既存評価関数のパラメタを活かした適応学習

矢野友貴[†1] 三輪誠[†2]
横山大作[†3] 近山隆[†1]

コンピュータゲームプレイヤの評価関数のパラメタ調整において, 新たに特徴を加えて学習を行う際, パラメタを零から調整し直すのが一般的である. パラメタを零から調整し直すということは, 今まで蓄えてきた知識を持つ既存パラメタを捨てることを意味する. 一方, データマイニングや自然言語処理の分野では関連性の高いドメインの既存パラメタを活用して調整を行うドメイン適応という手法が研究されており, 高い成果を上げている. 本稿では, ドメイン適応の手法を評価関数のパラメタ調整に導入することで既存パラメタを活かす学習手法を提案する. 本手法を用いて将棋の評価関数のパラメタ調整を行ったところ, 単純に既存パラメタを初期値等に用いる手法に比べて総合成績で勝る結果を得ることに成功した.

Adaptive Learning Utilizing Parameters of Existing Evaluation Function

Yuki YANO,[†1] Makoto MIWA,[†2] Daisaku YOKOYAMA[†3]
and Takashi CHIKAYAMA[†1]

When new features are added to evaluation functions of computer game players, their parameters are usually tuned from scratch. This, however, means throwing out existing parameters which should reflect already acquired knowledge. On the other hand, for data mining and natural language processing, domain adaptation methods have made successes, which utilize existing parameters already tuned for related domains as the basis of parameter tuning. In this paper, we propose a method to utilize existing parameters as the basis for tuning a new evaluation function with added features, adopting the ideas of domain adaptation. We applied this method to tune evaluation function for shogi and have shown that a player tuned with our method showed better results against players tuned with more straightforward methods.

1. はじめに

コンピュータゲームプレイヤにおいて, 局面の有利不利を正確に判断する評価関数は強いコンピュータプレイヤを作成する上で重要な要素の一つである. 正確な評価関数を作成するためには, それを構成するパラメタをうまく調整する必要がある. この調整を人手で行うことは, 扱うゲームに対する熟練した知識と膨大な労力が要求されるため, 極めて困難である. そのため近年では, 機械学習を用いてパラメタを自動調整する手法が広く研究されている[1)–4)]. 機械学習では膨大な棋譜データを用いてパラメタの調整を行うことで, 精度の高い評価関数を作成することが可能であり, 現在多くのトップレベルのコンピュータゲームプレイヤで利用されている.

より精度の高い評価関数を作成するためには, パラメタを適切に調整するだけでなく, 局面の状態をより正確に識別できるような特徴を用意する必要がある. 既存の評価関数に対して新たな特徴を追加して, よりよい評価関数を構築していくことも強いコンピュータゲームプレイヤを作る上で重要な要素である. 評価関数に新たな特徴を追加した場合, 新しい評価関数のパラメタを零から調整し直すのが一般的である. しかし, 既存の評価関数のパラメタが膨大な棋譜データや人間の熟練した知識によって十分に調整されている場合, 零から学習しなおすことはこれらの今まで蓄えてきた知識を捨てることとなってしまう.

一方, データマイニングや自然言語処理の分野では, 今まで学習によって蓄積してきた特定のドメイン (元ドメイン) の知識を, それと関連のある他のドメイン

†1 東京大学大学院工学系研究科
Graduate School of Engineering, The University of Tokyo
†2 東京大学大学院情報理工学系研究科
Graduate School of Information Science and Technology, The University of Tokyo
†3 東京大学 IRT 研究機構
IRT Research Initiative, The University of Tokyo

(目標ドメイン) での学習に活かすドメイン適応と呼ばれる手法が広く研究されている[5),6)]. ドメイン適応では, 元ドメインの訓練データによって得られたパラメタを目標ドメインでの学習の指標として用いることで, 元ドメインの知識によって学習の範囲をある程度絞りつつ目標ドメインに適合するようなパラメタ調整を行う. これにより, 例えば目標ドメインにおける訓練データが十分にない場合, 訓練データの豊富な元ドメインにてパラメタの推定をあらかじめ行うことで, ドメイン間に共通する一般的知識を活用して精度の高い学習を可能とする.

本稿では, 既存パラメタが存在する条件下でパラメタ調整を行う際にドメイン適応を導入することで, 既存パラメタに蓄えられている知識を活かしつつ, 新たなパラメタの調整を行う手法を提案する. 将棋を例に既存パラメタを初期値とする手法や, 既存パラメタを固定値として用いる手法等と比較実験をした結果, 総合評価において最もよい成績をあげることに成功した.

本論文では以降, 2 章にて関連研究について述べたのち, 3 章にて提案手法を, 4 章に提案手法に関する比較実験を行う. 最後に 5 章にてまとめと今後の課題について述べる.

2. 関 連 研 究

2.1 兄弟局面の比較を用いたパラメタ調整

将棋では, 棋譜の善し悪しを適切にラベル付けすることが難しいため, 棋譜の局面の状態を直接用いて学習を行うことは困難である. これに対して, 棋譜で指された手の後の局面と他の合法手の後の局面 (兄弟局面) の比較を行い, 棋譜で指された手の方の評価値が高くなるようにパラメタの調整を行う手法が提案されている[2)–4)]. 具体的な調整方法は次の通りである.

(i) 棋譜中の各局面に対して以下の操作を施す
  (a) 全合法手を生成する
  (b) 棋譜で指された手の後の局面 s_i と, 他の合法手の後の局面 t_i のペア (s_i, t_i) を作る
  (c) 各局面に対して探索を行い、最善応手手順を求め、ディスクに格納する

(ii) 求めた最善応手手順を利用して以下のようにパラメタの更新を行う
  (a) 各ペア (s_i, t_i) に対して、それぞれの局面の最善応手手順後の局面を作成し、その局面での特徴ベクトル $(\mathbf{x}_{s,i}, \mathbf{x}_{t,i})$ を求める
  (b) 特徴ベクトルの差分 $\mathbf{x}_i = \mathbf{x}_{s,i} - \mathbf{x}_{t,i}$ と評価値の差が正であるべきか負であるべきかを表す変数 y_i(先手なら $y_i = 1$, 後手なら $y_i = -1$) を求める
  (c) $y_i \mathbf{w}^T \mathbf{x} > 0$ となるようにパラメタ $\mathbf{w}$ を調整する

実際の計算では時間の大部分が操作 (i) で費やされるため, 一回のパラメタの調整による最善応手手順の変化が十分に小さいと仮定し, (ii) を複数回行うことで計算コストの削減を図る.

パラメタの調整を行う際, $y_i \mathbf{w}^T \mathbf{x} > 0$ の満たされなさの指標として損失関数 $L(\mathbf{w}, S)$ を定義し, その勾配を用いる. 損失関数には $y_i \mathbf{w}^T \mathbf{x}$ が正になるほど値の小さくなる関数が選ばれる. 保木の手法[2)] では損失関数として (1) 式のようなシグモイド関数が用いられている.

$$L(\mathbf{w}, S) = \sum_{(\mathbf{x},y)\in S} \frac{1}{1 + e^{ay\mathbf{w}^T\mathbf{x}}} \qquad (1)$$

(1) 式は, 評価値の差がある程度の範囲にあるペアについて評価値の差が改善するようにパラメタ調整を行う一方で, 極端に差のついたペアについては重要視しないという特徴がある. また, 保木の手法では損失関数の勾配を直接用いずに, 勾配の符号に応じた定数幅による調整を行っている. 一方, 金子等の手法[3),4)] では損失関数といて (2) 式のようなロジスティック回帰が用いられている.

$$L(\mathbf{w}, S) = \sum_{(\mathbf{x},y)\in S} \log\left(1 + e^{-y\mathbf{w}^T\mathbf{x}}\right) \qquad (2)$$

(2) 式は, 評価値の差が正の部分よりも負の部分の方が勾配が大きいため, すでに正しい順序の局面のペアの評価値をさらに広げる改善よりも, 間違った順序の局面のペアを正す方を重視する性質がある. また, 金子等の手法では Stochastic Meta Descent[7)] 等の 2 次収束の性質をもつ最適化手法を用いた勾配法によるパラメタ調整が行われている.

2.2 ラベル付きデータを用いたドメイン適応

ドメイン適応とは, 図 1 のように異なるドメイン (元ドメイン) で得られた知識を用いて実際に分類器を用いるドメイン (目標ドメイン) での学習を行う手法である. ドメイン適応には, 元ドメインや目標ドメインの訓練データのラベルのありなしによって様々な手法が存在する[5),6)] が, ここでは元ドメインと目標ドメインの双方にラベルが付いている場合の手法である Bayesian Divergence Prior[8)] の考え方を用いる.

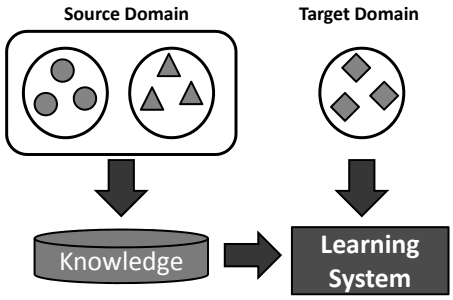

図 1 ドメイン適応

まず初めに, 一般的なパラメタ調整を考えてみる. パラメタの調整は, 与えられた訓練データ群 $(\mathbf{x}, y) \in S$ を用いてパラメタ $\mathbf{w}$ の事後確率 $\prod_{(\mathbf{x},y)\in S} p(\mathbf{w}|\mathbf{x}, y)$ を最大にするように調整を行うことを意味する. ベイズの定理より, 事後確率 $\prod_{(\mathbf{x},y)\in S} p(\mathbf{w}|\mathbf{x}, y)$ は (3) 式のように尤度 $p(y|\mathbf{x}, \mathbf{w})$ とパラメタの事前分布 $\pi(\mathbf{w})$ の積に比例することが導かれる.

$$\prod_{(\mathbf{x},y)\in S} p(\mathbf{w}|\mathbf{x}, y) \propto \pi(\mathbf{w}) \prod_{(\mathbf{x},y)\in S} p(y|\mathbf{x}, \mathbf{w}) \quad (3)$$

ここで, $\mathbf{x}$ と $\mathbf{w}$ は独立であると仮定している. 実際の計算では, (3) 式の対数をとることで積を和に変換して用いられる. また, 多くの場合には全体に -1 を掛けることで最小化問題としてパラメタ調整を扱う. 最終的に, パラメタ調整は (4) 式のような目的関数の最小化問題へと帰着される.

$$\min_{\mathbf{w}} \quad L(\mathbf{w}, S) - C \ln(\pi(\mathbf{w})) \quad (4)$$

(4) 式の第 1 項は損失関数であり, 2 章で述べた (1) 式や (2) 式のことである. 第 2 項は事前分布に伴う正則化項であり, 例えば事前分布としてガウス分布を仮定した場合は, 第 2 項は $\frac{C}{2}\|\mathbf{w}\|^2$ と L2 正則化項となる. なお, C はハイパーパラメタである.

次に、元ドメインで学習されたパラメタ $\mathbf{w}^{tr}$ を保持しているときに, 目標ドメインの訓練データ群 $(\mathbf{x}, y) \in S^{ad}$ に適合するパラメタ $\mathbf{w}^{ad}$ を学習する場合を考える. 最も単純な方法としては, $\mathbf{w}^{ad} = \mathbf{w}^{tr}$ として事前に持っているパラメタを直接用いる方法, $\mathbf{w}^{ad}$ を S^{ad} を用いて零から学習しなおす方法が考えられるが, 前者の方法ではもし元ドメインと目標ドメインでのデータの分布が大きく異なる場合によい精度を得られず, また後者の方法では目標ドメインの訓練データが少ない場合にうまく学習できない危険性がある. そこでドメイン適応では, 目標ドメインの訓練データ群での損失関数を最小化しつつ, 元ドメインで得られたパラメタとの差異を最小化するように学習が行われる. この「元ドメインとの差異」を表す指標は Bayesian Divergence Prior と呼ばれ, 尤度を用いて (5) 式のように定義される.

$$\begin{aligned} \ln(p_{div}(\mathbf{w})) = &- D(p(y|\mathbf{x}, \mathbf{w}^{\mathbf{tr}})||p(y|\mathbf{x}, \mathbf{w})) \\ &+ \ln(\pi(\mathbf{w})) + \beta \quad (5) \\ \text{where} \quad &\beta > 0 \end{aligned}$$

ここで, β は規格化定数, $D(p(x)||q(x))$ は $p(x)$ と $q(x)$ の KL ダイバージェンスである. KL ダイバージェンス $D(p(x)||q(x)) \geq 0$ は分布 $p(x)$ と $q(x)$ の間の相対エントロピーを表し, $p(x)$ と $q(x)$ の分布が近ければ近いほど小さい値をとる. KL ダイバージェンスは具体的には (6) 式で定義される.

$$D(p(x)||q(x)) = \int p(x) \ln\left(\frac{q(x)}{p(x)}\right) dx \quad (6)$$

(5) 式を用いると, (4) 式は (7) 式のようになる.[⋆1]

$$\min_{\mathbf{w}} \quad L(\mathbf{w}, S^{ad}) - C \ln(p_{div}(\mathbf{w})) \quad (7)$$

ここで, $p(y|\mathbf{x}, \mathbf{w})$ がクラス分類問題で一般的に用いられるロジスティックシグモイド関数, すなわち

$$p(y|\mathbf{x}, \mathbf{w}) = \frac{1}{1 + \exp(-y\mathbf{w}^T\mathbf{x})} \quad (8)$$

であると仮定すると, (7) 式は次のように展開される.

$$\min_{\mathbf{w}} \quad L(\mathbf{w}, S^{ad}) + C_1\|\mathbf{w} - \mathbf{w}^{tr}\| - C_2 \ln(\pi(\mathbf{w})) \quad (9)$$

(9) 式の第 2 項は L1 正則化の形式をしているが, 扱いやすさの点から L2 正則化に置き換えて用いられることが多い.

3. 提 案 手 法

コンピュータゲームプレイヤの評価関数の学習において, 棋譜を用いた機械学習や手調整によって十分に調整された既存パラメタが存在するような場合を考える. 評価関数の精度を上げるために, 既存評価関数に新たに有効と考えられる特徴を組み込み, 評価関数を拡張したとする. このとき, 評価関数の形状が変化しているため, すべてのパラメタを零から学習し, 新たな最適解を探すのが一般的である. しかし, 既存パラメタには

- 機械学習で調整されたパラメタなら, 調整で用い

⋆1 なお, $\ln(p_{div}(\mathbf{w}))$ はパラメタ間の相違が小さいほど大きくなる点に注意

た棋譜からの知識

- 人手で調整されたパラメタなら, 機械学習では補えない可能性のある人間独自のヒューリスティクスが含まれている. そのため, 既存パラメタの知識を新たな学習でも活用することができれば
- 既存パラメタの持つ知識を新たな評価関数に活かすことで, より強いプレイヤを作成できる
- ある程度学習の進行したパラメタから調整を始めることで, 目的関数の収束が速くなる

といったことが期待できると考えられる.

本研究では, 既存パラメタを新たな学習に活かすことを既存パラメタを新たな棋譜に適応させることで実現する手法を提案する. 具体的には, 2.2 節にて述べたドメイン適応を用い, 既存パラメタをドメイン適応における元ドメインで得られたパラメタと, これから行う新たな評価関数の学習を目標ドメインでの操作と見なすことで, (9) 式を用いた新たな目的関数を定義する. 既存パラメタは (9) 式における $\mathbf{w}^{tr}$ となる. なお, 今回のような評価関数の学習では新たな特徴の追加に伴い既存パラメタと実際に調整するパラメタでは要素数が異なるが, 既存パラメタと対応しない新規パラメタ部分をすべて 0 と見なすことで要素数の差を補う (ドメイン適応においても同様に, 元ドメインにない特徴を 0 とおくことが一般的に行われている[9]). 新規パラメタを 0 で補った既存パラメタを $\mathbf{w}_0$ としたとき, 本手法において最小化する目的関数は (10) 式のようになる.

$$\min_{\mathbf{w}} \quad L(\mathbf{w}, S) + \frac{C}{2}\|\mathbf{w} - \mathbf{w}_0\|^2 \qquad (10)$$

ここで, 簡単のためにパラメタの事前分布 $\pi(\mathbf{w})$ は一様分布である ($\pi(\mathbf{w}) = 1$) と仮定した. 本手法では, 既存パラメタとの相違度を目的関数に加えることで, 学習に用いる棋譜との一致を図りつつ, 既存パラメタの形状の維持を同時に行い, 既存パラメタを活かしたパラメタ調整を実現する. また, 抑制パラメタ C によって既存パラメタをどの程度維持するかを調整することで, 問題に応じた柔軟な対応を可能とする.

なお, 本稿では Bayesian Divergence Prior により既存パラメタとの相違度を設定したが, (10) 式は (4) 式において「パラメタの事前分布 $\pi(\mathbf{w})$ を既存パラメタを中心とするガウス分布と仮定している」とも解釈することもできる[9]。

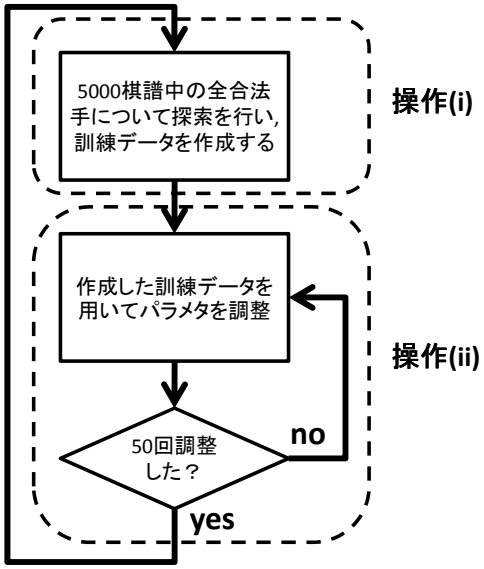

図 2 実験での学習手順

4. 実　　験

4.1 実験方法

本手法の有用性を評価するために, 将棋プログラム「激指」[10] に新たな特徴を追加して実験を行った. 激指は実現確率探索[11] を用いた将棋プログラムであり, 第 18 回世界コンピュータ将棋選手権で優勝するなど, コンピュータ将棋では有数の強さを誇っている. 激指の評価関数を展開したところ, 73 種類, 410 個の特徴を得た. さらに, 激指の既存評価関数に以下の 4 種類の特徴を追加した.

(i) 玉と他の駒の相対位置 (9,248 個)
(ii) 玉と他の駒の絶対位置 (209,952 個)
(iii) 2 駒の相対位置 (147,968 個)
(iv) 隣接する 2 コマの相対位置 (331,776 個)

なお, 玉に関する新規特徴は進行度に応じて特徴量を変化させた. オリジナルと合わせて特徴の総数は 699,354 個となった.

実験では損失関数として (2) 式のロジスティック回帰を用い, 図 2 のように 2.1 節における操作 (i) にて 5,000 棋譜から訓練データを作成したのち操作 (ii) にてその訓練データを用いて 50 回[⋆2]のパラメタ調整を行うという手順を繰り返した. なお, 操作 (i) では深さ 5 の探索を行うことで訓練データの作成を行った.

また, パラメタ更新には stochastic meta descent[7] を用いた. stochastic meta descent では各要素ごと

⋆2 金子等の論文[4] を参考とした

に個別の学習率を持たせ, それぞれの過去のパラメタの更新情報に基づいて学習率を変化させる. 具体的なパラメタの更新式は (11) 式のようになる.

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \mathrm{diag}(\boldsymbol{\eta}_t)\nabla_t \tag{11}$$

ここで, ∇_t は目的関数の $\mathbf{w}$ に関する勾配, $\boldsymbol{\eta}_t$ は学習率を表す. stochastic meta descent では学習率 $\boldsymbol{\eta}_t$ を以下の式によって更新する.

$$\boldsymbol{\eta}_{t+1} = \mathrm{diag}(\boldsymbol{\eta}_t)\mathrm{Max}_{0.5}(1 - \mu\mathrm{diag}(\nabla_t)\mathbf{v}_t) \tag{12}$$

$$\mathbf{v}_{t+1} = \lambda\mathbf{v}_t - \mathrm{diag}(\boldsymbol{\eta}_t)(\nabla_t + \lambda\mathbf{H}_t\mathbf{v}_t) \tag{13}$$

μ はメタ学習定数, $\mathrm{Max}_{0.5}$ は 0.5 より小さい要素を 0.5 に置き換える関数である. また, λ は減衰定数であり, 過去のパラメタの更新情報をどれだけ現在に反映させるかを決めるパラメタである. $\mathbf{H}$ は目的関数の $\mathbf{w}$ に関するヘシアンを表す. ヘシアンを直に求めることは計算コスト, メモリコストの両面から困難であるが, $\mathbf{Hv}$ という形式は $O(n)$ で求める手法が知られており[12], 実装上は大きなロスとはならない.

4.2 評価方法

以降, 実験で得られた結果は次の 3 つの方法によって評価した.

(i) 学習に用いていない 250 棋譜に対する一致率
(ii) 学習に用いていない 250 棋譜に対する不一致度
(iii) 初期値に対する勝率

ここで, 不一致度は (14) 式のように定義した.

$$\text{不一致度} = \frac{1}{\text{局面数}} \sum_{\text{合法手 } i} T(\xi(i) - \xi(\text{棋譜の手})) \tag{14}$$

(14) 式の ξ は最善応手手順後の局面の評価値を, $T(x)$ は (15) 式で定義されるシグモイド関数である.

$$T(x) = \frac{1}{1 + \exp(-7x/256)} \tag{15}$$

対戦は双方合わせて初めの 30 手を固定とし, そのあと 1 手最大 10 秒で探索を行わせた[⋆3]. 初期局面は棋譜から用意し, 先手後手を入れ替えて対戦をし勝率を求めた. なお, 特徴の大幅追加に伴い評価関数の計算コストが増大し, オリジナルの激指に対して探索時間の点で大きく負け越してしまうことがわかったため、対戦では純粋な学習結果を評価するために新規パラメタをすべて 0 としたものをオリジナルとして用いた.

4.3 抑制パラメタの調整

(10) 式の抑制パラメタ C の値に違いによる学習結果の変化を調べるために, C を 10^{-5} から 10^5 と 10 倍ずつ変化させて, 各パラメタに対して既存パラメタを初期値として図 2 の操作を 10 回行い, 結果の評価を行った.

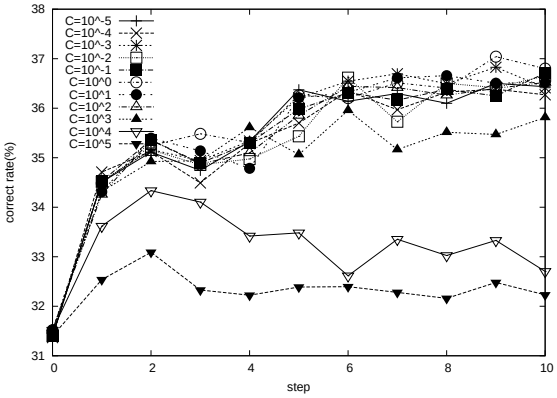

図 3 抑制パラメタと一致率

学習時に用いた各種パラメタの値は, 1 回目の反復に対してパラメタを変化させて調整を行い, 学習率の初期値 $\eta_0 = 10^{-6}$, メタ学習定数 $\mu = 10^{-7}$, 減衰定数 $\lambda = 0.7$ とした. ここで, $C = 10^{-4}$ では 5 回目, $C = 10^{-2}$ は 6 回目, $C = 10^0$ では 8 回目の反復においてパラメタが発散したため, それぞれ途中から減衰定数 λ を 0.5 に引き下げて実験を行った. パラメタが発散した原因は分かっていないが, Stochastic Meta Descent はヘシアン $\mathbf{H}$ が正定値でないときに λ の値が大きいと発散する可能性があることが示されている[7].

なお, 実験にはネットワークに広域分散したクラスタ群である InTrigger[14] を用い, 約 80 並列計算で 10 回の反復に 20 時間程度を要した.

各抑制パラメタにおける一致率の推移を図 3 に示す. 図 3 をみると, $C = 10^3, 10^4, 10^5$ の場合の一致率の推移が他のパラメタに比べて鈍いことがわかる. これは, 抑制項の力が大きいために, 他のパラメタよりも棋譜に一致させるよりも元のパラメタの形状を維持しようという方向に学習が進行したためである. 一方, $C = 10^2$ 以下に関して顕著な違いはないと判断できる. このことから, $C = 10^2$ 以下では抑制項より棋譜に一致させる力が相対的に大きくなることが分かる. また, $C = 10^4, 10^5$ で一致率が一度上がった後で減少しているが, これは初めはパラメタの調整幅が小さいため抑制項の影響が小さく一致率が伸びるものの, 途中から抑制項の寄与分が大きくなり, 元のパラメタに戻そうとする力が強く働き一致率が下がったためと推測される.

⋆3 初め 30 手は竹内等の論文[13] を, 1 手 10 秒による評価方法は鶴岡等の論文[11] を参考とした

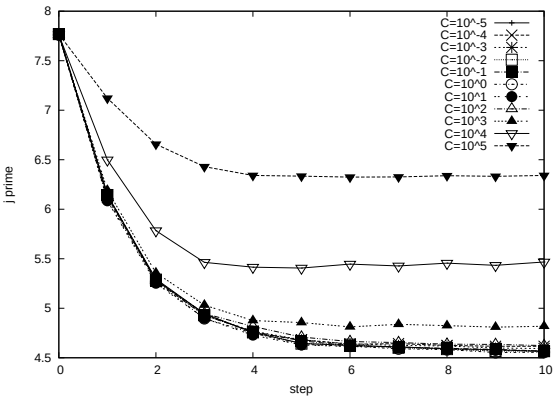

図 4 抑制パラメタと不一致度

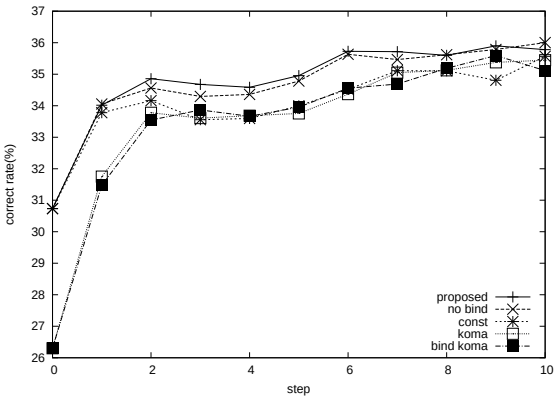

図 5 各手法と一致率

次に, 各パラメタにおける不一致度の推移を図 4 に示す. 不一致度の推移にも一致率と同じような傾向が表れている. すなわち, 抑制項の相対的に強い $C = 10^3, 10^4, 10^5$ では他に比べて不一致度が高い数値で収束しており, また, それ以外のパラメタでは目立った違いが見られない.

各パラメタでのオリジナルに対する 100 試合での対戦結果を表 1 に示す. ここで, 太字の勝率は 50%以上の結果を, *のついた勝率は有意水準 5%の二項検定で有意に勝ち越した結果を表す. 表 1 を見ると, ほとんどのパラメタにおいてオリジナルに対して勝越しに成功していることがわかる. 特に, $C = 10^2$ における結果は有意水準 5%の二項検定において唯一有意に勝ち越すことができており, また図 3 及び図 4 においても一致率, 不一致度ともに大きく改善できていることから, $C = 10^2$ が最適なパラメタであると推測される.

以上の結果より, 以降の実験では $C = 10^2$ における結果を比較対象として用いることとした.

4.4 各種手法との比較

次に, 以下の 4 つの場合について 4.3 節と同様の方法で実験を行い, $C = 10^2$(proposed) の結果と比較した.

(i) 既存パラメタを初期値とし, 抑制項なしで学習 (no bind)

(ii) 既存パラメタを固定し, 新規パラメタのみを学習 (const)

(iii) 駒割のみを初期値として与え, 抑制項なしで学習 (koma)

(iv) 駒割のみを初期値として与え, $C = 10^2$ の正則化付きで学習 (bind koma)

bind koma は初期値として駒割を与えているため, 正則化項として (10) 式の $\mathbf{w}_0$ として駒割のみにパラメタを与えたものを用いた. また, bind koma は 8 回目の反復においてパラメタが発散したため, 途中から減衰定数 λ を 0.5 に引き下げて実験を行った. 比較は前節と同じように 4.2 節の方法を用いた. なお, ここでは初期値に対する勝率だけでなく相互対戦の勝率も求めた. 相互対戦の方法は初期値の場合と同様の方法によって行った. さらに, 一致率及び不一致度を求めるためのテストセット, 勝率を求めるための初期局面は 4.3 節とは異なるものを用いた.

表 1 各抑制パラメタとオリジナルとの 100 試合での対戦結果

抑制パラメタ C	勝率 (%)
10^{-5}	**54**
10^{-4}	**56**
10^{-3}	**60**
10^{-2}	**52**
10^{-1}	44
10^{0}	**58**
10^{1}	**53**
10^{2}	**63***
10^{3}	**59**
10^{4}	**57**
10^{5}	**53**

各手法の一致率の推移を図 5 に示す. 図 5 を見ると, const の一致率が他の既存パラメタを用いた手法に比べてやや低いことがわかる. これは, 既存パラメタを固定したことによってパラメタ調整の自由度が減ったためと考えられる. 一方で, 初期値として駒割のみを用いた koma 及び bind koma は最終的な一致率が proposed, no bind に比べわずかに小さくなっている. また, $C = 10^2$ の条件で抑制項を用いた proposed 及び bind koma はそれぞれ抑制項のない no bind 及び koma と同じような挙動をしていることがわかる.

次に, 各手法の不一致度の推移を図 6 に示す. 不一致度に関しては, const も no kind や proposed と同じような曲線を描いていることがわかる. 一方で, 初

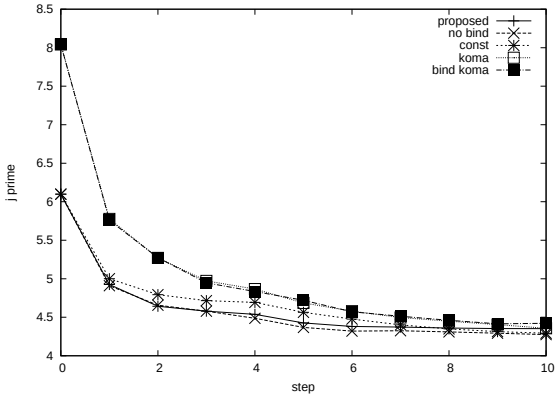

図 6 各手法と不一致度

期値として駒割のみを用いた koma 及び bind koma の 2 つは収束に時間がかかっており, すでに調整のされたパラメタを初期値として与えることで, 収束を早める効果があることが示唆されている. また, 初期値の異なる koma, bind koma も含めてすべての手法が最終的に同じような不一致度に収束していることがわかる.

各手法のオリジナルに対する 100 試合での対戦結果を表 2 に, 200 試合での相互対戦の結果を表 3 に示す. ここで, 太字の勝率は 50%以上の結果を, *のついた勝率は有意水準 5%の二項検定で有意に勝ち越した, もしくは負け越した結果を表す. まず, 表 2 を見ると, propsed のみがオリジナルに対して有意に勝ち越していることがわかる. また, proposed や no bind といった初期値として既存パラメタを用いた手法が, 初期値として駒割のみを用いた koma, bind koma に比べ高い勝率を記録しており, 既存パラメタを初期値で用いることの有用性がうかがえる. 一方, 表 3 の結果を見ると, 表 2 の場合と同様に, 初期値として既存パラメタを用いた手法の成績がよい. 特に proposed は他の 4 つの手法のうち 3 手法に対して有意に勝ち越していることがわかる. no bind に対しては単純な勝率では直接対決において proposed が勝っているものの, 十分有意に勝ち越す結果を得ることはできなかった. しかし, 表 2 において proposed が no bind よりもよい結果を残していること, 表 3 において proposed が no bind に比べより多くの対戦で有意に勝ち越していることから, proposed が総合的に最もよい結果を記録しているといえる. この結果は, no bind が proposed と違い抑制項がないため過学習が起き, 汎化性能の面で proposed に劣ったことが原因ではないかと考えられる. しかし一方で, koma と bind koma では抑制項の導入による改善が見られない. これは, bind koma における抑制パラメタの値を単純に proposed と同じにしたことが影響していると推測され, bind koma においても最適な抑制パラメタを調整することで結果が改善される可能性がある. また, const の結果が proposed と no bind に比べて悪いことから, 既存パラメタを完全に固定することが逆に新規特徴を加えた新しい評価関数の学習の足枷になっていると考えられる.

表 2 各手法とオリジナルとの 100 試合での対戦結果

手法	勝率 (%)
proposed	**61***
no bind	**55**
const	**52**
koma	48
bind koma	49

表 3 各手法間の 200 試合での対戦結果

	pr	nb	co	ko	bk
proposed		**53.5**	**60***	**61.5***	**60***
no bind	46.5		**61.5***	**55**	**60***
const	40*	38.5*		**59***	**52**
koma	38.5*	45	41*		**54**
bind koma	40*	40*	48	46	

以上の実験結果をまとめると, 次の二つの結論が得られる.

(i) 初期値として十分調整された既存パラメタを用いることで学習効率がよくなる

(ii) 既存パラメタを用いる場合, パラメタ調整を適度に抑制することで汎化性能が高まる

結論 (i) は, 図 5 において proposed, no bind の一致率が koma, bind koma に比べ高いこと, 図 6 において proposed, no bind, const がより素早く収束していること, 表 2, 3 において既存パラメタを用いた手法が相対的に高い勝率を記録していることから裏付けられる. 結論 (ii) は, 表 3 において proposed が no bind, const に比べてよい結果を残していることに由来する. これら二つの結論は, 提案手法の特徴である「棋譜との一致を図りつつ, 既存パラメタの形状を維持する」を支持するものであり, また実験的にも本手法が最もよい結果を残していることから, 本手法の有用性が示されたといえる.

5. おわりに

本稿では, 既存パラメタの形状を維持しつつ, 棋譜との一致率を向上させるために, ドメイン適応の一種である Bayesian Divergence Prior を用いる学習手法の提案を行った. 実験では単純に既存パラメタを用いるものや, 既存パラメタを固定するもの, 既存パラメタ

を用いないものとの比較実験を行った. その結果, 抑制パラメタの調整を行った本手法が総合的に最も良い結果を得ることに成功し, 本手法の有用性を示すことができた. 現在, 多くのコンピュータゲームプレイヤが作成され, それに伴い様々な既存パラメタが存在する. そのため, 本稿のような既存パラメタを新たな学習に活かすという試みの重要性は高いと考えられる.

今後の課題としては, より特徴を吟味した評価関数を用いた本手法の評価が挙げられる. 本実験では, 駒の相対位置, 絶対位置といったとても単純な特徴を追加した. しかし, このような特徴は非常にスパースであり, 表 2 において proposed 以外が有意に勝ち越す結果を得られていないことからもこれらの特徴が十分に有効であるとは言い難い. そのため, GPS 将棋[15]の特徴などその有用性が十分に評価されている特徴を組み込んで実験を行ってみる必要があると考えられる.

謝辞 本研究の一部は文部科学省科学研究費補助金特定領域研究「情報爆発に対応する高度にスケーラブルなソフトウェア構成基盤」の助成を得て行われた.

参 考 文 献

1) M. Buro, M. Evaluator, P. Selfplayed, P. Update, O.B. Play, G.T. Searcher, and P. Correction. LOGISTELLO—A Strong Learning Othello Program. *NEC Research Institute, Princeton, NJ. http://www. cs. ualberta. ca/mburo/ps/log-overview. pd f.*
2) 保木. 局面評価の学習を目指した探索結果の最適制御. 第 11 回ゲームプログラミングワークショップ, pp. 78–83, 2006.
3) 金子知適. 兄弟節点の比較に基づく評価関数の調整. 第 12 回ゲームプログラミングワークショップ, pp. 9–16, 2007.
4) 金子知適, 山口和紀. 将棋の棋譜を利用した, 大規模な評価関数の調整. 第 13 回ゲームプログラミングワークショップ, pp. 152–159, 2008.
5) S.J. Pan and Q. Yang. A survey on transfer learning. Technical report, Technical Report HKUST-CS08-08, Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, China, 2008.
6) J. Jiang. A Literature Survey on Domain Adaptation of Statistical Classifiers. 2007.
7) N.N. Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural Computation*, Vol.14, No.7, pp. 1723–1738, 2002.
8) X. Li and J. Bilmes. A Bayesian divergence prior for classifier adaptation. In *Eleventh International Conference on Artificial Intelligence and Statistics (AISTATS-2007)*, 2007.
9) C.Chelba and A.Acero. Adaptation of maximum entropy capitalizer: Little data can help a lot. *Computer Speech & Language*, Vol. 20, No.4, pp. 382–399, 2006.
10) 将棋プログラム「激指」のページ. http://www.logos.ic.i.u-tokyo.ac.jp/gekisashi/.
11) Y.Tsuruoka, D.Yokoyama, and T.Chikayama. Game-tree search algorithm based on realization probability. *ICGA Journal*, Vol.25, No.3, pp. 132–144, 2002.
12) B.A. Pearlmutter. Fast exact multiplication by the Hessian. *Neural Computation*, Vol. 6, No.1, pp. 147–160, 1994.
13) 竹内聖悟, 林芳樹, 金子知適, 山口和紀, 川合慧. 勝率に基づく評価関数の評価と最適化. 情報処理学会論文誌, Vol.48, No.11, 2007.
14) InTrigger's homepage. http://www.intrigger.jp.
15) GPS 将棋. http://gps.tanaka.ecc.u-tokyo.ac.jp/gpsshogi/.