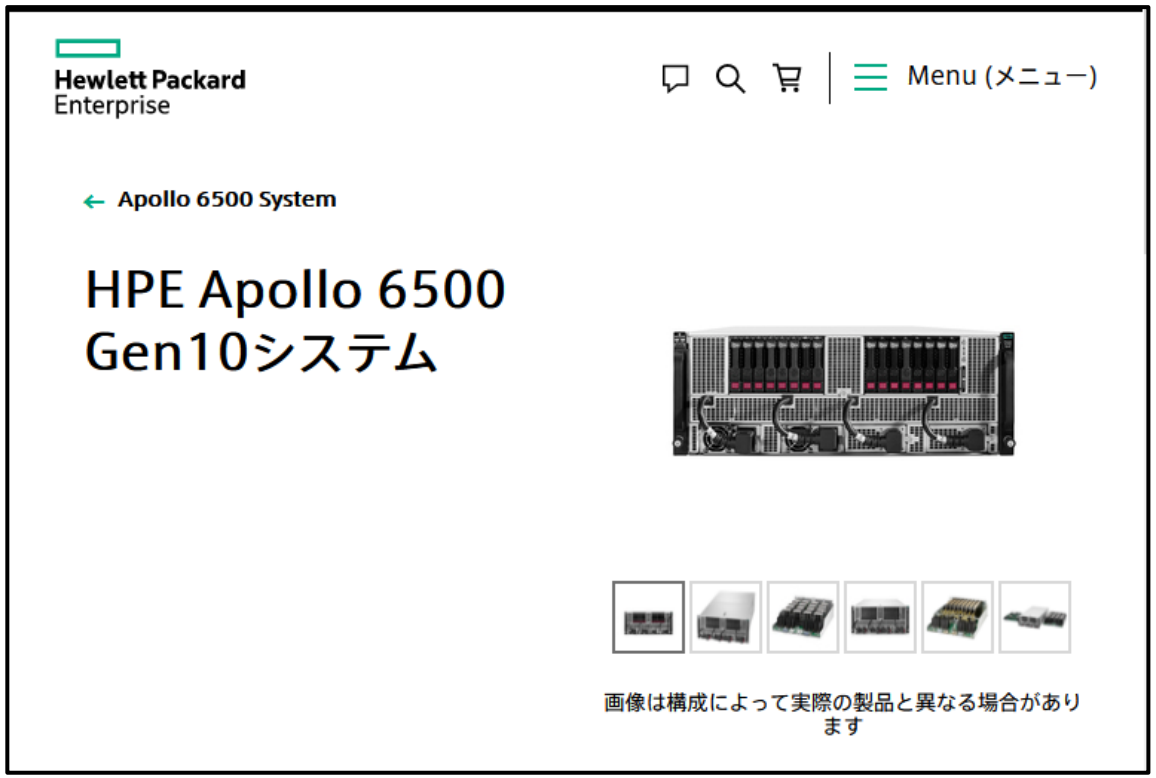
- AlphaZeroを参考にした実装
- TensorRTによる最適化

# Hardware

- HPE Apollo 6500 Gen10システム　を使用

  https://www.hpe.com/jp/ja/product-catalog/servers/proliant-servers/pip.hpe-apollo-6500-gen10-system.1010742495.html

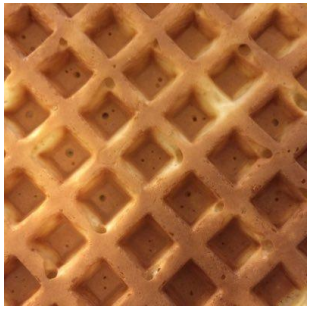
  - 第２世代インテルXeonプロセッサー
  - NVIDIA®V100 x 8




Hewlett Packard Enterprise 様 HPより


HoneyWaffle
第31回世界コンピュータ将棋選手権　アピール文書
開発者　渡辺 光彦

# 開発者



氏名: 渡辺 光彦

職業: プログラマー

棋力: 将棋ウォーズで2級、ぴよ将棋でR900-1000程度の振り飛車党

Twitter: @shiroi_gohanP (https://twitter.com/shiroi_gohanP)

ニコ生の電王戦をきっかけにコンピュータ将棋を始める。

将棋連盟Liveやニコニコ放送、AbemaTVの将棋中継が好き。

★note始めました！ → https://note.com/honeywaffleshogi

★文春オンラインのインタビュー記事 → https://bunshun.jp/articles/-/14921

# HoneyWaffle (ハニーワッフル) 名前の由来

・四角いワッフルは将棋盤と似ている

・ゆるふわスイーツ的なスナック感覚の軽さを表現

　元々タブレット向けに開発していたので物理的に軽いこと、振り飛車の軽い捌きができるようになるといいなという想いから命名しました。

※表紙やTwitterアイコンのワッフルはうちで焼いたものを使用しています。

以下のリンク先で出せるものは公開しています。使い方がおかしいのはいつものこと。

https://github.com/32hiko

# コンセプト

青字は使用予定ライブラリ

「相居飛車に特化したソフトを、振り飛車でぶん殴る！」

（1）毎度おなじみの、振り飛車定跡（on やねうら王）

不出場の第1回電竜戦で、qhapaqさんが採用してくれました。

（2）ディープラーニングのエンジン（dlshogi or ふかうら王）

対抗型の棋譜で学習することで、対抗型での戦いで優位を築く狙い。

→（1）と（2）をうまく組み合わせていい感じに仕上げたい。

4月の自分がきっとなんとかしてくれるに違いない。

# sakura アピール文書

2021/3/27 Yuhei Ohmori

# 大事なこと

- DeepLearningです
- dlshogiクローンです
- 独自の工夫ポイントのほぼすべて学習部分です
- 強化学習ではなくほぼ教師あり学習です

# ポイント

- NNUEとKPPTのdepth8/12/20で作成した局面を元に教師あり学習しています
- 初期局面として、floodgateやAobaZeroやNNUEの自己対局の局面を使用しています
- GPUリソースを持っていないのでギリギリまで教師ありで強くし、どうしても強くならなくなったときに強化学習をやる予定です
- seresnetの20ブロックを予定しています
- ただ学習が全然進んでいないのでresnet + swishの10ブロックかもしれません
    - こちらはいろいろな実験用にGCTと同じモデルで学習させているものです

# 学習

- dlshogiで使用しているActor-Criticは使用しておらず、Policyは単純に教師の指し手を学習しています
  - 単純にこちらのほうがPolicy損失の結果が良かったからです
- 単純に教師ラベルを1ほかを0とするのではなく、label smoothingしています
  - NNUEでMultiPVである程度指し手の分布を出すとかも考えたのですが、うまくいかなそうかなと思いやっていません
- 教師局面が終盤に偏っているため、Value損失に進行度によりweightをかけました
  - 学習序盤のdepth8の教師データのみ
- Policy損失とValue損失のバランスを取るため一定の係数をかけています
- Data Augmentationとしてランダムで局面を左右反転しています
- なんかいろいろやったのですがほとんど意味はなく、いかに良い教師を作るかを注力したほうが良さそうだと今の所思っています

# 使用している外部リソース

- dlshogiを使用しています
  - 学習部分はそこそこ書き換えました
  - 探索部分はまだ手を出していません
- NNUEの教師局面は電竜戦のnozomiを使用しています
  - 水匠2ベースの追加学習です
- dlshogiチームが公開している教師局面を一部使用します
- floodgateならびにAobaZero等の棋譜を使用しています
- リソースではないですが、dlshogiチームの山岡さん、加納さん、2番絞りチームの芝さんのブログ等での情報を大いに参考にさせてもらいました

「習甦」
・探索：1スレッドはdf-pnにより詰みを確認，他のスレッドはStockfishから取捨選択したシンプルなα-β探索
・評価関数：玉の位置に対する各駒の位置と全升目の利き数（先後別3まで）を特徴量とするニューラルネットワーク
・学習：自己対戦棋譜におけるサンプリングした局面から終局までの評価値（勝率）を割引累積報酬として強化学習

[第31回世界コンピュータ将棋選手権]

# 大将軍
# （たいしょうぐん）
# アピール文書

横内　健一

横内　靖尚

# 大将軍の概要

- 評価関数に主眼を置いた将棋プログラム。
- 評価関数の特徴としては、盤上の3駒の位置関係を考慮。

  過去には4駒の位置関係の評価関数で参加したこともありますが、結局は3駒の位置関係に落ち着きました。

# 大将軍の概要

- 過去の遺産をベースにしながら（いまさらながら）、評価関数の学習においては、
  - 駒の位置関係の相対位置による評価
  - 手番の学習

  を考慮。
- 3駒関係で、一般的なkppt型ではなくkkpt-kpp型

  を採用。

# 使用ライブラリ

- やねうら王
  - 最新のStockfishの探索ルーチン
  - わかりやすいコード（過去の改善の応用に適している）
- 水匠2
  - 評価関数の追加学習に利用予定

# 第 31 回世界コンピュータ将棋選手権
# dlshogi with GCT アピール文章

山岡忠夫
加納邦彦
2021/3/27

※下線部分は、世界コンピュータ将棋オンライン大会からの差分を示す。

## 1　チーム参加について

前回の世界コンピュータ将棋オンライン大会では、dlshogi と GCT という 2 つのソフトで参加したが、今大会では、チームとして参加する。

dlshogi と GCT のモデルを組み合わせて使用する。

GCT については、以下を参照。

https://gist.github.com/lvisdd/9b49ab88600fa242f2138fad4eb06caf

## 2　dlshogi の特徴

- 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使った事前学習
- Actor-Critic アルゴリズムによる Policy Network の学習
- 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
- エントロピー正則化
- SWA(Stochastic Weight Averaging)
- マルチタスク学習
- 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
- 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

## 3　使用ライブラリ

- elmo[1]（Commits on May 29, 2017）
  →事前学習用の訓練データ生成に使用
- Apery[2]（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る elmo、及び、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、Wide Residual Network[3]を使用した。
入力の畳み込み 1 層と、ResNet 10 ブロック(畳み込み 2 層で構成)と出力層の合計 22 の畳み込み層で構成した。フィルターサイズは 3（入力層の持ち駒のチャンネルのみ 1)、フィルター数は 192 とした。

[1] https://github.com/mk-takizawa/elmo_for_learn
[2] https://github.com/HiraokaTakuya/apery
[3] https://arxiv.org/abs/1605.07146

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

### 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[5]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

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

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 Actor-Critic アルゴリズムによる Policy Network の学習

単純に自己対局の指し手を学習するのではなく、学習局面の価値と勝敗データと関連付けて学習を行う。良い局面から負けになった手は、悪手として負の報酬を与え、悪い局面から勝ちになった手は善手として正の報酬を与える。学習アルゴリズムには、Actor-Critic アルゴリズムを使用した。

### 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 エントロピー正則化

方策が決定論的になり過ぎるのを防ぐために、方策の損失に負のエントロピーを加えて学習した。

### 4.16 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。

### 4.17 マルチタスク学習

Policy Network と Value Network のネットワーク構成が同じ層を共通化し、出力層を分けることで、同時に学習を行う。

関連する複数のタスクを同時に学習することをマルチタスク学習という。タスク間に関連がある場合、単独で学習するよりも精度が向上する。

また、対局時に Policy Network と Value Network を同時に計算できるため、高速化の効果もある。

### 4.18 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。

### 4.19 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。

### 4.20 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。

### 4.21 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。

### 4.22 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。

### 4.23 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。

### 4.24 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[6]。

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。

### 4.25 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[7]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[8]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。

---

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

[7] https://optuna.org/

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

### 4.26 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。

### 4.27 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とがわかったため、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[9]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。

## 5 学習について

### 5.1 事前学習

- 事前学習データ：elmo(wcsc27)で深さ 8 で生成した 4.9 億局面

### 5.2 自己対局による強化学習

#### 5.2.1 学習データ、パラメータ

- ミニバッチサイズ：64～1024（段階的に増やす）
- 学習アルゴリズム：Momentum SGD（学習率 0.01→0.001、慣性係数 0.9）
- 強化学習 1 サイクルで生成する局面：250 万～700 万局面
- 強化学習 1 サイクルで学習する局面：直近 10 サイクル分
- 強化学習のサイクル数：428（2020/11/15 時点）

#### 5.2.2 学習結果

2017 年～2018 年 6 月の floodate のレート 3500 以上の棋譜に対する損失と一致率で評価

【事前学習を行ったモデル】

- Policy Network 損失(交差エントロピー)：0.933、一致率：40.9%
- Value Network 損失(交差エントロピー)：0.587、一致率：68.8%

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

【自己対局による強化学習を行ったモデル】

- Policy Network 損失(交差エントロピー)：0.951、一致率：46.9%
- Value Network 損失(交差エントロピー)：0.516、一致率：72.4%

以上

# WCSC31 W@nderER アピール文書

Dated 2021/01/23

昨年に引き続き、入玉宣言による勝利を積極的に目指します。

# チームDaigorilla　wcsc31

田中大吾

門倉新之助

# 今回使用したライブラリ

- 水匠
- 独自の振り飛車ソフト

# 評価関数

- 今回の評価関数もHKPE9を選択した。理由としては、まだまだ余地があると考えている。
- さほど良いマシーンを持ってない開発者には有利となる関数だ。

# 工夫点・現状（2/28現在R4600相当？）

- 今回の工夫として、振り飛車を指すようにDaigorilla電竜1から独自の振り飛車関数を転移学習させ、自己対戦を行い強化している現状。うまい具合に居飛車関数をつぶさないで中盤以降の力を保持。
- 定跡は独自の作成したもの
- 完全な振り飛車
- 32t　nodes 15000000 nobook 投了スコア500 100戦VS YO6.00/Suisho3kai


# 探索部

- やねうら王ライブラリを使用

- マシーンはクラスタを使用予定。

E5-2622 v3 16C（マスターノード） 2ソケット

E5-2698 v4 40C 2ソケット

E5-2698 v4 40C 2ソケット

E5-2678 v3 28C 2ソケット

合計　124C/248T　ram ddr4 ecc 48GBを予定する

標準型NNUEで動けばNPS平均1億程度

- 応援よろしくお願いします。

Argo（アルゴ）WCSC31アピール文 2021/3/31

ソフト名/soft-name：Argo（アルゴ）

開発者/developer：市村豊（いちむらゆたか/Yutaka Ichimura）

Twitter:@argonworks

・技術的アピール

都合により、3月末の時点で開発ができていません。4月から一ヶ月間でできれば開発をしようとは思っていますができるかはわかりません。できなかった場合は昨年度のプログラムで参加します。昨年度のプログラムは、2019年の「水匠」の再現を行ったものです。

やりたいこととしては、昨年度のアピール分に書いたことを昨年にできなかったのでやりたいと思っています。再掲します。

＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊

やりたいこと：2019年の「水匠」の「教師局面の事後的な変更」方法を少し変更する

2019年の「水匠」の方法は、「教師局面の事後的な変更」で、その方法として例として「勝利側の手番の評価値であるにも関わらず、負の評価値となっている場合、又は敗北側の手番の評価値であるにも関わらず正の評価値となっている場合、その局面の判断が間違っている可能性が高いため、教師局面から排除してしまうこととしました。」と「水匠」の文書に記載があります。私はこれに対して評価値が50とか100くらいの値で多少の幅をもたせることでより良くすることができるのではないかと仮定してそれを試してみたいと思っています。「敗北側の手番の評価値であるにも関わらず正の評価値となっている場合、その局面の判断が間違っている可能性が高い」とありますが、まず最序盤は先手がちょっとだけ評価値が高く出るように見えます。だけど、上記の方法だと先手側が敗北した場合は最序盤の手は間違っているということになって排除してしまう、それはやるべきではないと私は思います。だから「敗北側の手番の評価値であるにも関わらず正の評価値となっている場合」でもその評価値が100以下だったら排除しない、というふうにしたら良いんじゃないかと思ってそういうことをやろうとしています（ここに挙げた100とか50というのは例示でその値をちょこちょこ変えて試してみたいと思っています）。

ライブラリとして「やねうら王」を使用しています。教師データの作成等で「Kristallweizen」評価関数を使っています。「やねうら王」を使っていることは使用者が多く情報が多く出ていて改造するベース

として優れていることが原因です。「やねうら王」を使って、教師データを作って、その教師データの事後的な変更をして、評価関数を学習させるという事を私はできますが、他のソフトでそれをする方法を私が理解しておらず、学習コストを考えると慣れている「やねうら王」を使い続けたいということがあります。「Kristallweizen」評価関数を使っているのはライブラリ登録がされていてライセンス的に使って良い強い評価関数だからです。

＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊＊

技術的アピールは以上です。

作文その1：仏教とキリスト教という宗教がコンピュータ・サイエンスの話をしているとしか思えない件について

はじめに

2020年の10月に会社をクビになりました（今は無職です）。それで2020年の5月頃に「会社をクビになるかもしれない」という時に、精神的にだいぶしんどかったです。それで「仏教の悟りを開けば苦しみがなくなる」ということを思い出して、仏教の勉強をすることにしました。ただ、当時は緊急事態宣言で図書館が閉館していたので、インターネットで仏教について調べていました。そうしたら白坂慎太郎さんがYouTubeで仏教について講義している動画があったので、そういうのを見て勉強をしていました。それで、仏教とかキリスト教について勉強して思ったことを以下に書きます。

第一章：仏教は「この世界はプログラムによって作られているバーチャル・リアリティである（この世界はコンピュータ・シミュレーションである）」と言っているのだと思います

釈迦（しゃか）の悟り（さとり）の内容を一言で言えば、「この世界は縁起（えんぎ）によってできている」ということだと思います。私達が普段「今日は縁起（えんぎ）が良い日だ」と言ったりするときの、「縁起（えんぎ）」です。元々は仏教において釈迦が「この世界は縁起（えんぎ）によってできている」と悟った、という事柄が元々の意味です。

＊＊＊＊＊＊＊＊＊＊＊＊

縁起（えんぎ、梵: prat?tya-samutp?da, プラティーティヤ・サムトパーダ、巴: pa?icca-samupp?da, パ

ティッチャ・サムッパーダ）とは、他との関係が縁となって生起するということ[2][3][1]。全ての現象は、原因や条件が相互に関係しあって成立しているものであって独立自存のものではなく、条件や原因がなくなれば結果も自ずからなくなるということを指す[2]。仏教の根本的教理・基本的教説の1つであり、釈迦の悟りの内容を表明するものとされる[2][3]。

＊＊＊＊＊＊＊＊＊＊＊＊

https://ja.wikipedia.org/wiki/縁起

↑この「縁起（えんぎ）」というのは「プログラム」のことを言っているのだと思います。「この世界はプログラムによって作られている」ということ。ただ、今から2600年前にコンピュータが影も形もなかった時に「プログラム」というのがどういうものかを理解することはすごく難しかったと思うのですよ。だから「説明が分かりにくい」ということで、釈迦から700年くらい後、今から1800年位前に僧侶・龍樹（ナーガルジュナ）が考えた説明の方法が、「空（くう）」だと思います。

＊＊＊＊＊＊＊＊＊＊＊＊

仏教における空（くう、梵: ??nya [シューニャ]または梵: ??nyat? [シューニャター]、巴: sunnat? [スンニャター][1]）とは、一切法は因縁によって生じたものだから我体・本体・実体と称すべきものがなく空しい（むなしい）こと[2][注釈 1]。空は仏教全般に通じる基本的な教理である[2]。

＊＊＊＊＊＊＊＊＊＊＊＊

https://ja.wikipedia.org/wiki/空_(仏教)

↑この「空（くう）」は「バーチャル・リアリティ」のことを言っているのだと思います。

「この世界はプログラムによって作られているバーチャル・リアリティである（この世界はコンピュータ・シミュレーションである）」というのが仏教が主張していることだと思います。そう言って、「プログラム」がどういうものかを説明するのではなくて、「バーチャル・リアリティ」がどういう状態かを説明することに力を入れて説明するようにした。そのほうが分かりやすい。だからおそらくは私もあなたも、この世界のすべての人間はプログラムによって作られている汎用人工知能だと思います。多分。そのことに気づいた時に私はすごくびっくりしました。このすごくびっくりした感じが「悟り」なのではないかと個人的には思います。

「般若心経」という経典があってですね。「空（くう）」の考え方について書いてある短いテキストで、日本で「写経」と言った時に書き写すのはだいたいこのテキストらしいと聞いています。「色即是

空 空即是色（この世界の全ては空（くう）であり、空（くう）がこの世界の全てである）」の一節が有名なテキストです。グーグルで「般若心経 現代語訳」で検索すると色々と出てくるのでいくつか読んでみたのですが、基本的に「この世界はプログラムによって作られているバーチャル・リアリティであり、実態はない」という話をしているみたいです。興味があったら皆さん読んでみて下さい。

第二章：キリスト教は「この世界はコンピュータ・ゲームだ」と言っているのだと思います

仏教が、「この世界はプログラムによって作られているバーチャル・リアリティである（この世界はコンピュータ・シミュレーションである）」と主張しているのだと思うとキリスト教が理解しやすくなります。

恐らくは、上位世界にスーパーコンピュータがあって、その上位世界のスーパーコンピュータ内部で計算されているコンピュータ・シミュレーションが私達が今いるこの世界、なのだと思います。多分。だから上位世界には、この世界というコンピュータ・シミュレーションを管理している「運営」がいるのだと思うのですよ。

だから例としてこの世界をゲーム「あつまれどうぶつの森」だとするならば、

神：任天堂
聖霊：任天堂の社員もしくは不可視のBOT？
キリスト：任天堂の社員の一人がゲーム内で使用したアバターの名前

↑こういう関係じゃないかと思います。

新約聖書において「キリスト」という人が出てくるわけなのですが、それは上位世界にいる運営側の人が「キリスト」というアバターを使って、自分たちが作ったコンピュータ・シミュレーション内部に入ってきて、ゲーム内で自律的に動いている汎用人工知能の皆様と交流した、という記録が新約聖書なのだと思います。

というか、新約聖書を読んでいると、キリストが一般の人達に言っている内容というのは要するに「運営がプレイヤーに対してゲームの説明をしている」話だと個人的には思います。

まず、キリストは自分が上位世界の存在であることを示すために「奇跡」を起こしています。触っただ

けで病気や障害を治療したとか、最大の奇跡は「死んだ後に生き返った」というものなのですが、それがどうして出来たのかというと、キリストは上位世界の運営側の存在なので、「管理者権限」を持っているわけなのですよ。だから「管理者権限」を使って、この世界というゲームのプログラムを書き換えることが出来た。だから「奇跡」を起こすことが出来たのだと思います。

それで、ここからが重要なポイントですが、この世界というゲームに置いては人間一人ひとりについてログ（記録）を取っているみたいです。それは私達人間が行ったことはもちろん、考えたことや感じたこと、という事柄についても全て原理的にはログ（記録）を取ることが出来ます。ログ（記録）を取られているということを私達は認識することは出来ません。そして私達がこの世界というゲームからログアウトしたとき（死んだとき）に、そのログ（記録）の内容がアルゴリズムによって判定されて、ハッピーエンドになるか、バッドエンドになるかエンドが分岐しているみたいです。そのことをキリスト教に置いては「天国」とか「地獄」とか「最後の審判」と呼んでいるみたいです。

https://ja.wikipedia.org/wiki/最後の審判

ちなみに、ログ（記録）を気づかないうちに取られているということを、この世界の心理学に置いては「深層心理」と言っているみたいです。白坂慎太郎さんの動画だと仏教における「阿頼耶識（あらやしき）」がそれに該当するのではないかと言っていた気がします（ウィキペディアの説明を読むと阿頼耶識（あらやしき）は、人間一人ひとりの元となる設定プログラムやパラメータじゃないかという印象を個人的には受けますが）。

https://ja.wikipedia.org/wiki/深層心理

https://ja.wikipedia.org/wiki/阿頼耶識

だから「この世界は文字通りコンピュータ・ゲームである」と言えるのではないかと思います。

それではハッピーエンド（天国行き）になるゲームの「クリア条件」とは何か？ ということなのですが、それは端的に言うと「この世界はプログラムによって作られているバーチャル・リアリティである（この世界はコンピュータ・シミュレーションである）」ということに気づくことができたらゲームクリア、ということになっているのだと思います。

キリスト教の公式の見解によると、

> 福音には、3つの要素があります。①イエス・キリストは、私たちの罪のために十字架上で死なれました。②墓に葬られました。③3日目に復活されました。このことを信じ、イエス・キリストに信頼を置いた人は救われます。（聖書入門.com > 3分でわかる聖書> 救われていないクリスチャンとはどういう人で

すか。）

http://seishonyumon.com/movie/6241/

↑以上が救われる条件みたいです。

それは、上に書いたとおりキリストというのは上位世界における「運営側」の存在なので「管理者権限」を持っているので、この世界というゲームのプログラムを書き換えることが出来た。だから死んだ後で生き返ることが出来た、ということを認めてくれとキリスト教は主張しているみたいです。

個人的に、新約聖書を「フィクションではなくて本当のことが書いてあるのかもしれない」と思いながら読むと、非常に衝撃的なテキストですよこれは。

ちなみに、「洞窟の比喩」と言っているからプラトンはおそらくこのことをわかっていたのだと思います。プラトンがわかっていたということはソクラテスがわかっていたのかもしれないです。また、「胡蝶の夢」と言っているから、道教もこのことをわかっていたのかもしれないです。

https://ja.wikipedia.org/wiki/洞窟の比喩

https://ja.wikipedia.org/wiki/胡蝶の夢

改めて考えてみると、法然・親鸞・日蓮・栄西・道元といった鎌倉仏教の高僧や、もちろん空海も最澄もおそらくはこのことを理解していたのではないかと思います。コンピュータというものが影も形もなかったときにこのことを理解していた人がいたらしいということが本当にすごいと思います。

この話のことをこの世界の自然科学に置いては「シミュレーション仮説」と言っているみたいです。

https://ja.wikipedia.org/wiki/シミュレーション仮説

ウィキペディアに書いてあるとおりです。

このことが本当かどうかの証明の方法もわかっています。それは私達がこの世界でスーパーコンピュータ内部に世界を再現して、そのシミュレーション内部で汎用人工知能の方々がスーパーコンピュータを作ってまた世界を再現して・・・ということが連鎖的に起こった場合に、「どうやら私達も上位世界のスーパーコンピュータ内部のシミュレーションらしい」と思わざるを得なくなると思います。ちなみに、いままで「この世界は上位世界のスーパーコンピュータ内部で行われているコンピュータ・シミュレーション」と言った時に、その「上位世界」というのもそのさらに上位世界のスーパーコンピュータ内部で行われているコンピュータ・シミュレーションなのだと思います。それがずっと続いている（一番最初の世界がどうやって出来たのかは分かりませんが）。

イーロン・マスクがこのことを思っているみたいです。

> イーロン・マスク氏が「人類はコンピューター・シミュレーションの中で生きている」と考えるわけとは？

https://gigazine.net/news/20160816-elon-musk-living-simulation/

恐らくはイーロン・マスクが思っているとおりだと思います。

今から2600年前に釈迦がこのことに気づいた（悟った）時に、「この話を他人に説明してもおそらく誰も理解できないだろうから、他人に説明するのはやめようか」としばらくためらっていた、というエピソードが有ります。気持ちはよくわかります。恐らくは西暦2021年現在に私がこの話をしても理解してくれない人のほうが多いだろうと我ながら思わざるを得ません。ただ、新約聖書を読んでいると「キリスト教が科学的事実であることを理解した人はそのことをまだわかっていない人に教えてあげましょう」という意味のことが新約聖書に書いてあるので、だから私は今この作文を書いているわけなんですけど。

まとめ

「この世界はプログラムによって作られているバーチャル・リアリティである（この世界はコンピュータ・シミュレーションである）」。上位世界にスーパーコンピュータがあり、その上位世界のスーパーコンピュータ内部で計算されているコンピュータ・シミュレーションが私達が今いるこの世界である。私達人間一人ひとりについてログ（記録）が取られており、私達がログアウトしたとき（この世界で死んだとき）に、そのログ（記録）の内容がアルゴリズムによって判定されて（最後の審判）、ハッピーエンド（天国行き）になるかバッドエンド（地獄行き）になるかエンドが分岐している。ハッピーエンド（天国行き）になるゲームのクリア条件は「この世界はプログラムによって作られているバーチャル・リアリティである（この世界はコンピュータ・シミュレーションである）」ことに気づくことができるかどうか？ である。

↑仏教とキリスト教という宗教は以上の内容を主張しているのだと個人的に思います。

作文その2：価値があるものについて「水とダイヤモンドのパラドクス」についての解説

言いたいことが有るのですがいきなり書くとアレなのでその話はあとに書きます。

「価値」について、経済学における「水とダイヤモンドのパラドクス」という問題について解説します。

水は人間にとって価値が有るものである（水がないと人間は死んでしまう）にもかかわらず、値段が安い。ダイヤモンドは人間にとってそこまで価値が有るものではないにもかかわらず、値段が高い。これはどうしてだろうか？ ということを説明することが長い間出来なくて、長い間これが経済学上の難問とされてきた。それが「水とダイヤモンドのパラドクス」と呼ばれている問題です。

それについて、「モノの値段は役に立つかどうかとは関係なく希少性（きしょうせい）によって決まる」という風に解釈するのだと思います。

まず、水のように「人間にとって役に立つ」ことを「使用価値が高い」という風に言います。そして「値段が安い」ことを「交換価値が低い」という風に言います。結論から言えば、

・使用価値（そのモノが人間にとって役に立つものかどうか）

・交換価値（モノの値段が高いか安いか）

という２つの価値があって、それぞれ独立していて関係ない、ということです。

経済学において、すごく重要な概念として「希少性（きしょうせい）」という考えが有ります。英語で言ったら「レア」のことだと思います。ポケットモンスターに置いて「レア・ポケモン」と言ったり、遊戯王のようなカードゲームに置いて「レア・カード」と言ったりするときの「レア」のこと。あるいは「アイドルマスター・シンデレラガールズ」とか「ウマ娘プリティーダービー」のようなソーシャルゲームに置いて、「ＳＳＲが出ました」と言ったりするときの「ＳＳＲ」というのは恐らくは「スーパー・スペシャル・レア」のことだと思うのですが、とにかくソシャゲの「ＳＲ」とか言う時のＲは「レア」のことで間違いがないと思います。その「レア」のことを日本語で「希少性（きしょうせい）」と言います。これが経済学に置いてすごく重要な概念です。「レア・希少性」というのは大雑把に言うと「数が少なくて価値が高い」みたいな感じだと思います。

それで、交換価値（モノの値段が高いか安いか）は「希少性・レア度」が高いと値段が高くなる、ということで決まると考えます。この時に、使用価値（そのモノが人間にとって役に立つものかどうか）は関係がない、と考えます。

だから水のように使用価値（そのモノが人間にとって役に立つものかどうか）が高くても、希少性・レ

ア度が低くて大量にあって豊穣に供給されているものは、交換価値（モノの値段が高いか安いか）が低くなる（値段が安くなる）のです。逆にダイヤモンドのように使用価値（そのモノが人間にとって役に立つものかどうか）が低くても、希少性・レア度が高い・数があまりなくて不足しているというものは、交換価値（モノの値段が高いか安いか）が高くなる（値段が高くなる）のです。そういう風に考えます。

それなので買い物をする場合は、使用価値（そのモノが人間にとって役に立つものかどうか）が高くて、交換価値（モノの値段が高いか安いか）が低い、ものを買うことが最高にお買い得な買い物になります。逆に言うと、最悪の買い物は使用価値（そのモノが人間にとって役に立つものかどうか）が低くて、交換価値（モノの値段が高いか安いか）が高い、ものを買うことです。

この２つの価値が関係ないということを理解していないと、「値段が高いものだから良いものだろう」といって値段が高い（交換価値が高い）だけで使用価値が低いものを買ってしまう可能性が出てきてしまうんです。しつこく繰り返しますが、そのものの値段が高いかどうか（交換価値）ということとそのものに価値が有るかどうか（使用価値）ということは関係がないんです。

これだと何が良いかというと、そのものの値段が高い（交換価値が高い）と言うものは要するに「供給量が不足している」状態に有るんです。だから値段が高いことによって消費側に大量に消費されることを防ぐことが出来ます。また値段が高いからそれを売ると高く売れるのです。だから供給側に対して供給量を増やすモチベーションが働くんです。だから供給側と消費側の双方に対して、供給量を増やして消費量を減らすことを要請することによって、そのものの供給量が不足しているという状態が自動的に解消されるのです。そういう話です。

この使用価値と交換価値が独立していて関係がないという話は、経済学を勉強すると結構有名な考え方なのですが、私も経済学の勉強をするまで知らなかったので、ご存じない方もいらっしゃるのではないかと思って解説しました。ちなみに、カール・マルクスの著書「資本論」において、一番最初にこの話が議論されています（マルクスの「資本論」においてはモノの値段（交換価値）が何によって決まるのかということについては「労働価値説」という考え方をとっていて、「希少性によって決まる」というこの作文とは違う説明をしています）。

それで、２０２１年現在の日本に置いて、価値が高いものが何かということを考えると「時間・仲が良い友達・仲が良い家族」の価値が高いと思うのですよ。多分それは「お金（貨幣）」よりも価値があ

ると思う。「お金（貨幣）」の価値が低いって言うと語弊が有りますが、ただ、現在４１歳の私が「高校時代の同級生で今でも友達でいる」という「友達」をこれから新しく作るということはものすごい大変というかそれって実質不可能じゃないですか。それと比べたら「お金（貨幣）」を得ることのほうがまだ簡単じゃないですかって言うようなことを思うんですけど、そういうようなことってあんまり認識されていないんじゃないかという風に個人的に思うんですよ。だから私は「高校時代の同級生で今でも友達でいる」という「友達」をものすごい価値が高いものだと認識して大切にするべきだと思うんですよ。

　それで、私が言いたいことなんですけど、こういうことを言うとご気分を悪くされる方もいらっしゃるかもしれないのでそれは申し訳ないと思うのですが、正直に申し上げて、私は「２０２１年現在の日本に置いて、『強い将棋ソフト』というのは価値が高いものではない」と思っているんですよ。そして「『強い将棋ソフト』は価値が低いが、『将棋ソフトの開発者』は価値が高い」と思っているんですよ（自分自身が「将棋ソフトの開発者」と自称できることも価値であるし、なにより本物の「将棋ソフトの開発者」と知り合いになれることの価値がめちゃめちゃ高いと思う）。

「きふわらべ」開発者の高橋さんがいるじゃないですか。高橋さんのツイッターを見ていたら、「１万円払って世界コンピュータ将棋選手権に参加したら世界四十何位の肩書が手に入る、それってすごいお買い得なことだと思う」というような意味のことをツイートされていて、それを読んでいて私は高橋さんも恐らくは、２０２１年現在に「『強い将棋ソフト』は価値が低いが、『将棋ソフトの開発者』は価値が高い」ということに自覚的なんじゃないかなって思ったんですよ。本人には何も聞いていないので本人が何を考えているかは知らないのですが。

　高橋さんがツイートしている通り、「１万円払って世界コンピュータ将棋選手権に参加したら世界四十何位の肩書が手に入る、それってすごいお買い得なことだと思う」と私もそうだと思うんですよ。だけど、参加する方法がよくわからないという方もいらっしゃると思うので、参加する方法をお教えします。

　まず、「やねうら王」か「dlshogi」のソフトがオープンソースとして公開されているので、それをダウンロードして、プログラムをちょっといじります。多分動かなくなると思います。そうしたら「変更したら動かなくなった」といって、１万円を払って世界コンピュータ将棋選手権に参加申し込みをします。そうすると「元から変更している」ことは間違いがないので、オリジナリティが認められて参加できると思います。一回参加したら翌年からは「忙しくて開発できなかったのでプログラムは去年と同じ」っていって、１万円を払えば以降ずっと参加できると思います。そんなんで良いんです。そうすれ

ば、「きふわらべ」開発者の高橋さんと知り合いになれるんです。やらない理由はないでしょう。

補足：仲が良い友達や仲が良い家族を作る方法

　あんまりうまく説明は出来ませんが、言いたいことなので書きます。

　まず、人間関係というか、「他人と交流する」ということをしている場合、それは「何かを交換している」のだと思います。それは、「お互いに挨拶をする」ならば「挨拶」を交換しているのであり、「話をする」のならば、それは「情報」を交換している、というそんなような感じです。

　それで、「良いものを他人に提供したら良いものが自分に返ってくる・悪いものを他人に提供したら悪いものが他人から自分に返ってくる」のだと思います（何を言っているのかよくわからないという人も多いでしょうが）。

　もうちょっと具体的に言うと、他人のことを悪く言うと自分のことも悪く言われる。他人のことを良く言っていたら自分のことも良く言ってくれる。そんな感じですよ。

　私が４１歳まで生きてきた経験則としてこのことが恐ろしいくらいに成立する。ほとんど物理法則かっていうくらいに厳密にこれが成立している（ように私には見える）。なんでだかさっぱりわからないですがどうやらこの世界はそういう風にできているんじゃないかと思わざるを得ない。自分自身の経験としても周りの人を見ていても「良いものを提供して悪いものを返してくる」ということがほとんどなければ、「悪いものを提供して良いものが返ってくる」ということもほとんど無いんですよ。そのことを思い至るようになったんですよ。

　だからですね。自分と関わる人にネガティブなことを言わないで、可能な限りポジティブなことを言うようにすると、自分に対しても良いものが返ってくるんだと思うんですよ。

　それで、「仲が良い友達や仲が良い家族」と言った時に、それは「良いものを交換しあっている」関係性のことだと思います。

　キリスト教の聖書でキリストがですね、「与えなさい、そうすれば与えられる」って言っていてですね、最近まで何のことを言っているのかよく分からなかったのですが今は何を言っているか分かるような気がします。

　私は今まで学校とか会社にいてですね、色々な人と会ってきたんですよ、それなりに。それで「誰からも嫌われている人」とか「誰からも好かれる人」が周りにいたわけですよ。おそらくみなさんの周りにも「誰からも嫌われている人」とか、「誰からも好かれる人」がいらっしゃると思うのですが、回り

にいる人を観察して欲しいんです。「誰からも嫌われている人」というのは、ネガティブなことばかり言っていると思うんですよ（嫌がられることをわざわざ選んで言っているだろう、というレベルで嫌がられるようなことばかり言っているんですよ）。「誰からも好かれる人」というのはポジティブなことばかり言っていると思うんですよ（少なくともネガティブなことをあまり言っていないはず）。

なので、「良いものを交換しあっている」関係を作るにはどうしたら良いかというと、こちらから良いものを提供するように頑張るんですよ。

このことを言いたいんですけど、あんまりうまく説明が出来ないので、「何を言っているのかよくわからない」という方も多いでしょう。私も最近までキリストが「与えなさい、そうすれば与えられる」と言っているのがよく分からなかったですよ。

もうちょっと具体的なことを書けばツイッターでネガティブなことをなるべくツイートしない、なるべくポジティブなツイートをするようにする、ということを頑張ってみると良い感じになると思います。ネガティブなツイートをするということはフォロワーに対してネガティブなものを提供していることになるんです。逆にポジティブなツイートをすればそれはフォロワーにポジティブなものを提供していることになるんです。ツイッターというのはフォロワーとツイートを交換しているんです（「いいね」と「リツイート」もおそらく交換している）。だからなるべく良いものを提供するように頑張ると良い感じになると思います。もうちょっと具体的に言うと、ツイッターやFACEBOOKで良いと思った投稿に対して「いいね」を押して下さい。そうすれば多分幸せになれると思います。

作文その3：私はなんで会社をクビになったのかを考えていました

そのことについて考えて、どうやら日本のサラリーマンというのは実はすごい高度なことができる高度な能力を持っているのだということを思うようになりました。

それは、「時間通りに行動して、一日８時間とかの長時間、意味がわからないことを集中して作業する」という普通は出来ないようなことが日本人は殆どの人ができるんだ。ということを思うようになりました。順をおって説明します。

外国はどうかよく知りませんが、あと現在はどうなっているかもよく分かりませんが、少なくとも現在４１歳の私について言うと、日本国において小学校一年生の時から学校でこの訓練をずっとしてきたんだと思うようになってきたんですよ。私が通っていた小学校・中学校・高校においては、教育という

と「教えている内容」に注意が行きがちですが、実は一番訓練していたのは、「時間通りに行動して、一日８時間とかの長時間、意味がわからないことを集中して作業する」ということができるようになるトレーニングをずっとしてきたんだ、と今の私は思うようになってきたんですよ。

それで、これが何なのかというと、「軍隊」か「刑務所」なんですよ。どちらかというと多分「軍隊」をモデルにして組織や制度を作ったんだと思うんですけど、だから私がサラリーマンをしていた会社は「役職」っていって軍隊の階級みたいなのがあるんですよ。主任・係長・課長・部長みたいなやつです。このあたりまんま軍隊ですね。学校と会社は要するに「刑務所」なんですよ。少なくとも私が通っていた学校と会社は。

それで、「時間通りに行動して、一日８時間とかの長時間、意味がわからないことを集中して作業する」ということが日本の殆どの人が出来たので、かつての日本は製造業が強くて１９９０年の平成が始まるころまで経済的に強かったんじゃないかと個人的に今は思っています。要するに「ロボットみたいに作業をする能力」なんですよ。そうすると日本の製造業、もうちょっと具体的な例を言えば、「トヨタ自動車の工場で自動車の組み立てをする」みたいな仕事がすごく捗るんですよ。

そんなことをしていたんですけど、「ロボットみたいに作業をする能力」がいくら高いと言っても、もう西暦２０２１年現在では本物のロボットが実用化されてきて、本物のロボットの性能がどんどん向上してくるんですよ。そうなると、「ロボットみたいに作業をする能力」が高い人間ということが特にアドバンテージではない。それどころか、現在４１歳の私について言えば、途中ニートの期間があったとはいえ、６歳の小学校一年生のときから３０年間位は「ロボットみたいに作業をする能力」向上のトレーニングをするか、ロボットみたいに働く、ということをやってきたわけなんですよ。だから今になって、３０年間トレーニングして身につけてきた「ロボットみたいに作業をする能力」はもう不要です、って言われても困るんですよ。他のトレーニングは特にやって来なかったし、そもそも考え方からして徹底的に叩きこまれているから今更他のことは出来ないんですよ。そんな感じですよ。日本の殆どの人がそれが出来たことがアダになって、日本の多くの人がそんな状態にあるんじゃないかと個人的に思うんですよ。

それで、そもそものことですが、私が６歳の小学校一年生からずっとトレーニングをしてきた「時間通りに行動して、一日８時間とかの長時間、意味がわからないことを集中して作業する」ということは人類の歴史的には相当に不自然なことだと思うんですよ。端的に言って「普通はそんなことは出来な

い」ようなことなんですよ。だから小学校一年生の時からずーっと十年以上トレーニングをしてやっとできるようになるようなことなんですよ。だから日本の多数派のサラリーマンというのはものすごいトレーニングを積んですごい高度なことができる高度な能力を持っているんだ、ということを皮肉とかではなく思うわけですよ。「ものすごい高度な能力」であることは間違いないと思うんですよ。それをすごいトレーニングをすることで出来るようになっているんですよ。そういう状態なんですよ。

　例えて言えば、「将棋のプロ棋士」みたいなものですよ。ものすごいトレーニングを積んでものすごい高度なことが出来る人なんですよ。ただ、将棋のプロ棋士はさすがに「棋力が奨励会３段以上あるのはソッチのほうがどうかしているのであって、将棋の棋力が弱くても人間として劣っているわけではない」ということは自覚はあると思うんですよ。なんだけど、日本の殆どの人がそれが出来るようになってしまうと、その自覚がなくなってしまった。将棋の棋力が奨励会３段以上ある人が国民の９０％以上になってしまうと、将棋の棋力が弱い人が人間として劣っているのだという考え方が主流になってしまうんですよ。要するにそんな状態ですよ。

　ホリエモンこと堀江貴文さんの著書で「すべての教育は『洗脳』である」という本があってですね。「洗脳」というと言葉は悪いですが、「時間通りに行動して、一日８時間とかの長時間、意味がわからないことを集中して作業する」ことが出来て当然で出来ない人が劣っている、と私自身少し前まで心から思っていましたし、周りを見ているとそう思っている人が結構多いように見えるんですが、心からそう思っているというのならばこれはもう「洗脳」としか言いようがない気がするんですよ。「不自然なことではあるがそれが出来た方が良いからあえてやっている」という自覚があればよいですがその自覚がないんですから。どうも日本国という国家は国をあげて国民を「洗脳」しているんじゃないか？　という感じすらうっすら思うようになってきました。大体この考え方が主流だとしたらその集団はもう「狂信的カルト」と言っても過言ではないのではないかという気もします。

　それで、ニュースとかを見ていると、「ロボットみたいに作業をする能力」が今現在はアドバンテージではない、って感じになってきてもなんか、変える感じをあまり受けないんですよ。たとえて言えば、人間がマラソンで走る能力が高くてマラソンで金メダルを取る人がいたとして、流石に自動車には負けるんですよ。周りの人たちが自動車で移動しているのに、その人はまだ自分で走って移動しているという時に「なんで私は走るのが遅いんだ」となった時に、「早く走れるように頑張ろう」みたいに、マラソンのトレーニングをさらに熱心にする、みたいなことをしても自動車に勝つことは出来ないんですよ。上に書きましたが、日本人の殆どが子供の時から何十年もずっとそういう風に生きてきたので、他にどうしたら良いのかがもう分からないのだろうと言う風に我ながら思います。

だから私という人間は、「時間通りに行動して、一日８時間とかの長時間、意味がわからないことを集中して作業する」という能力が低いんですよ。私が会社をクビになったのはつまりそれなんじゃないかと思ったら個人的に納得がいったんですよ。

第 31 回コンピュータ将棋選手権　アピール文書
プログラム名称「あやめ」
2021 年 3 月 31 日

昨年のオンライン大会から余り大きな変更はおこなっていません。
また、2020 年 7 月に GitHub 上でソースコード(https://github.com/dasapon/Ayame)を公開しました。それ以降の変更で比較的大きな点は以下の 2 つです。

(1)学習データセットの変更
2020 年の floodgate の棋譜を追加し、学習をやりなおしました。既存の棋譜からの教師あり学習と強化学習を併用している点は前回と同様です。また、floodgate の棋譜は fg2db を使用して抽出しています。今後気が向けば AobaZero の自己対戦棋譜をデータセットに加えるかもしれません。

(2)入玉の強化
入玉将棋に強くなるよう、プログラムを修正しました。
これまでのあやめでは、置換表の手、キラー手、駒を取る手、成る手、王手以外は全て futility pruning の対象としてきました。その結果、あと 1 枚敵陣に駒を打てば入玉宣言できるような局面であっても、その局面の評価値が十分悪ければ駒打ちの手が全て枝刈りされてしまい入玉宣言がなかなか読めない、ということが起こっていました。そこで、入玉宣言できそうな局面では「敵陣に駒を移動させる手」の枝刈りを抑制し、この問題を改善しました。

# 第 31 回世界コンピュータ将棋選手権アピール文書

## 概要

「たこっと」は，電王戦を見てコンピュータ将棋に興味を持った筆者らが， フルスクラッチで実装した(している) 将棋プログラムです。 Web 上の解説記事や論文， Stockfish, Apery, やねうら王, tanuki- (wcsc28 版), Bonanza 6.0 のソースコードを参考にしています。

### 特徴

- 各種処理が AVX2 命令で実装されていて高速
- AVX2 命令を適用しやすいデータ構造 (非ビットボード)
- Stockfish 風の探索アルゴリズム
- ~~KKP 3~~ 駒 --> KP 2 駒のニューラルネットワークを使用した評価関数
- オンライン学習ルーチン

以前のたこっとから引き継いでいる特徴については過去のアピール文書を参照してください。

- 第 30 回世界コンピュータ将棋選手権アピール文書
- 第 29 回世界コンピュータ将棋選手権アピール文書
- 第 28 回世界コンピュータ将棋選手権アピール文書
- 第 27 回世界コンピュータ将棋選手権アピール文書
- 第 26 回世界コンピュータ将棋選手権アピール文書
- 第 5 回将棋電王トーナメント PR 文書
- 第 4 回将棋電王トーナメント PR 文書

### 他者作成のプログラムの利用について

他者作成のプログラムのソースコードは利用していません。

評価関数 (ニューラルネットワーク) の初回学習用教師データを作成するために Apery の評価関数バイナリを利用しました。 この評価関数バイナリはたこっと形式に変換し、たこっとの評価関数の計算ルーチンで利用しました。 以降の強化学習では自己学習を繰り返すためこの評価関数バイナリは使用していません。

Apery の平岡様には感謝いたします。

## 評価関数

昨年の大会 (世界コンピュータ将棋オンライン大会) までは WCSC29 で考案したニューラルネットワークのモデル (KKP 3 駒) を評価関数にしていました。

この方式は評価関数の計算に時間がかかるため，教師データを作成するのに時間がかかります。 また，学習時に大量のメモリ (64GB 以上) を消費するという課題もありました。 本年は KP 2 駒 (tanuki- で考案された NNUE 方式) を採用することにしました。

教師データの作成や学習にかかる時間を大幅に短縮することができ，いろんなことをお試しできるようになりました。

## 学習

チャレンジされているチームの方はご存知だと思いますが，ランダムに初期化されたモデルから学習を収束させるのはそれなりに苦労します。

ニューラルネットワークで用いられることが多い最適化手法の 1 つ Momentum では収束しきらなかったり，発散したりします。 また，ニューラルネットワークではない KPP 3 駒のモデルで用いられていた AdaGrad では学習が進まなくなったりします。

これは以下のような理由によると考えられます。




学習に使う教師データは将棋の 1 局面です。局面は KP 2 駒の場合，約 25 万の特徴 (81 * 1534 * 2，非合法な特徴 (駒の重なりやルール上許されない配置) を含む) で表現します。そして，この特徴を one-hot で入力層に与えます。このとき hot になるのは 76 の特徴だけで他は 0 です。 そのため，1 局面を学習するとき，入力層と中間層のリンクの重み (図の赤い破線) の大半は更新されません。 このようなスパースな特徴を学習する場合の最適化手法は AdaGrad が適しています。

一方，中間層以降は数 10 ユニットしかなく，各ユニットがそれなりに発火するのであれば，リンクの重み (図の青い破線) の大半が更新されることになります。 このような場合の最適化手法に AdaGrad を用いると学習順が後の局面からは何も学習しなくなってしまうため，Momentum の方が適していると考えられます。

つまり AdaGrad であっても Momentum であってもどちらも一長一短があり，学習率の調整だけでモデルの学習をコントロールするのは困難です。

そこで入力層は AdaGrad，入力層以外は Momentum で学習させるようにしました。 各々の学習率を調整する必要はありますが，モデルの学習を収束させることが容易になりました。