†白美神†
PR文書

野田 久順
岡部 淳
鈴木 崇啓
日高 雅俊

目次

- †白美神†とは
- 名前の由来
- 自己対戦の棋譜を用いた大規模機械学習
- 評価関数の高速化
- 参考文献

†白美神†とは

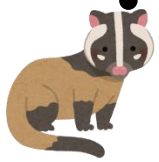
- 「はくびしん」と読みます
- 磯崎元洋氏(やねうらお)による『やねうら王』から派生したコンピューター将棋ソフトです
- 評価関数の強化と高速化に力を入れています
- 開発コンセプトは「楽に楽しく面白く」です

名前の由来

- 昨年は「tanuki-」という名前で参加しました
- 今年はやねうら王ベースでの参加です
- 「たぬき」と「やねうら」に関する名前にしたい
- 見た目が「たぬき」っぽく、
  民家の「やねうら」に住み着く習性がある
  「ハクビシン」という動物がいることを知りました
- そのままだとダサいので当て字を考えてみました

名前の由来 (続)

- 野田「白い美しい神っていう当て字を
  考えたんですけど…。」
- 岡部「…厨二病ですね。」
- 鈴木「…両脇にダガーとかつけたら
  いいんじゃないですか？」
- 日高「…まぁ…いいんじゃないですか？」
- 野田「(えっ…、止めないの…？)」



自己対戦の棋譜を用いた大規模機械学習

- NineDayFever・Ponanza等が導入している自己対戦の棋譜を用いた大規模機械学習を用いています
- 浅い探索の評価値で深い探索の評価値を近似できるようにする
- 通称『雑巾絞り』

自己対戦の棋譜を用いた大規模機械学習 (続)

- 問題を定式化すると以下のようになります

$$\underset{\omega}{\operatorname{argmin}} L(\omega)$$

ここで

ω: 調整対象となる重みベクトル

=評価関数ファイルの中身

L: 目的関数

自己対戦の棋譜を用いた大規模機械学習 (続)

$$L(\omega) = \sum_i l(\omega, \varphi_i, \omega', \varphi'_i)$$

ここで

l: 損失関数

i: 自己対戦で生成した棋譜の局面の通し番号

φ_i: i番目の局面に対する浅い探索のPVの末端ノードの特徴量ベクトル

ω': 自己対戦に用いた重みベクトル=評価関数ファイル

φ'_i: i番目の局面に対する自己対戦中の深い探索のPVの末端ノードの特徴量ベクトル

自己対戦の棋譜を用いた大規模機械学習 (続)

- 損失関数として考えられる関数は...
  1. 評価値の差の2乗
  2. 評価値から推定した勝率[1]の差の2乗
  3. 評価値から推定した勝率の確率分布の交差エントロピー
  4. ...
- †白美神†では3.の交差エントロピーを用いています

自己対戦の棋譜を用いた大規模機械学習 (続)

$$p = \sigma(\omega'^{t}\varphi'/600)$$

$$q = \sigma(\omega^{t}\varphi/600)$$

とおく。ここで

σ: 標準シグモイド関数

p: 深い探索の評価値から推定した勝率

q: 浅い探索の評価値から推定した勝率

自己対戦の棋譜を用いた大規模機械学習 (続)

$$l = \mathrm{H}(p, q) = \mathrm{E}p[-\log q]$$

$$= -\sum_{x} p(x) \log q(x)$$

ここでxは勝ちか負けの2値をとるので

$$= -p \log q - (1 - p) \log(1 - q)$$

自己対戦の棋譜を用いた大規模機械学習 (続)

lの勾配l'は

$$l' = \frac{\partial l}{\partial \omega_i} = -\frac{pq'}{q} - \frac{(1-p)(1-q)'}{1-q}$$

$$= -\frac{pq'}{q} + \frac{(1-p)q'}{1-q}$$

$$= \frac{-pq'(1-q) + (1-p)qq'}{q(1-q)}$$

自己対戦の棋譜を用いた大規模機械学習 (続)

ここで

$$\delta'(x) = \delta(x)\delta(1-x)$$

より

$$l' = \frac{-pq'(1-q) + (1-p)qq'}{q'}$$
$$= -p(1-q) + (1-p)q$$
$$= -p + pq + q - pq$$
$$= q - p$$

- 以上で求めた勾配の式を使ってミニバッチ勾配降下法を行います

自己対戦の棋譜を用いた大規模機械学習 (続)

- 機械学習は以下の条件で行っています
  - 深い探索の深さ: 3
  - 浅い探索: 0手読み+静止探索 置換表無効化
  - データ量: 20億局面×複数回
  - 学習手法: ミニバッチ勾配降下法
  - 最適化手法: Adam[2] Gradient Noise[3]
  - ミニバッチサイズ: 10万局面
  - 学習率: 評価値に対し1/64 学習率減衰無し
  - 損失関数: 交差エントロピー
  - 次元下げ: なし
  - 棋譜データの評価値の絶対値の上限なし
    - ただし詰みの局面は除く

評価関数の高速化

- 『tanuki-』『たぬきのもり』で培った
  SIMD命令を用いた評価関数の
  高速化技術を採用しています

評価関数の高速化 (続)

- 評価関数の処理時間のほとんどはメモリへの
  ランダムアクセスが占めています
- AVX2のVGATHERDD命令を用いると、
  このランダムアクセスを3駒関係8組み分
  まとめて処理させることが出来ます

評価関数の高速化 (続)

- 評価関数の高速化前の擬似コード

```
short kpp[81][1548][1548][2];  // 重みベクトル
int list[38];  // 駒番号の配列
int value[2] = { 0, 0 };
for (int i = 0; i < 38; ++i) {
  for (int j = 0; j < j; ++j) {
    for (int k = 0; k < 2; ++k) {
      // 以下の行がボトルネック
      value[k] += kpp[king_square][list[i]][list[j]][k];
    }
  }
}
```

評価関数の高速化 (続)

- 高速化後の擬似コード

```
__m256i zero = _mm256_setzero_si256();
__m256i sum = zero;
for (int i = 0; i < 38; ++i) {
  const auto* p = kpp[king_square][list[i]];
  for (int j = 0; j < i; j += 8) {
    __m256i index = _mm256_load_si256(reinterpret_cast<const __m256i*>(&list[j]));
    __m256i mask = MASK[std::min(i - j, 8)];
    __m256i w = _mm256_mask_i32gather_epi32(zero, reinterpret_cast<const int*>(p), index, mask, 4);
    __m256i wlo = _mm256_cvtepi16_epi32(_mm256_extracti128_si256(w, 0));
    sum = _mm256_add_epi32(sum, wlo);
    __m256i whi = _mm256_cvtepi16_epi32(_mm256_extracti128_si256(w, 1));
    sum = _mm256_add_epi32(sum, whi);
  }
}
sum = _mm256_add_epi32(sum, _mm256_srli_si256(sum, 8));
int value[2];
_mm_storel_epi64((xmm*)value, _mm_add_epi32(_mm256_extracti128_si256(sum, 0), _mm256_extracti128_si256(sum, 1)));
```

最期に

- たぬきシリーズメイン開発者野田は第3回電王トーナメントを期に磯崎元洋氏(やねうらお)
に弟子入りさせていただきました
- やねうらお門下生として先生の名に恥じないような活躍を見せられればと思っています

参考文献

[1] https://twitter.com/issei_y/status/589644174275674112

[2] Diederik Kingma and Jimmy Ba (2015): "Adam: a method for stochastic optimization," the 3rd International Conference for Learning Representations (ICLR 2015).

[3] A. Neelakantan, L. Vilnis, Q.V. Le, I. Sutskever, L. Kaiser, K. Kurach, J. Martens

Adding Gradient Noise Improves Learning for Very Deep Networks

arXiv, 2015.

[4] 海野裕也, 岡野原大輔, 得居誠也, 徳永拓之 (2015): "オンライン機械学習 (機械学習プロフェッショナルシリーズ)," 講談社.

[5] インテル® C++ コンパイラー 16.0 ユーザー・リファレンス・ガイド