自己対戦棋譜を利用した半教師あり学習による将棋の評価関数の学習

林　伸也[†1]　浦　晃[†2]　三輪　誠[†3]
田浦　健次朗[†4]　近山　隆[†2]

近年将棋の評価関数の学習は, 熟練者の棋譜を用いた教師あり学習を用いて行われている. しかし, 人間同士の棋譜の数には限界があり, 人間同士の対局の棋譜のみを使っていては現れにくい局面が存在する. 本研究では将棋の評価関数の学習に自己対戦棋譜を利用した Self-Training を適用した. Self-Training は半教師あり学習手法の一つであり, それを用いることで熟練者の棋譜に加えて自己対戦棋譜中の信頼できる局面のみを選択的に学習する. Self-Training ではラベルなしデータとして自己対戦棋譜を用い, 最善手と次善手の評価値の差に着目して学習に用いる局面を選択した. 評価として, 熟練者の棋譜のみを用いて学習をおこなったプログラムとの対戦実験を行い, 最大で 56.8%の勝率を得ることができた.

Learning Shogi evaluation functions by semi-supervised learning

Shinya Hayashi ,[†1] Akira Ura ,[†2] Makoto Miwa ,[†3]
Kenjiro Taura [†4] and Takashi Chikayama [†2]

Recently, shogi evaluation functions are usually trained by using professional players' game records. However, the number of the game records is limited, and it is difficult to train the features in the positions which rarely appear in the game records. This research proposes a method to apply a self-training algorithm to train the evaluation functions. Self-training is a semi-supervised learning algorithm, and, with the algorithm, our method can train evaluation functions using reliable positions in self-play records, in addition to the professional players' game records. The reliable positions are selected by using the difference of the evaluation scores between the best move and the second best move. Our method is evaluated by comparing the player trained with our method to one trained with supervised learning. The experimental results show that the player trained with our method can acheive a 56.8% winning percentage.

1. はじめに

将棋やチェス, オセロなどのゲームにおいて, 局面の有利不利を静的に評価し, それを数値として返す評価関数によりコンピュータゲームプレイヤ形成判断を行う. 強いコンピュータゲームプレイヤを作るためには, この評価関数の精度を上げることが不可欠である. 精度の高い評価関数を設計するためには, ゲームの性質をよく表す特徴を抽出し, それらに適切な重みを付ける必要がある. 一般的には特徴の線形和が評価値として利用され, それをもとに局面の形勢判断が行われる. 昨今の将棋プログラムの評価関数では膨大な数の特徴が用いられているが, それらの特徴に対してプログラマが手調整で適切な重みを与えることは事実上不可能である. そのため, 近年では機械学習によって重みを自動調整する方法が広く使われており, 保木[11] による将棋プログラムは世界コンピュータ将棋選手権で優勝を果たした.

将棋の評価関数の学習には, 通常熟練した人間の棋譜が教師データとして用いられる. 一方で, 将棋プログラムを用いた対戦では, 人間同士の対戦では見られない特徴を持つ局面が現れることがある. そのような局面に対して人間の棋譜では学習が十分に行われず, 適切な評価が難しいと考えられる. 例えば熟練者の棋譜の中には入玉をする局面が少なく, そこから学習を行ったプログラムは入玉局面に弱いと言われている[11].

また, 教師データを使用しない学習方法というのも存在し, その一つの方法として強化学習がある[4]. 強化

†1 東京大学工学部電子情報工学科
Engineering Department, The University of Tokyo
hayashi@logos.ic.i.u-tokyo.ac.jp
†2 東京大学大学院工学系研究科
Graduate School of Engineering, The University of Tokyo
†3 マンチェスター大学
University of Manchester
†4 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

学習とは, エージェントが環境から得られる報酬を最大化するように行動を学習していくというものである. 強化学習のアルゴリズムとしては Temporal Difference 法[4] もしくはそれを改良したアルゴリズムがチェスなどのゲームで成果を上げている[9][2]. しかし将棋のように探索空間が非常に広い問題では, 強化学習だけで強いプレイヤを作成したという報告はない. また, 得られる解が最適解である保証もなく, 局所解に陥る可能性がある.

そこで近年では自己対戦により生成した棋譜を学習に利用するという研究がいくつか行われている[7][10]. これらの手法は自己対戦により生成した棋譜を使って学習を行うというものであり,5 五将棋, ブロックスデュオなどの将棋より探索空間の小さいゲームでは一定の成果を上げている. しかし, 将棋において自己対戦棋譜が有効であるかどうかは現在のところ知られていない. また, 自己対戦棋譜は熟練者の棋譜ほど信頼性の高いものではなく, 学習すべきではない局面というのが熟練者の棋譜に比べて多く含まれていると考えられる. 自己対戦棋譜を利用する際には, そういった局面を学習してしまうことで性能を低下させる危険が常に伴う.

本研究では, 将棋の評価関数の学習に自己対戦棋譜を利用し半教師あり学習である Self-Training を用いることにより, 人間の棋譜のみでは考慮されていなかったと考えられる局面に対しても学習を行い, 評価関数を改善することを目的とする. 自己対戦によりコンピュータゲームプレイヤ特有の局面を生成し, さらに半教師あり学習を用いることで自己対戦棋譜中の信頼できる局面のみを学習することにより, より強い将棋プレイヤーを作る.

2. 関 連 研 究

2.1 評価関数の自動学習

評価関数を自動的に学習するプログラムの代表として, 保木によって作成された Bonanza[11] がある. Bonanza では膨大な数の特徴に対して, 適切な重みの自動調整が行われる. Bonanza の大きな特徴は, 学習の際に数手先までの探索を行い, それによって得られた局面の評価値をもとに重みを調節するということである. 保木によって提案されたこの学習手法はボナンザメソッドという名前で知られており, 現在数多くの将棋プログラムで利用されている.

2.2 半教師あり学習

半教師あり学習とは, 正解が付与されたデータ (ラベルつきデータ) だけでなく, 正解が付与されていないデータ (ラベルなしデータ) も学習に利用するという学習手法である[6].

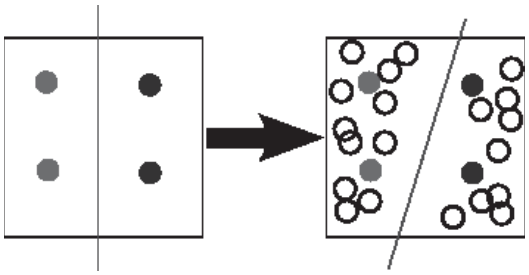

図 1 Self-Training

教師あり学習ではラベル付きデータのみを用いて学習を行うが, ラベル付きデータは大量に用意するのは困難であるとされる. 一方でラベルなしデータは比較的簡単に手に入れることができる. 半教師あり学習では, 少数のラベル付きデータと多数のラベルなしデータを用いることで, よい分類器を比較的簡単に作ることができる.

半教師あり学習にはいくつかの種類があり, これは扱う問題ごとに使い分ける必要がある. 半教師あり学習では使用するラベル付きデータは少数であるという前提のもとに行われるため, 各手法がそれぞれ強力な仮定の上に成り立っている. そのため, 各手法が置いている仮定が問題の構造に合致するような場合にのみ使用できる. 本稿では提案手法で利用している Self-Training について紹介する.

Self-Training は半教師あり学習の一種である. Self-Training はラベルなしデータのラベルを予測し, その予測ラベルを正解とみなしてラベルありデータに追加するという手法である. Self-Training の様子を図 1 に示す. 具体的には以下のようなステップにより行われる[6].

(1) 学習器は (少数の) ラベルつきデータを用いて学習を行う.
(2) この学習器を用いてラベルなしデータを分類する.
(3) ラベルなしデータのうち, 上の学習器による分類である程度高い信頼度で分類されたと判断された点をラベルつきデータに加える.
(4) 再び学習を行う. 1 から 4 を繰り返す.

Self-Training は適切に用いないと学習器の性能を下げる恐れがある. また収束条件もよく分かっておらず, 一般的な解析は難しい. しかし一方で, 既存の学習器をブラックボックスとして使用できること[8], どのような場合に「信頼度が高い」と判断するかの基準はある程

度自由に設定できることなどの利点がある.

Yarowsky らは語義曖昧性解消に Self-Training を用いた[5]. ここでいう語義曖昧性解消とは, 複数通りの意味に取れる英単語が, 現在の文脈でどちらの意味で使われているか特定するというタスクである. Yarowsky らの研究では, 自然言語が持つ以下の二つの特徴を利用することで, 手間のかかるラベル付けを回避した.

(1) 一つの連語の中には一つの意味しか現れない (One sense per collocation).

(2) 一つの会話・文書中には, 一つの意味しか現れない (One sense per discourse).

Rosenberg らは物体認識システムの学習に Self-Training を用いた[3]. Rosenberg らの研究では, 多数のラベル付きデータを用いて学習した既存のシステムと, Self-Training によって学習を行ったシステムがほぼ同等の性能を持つことを示した. またデータの選択方法について, 認識システムとは独立に最小二乗誤差をもとにデータを選択する方法が, 認識システムの自信の度合いによって選択する方法の性能を大きく上回ることを示した.

2.3 自己対戦の利用

2.3.1 自己対戦棋譜

近年, 熟練者の棋譜が存在しないゲームを中心に, 自己対戦の棋譜を学習に用いる研究が数は少ないが行われている.

柿木は, 5 五将棋において自己対戦の棋譜を用いた学習を行った. 柿木のプログラムは第一回 UEC 杯 5 五将棋大会で第二位の成績を残し, 自己対戦棋譜を利用した学習の有効性を示した[7]. 5 五将棋とは, 盤面を 5 × 5 マスにした将棋のことであり, 初期盤面, 相手の一段目に入ったときにのみ成れること以外は, 通常の将棋と同じである.

柿木は自己対戦による兄弟局面の学習を提案した. これは, 保木の方法[11] を基盤とし, 一般的に知られている「深く探索して得られた手は浅い探索で得られた手よりよい」という仮定の下で, 深い探索で得られる手を浅い探索でも得られるよう重みを調整する方法である.

使用する自己対戦の棋譜は, 探索の深さ・制限時間を変える, 乱数を使用する, 定跡をランダムに使用するなどして自己対戦を行うことでさまざまな種類のものを生成する. そうしてできた棋譜によって学習を行い, 再び棋譜生成をするというサイクルを繰り返した.

築地らは, 適切な評価関数の構成が確立していないブロックスデュオというゲームを題材に, 末端の勝敗情報を評価値に加える, ゲームの進行具合によって評価関数を変化させる, 思考時間を長くするなどの制御を加えることで, 柿木の手法の有効性を示した[10].

2.3.2 強 化 学 習

強化学習とは, エージェントが自らの経験をもとに, 環境から与えられる報酬が最大になるような行動を学習していくという手法である[4]. 強化学習は環境から与えられる報酬を適切に設定することができれば事前知識を必要とせずに学習を行うことができる.

強化学習のアルゴリズムとしては Temporal Difference(TD) 法[4] もしくはそれを改良したアルゴリズムがゲームにおいて用いられる. Baxter らは, TD(λ) において minimax 探索によって得られた局面をもとに静的な評価値を更新していく TD-leaf(λ) というアルゴリズムを提案し, 特にチェスにおいて性能が改善することを示した[2].

大崎らは,TD(λ) とモンテカルロ法を組み合わせた TD(λ)-MC 法という新しいアルゴリズムを提案し, ブロックスデュオにおいて TD(λ) と比べてわずかに高い性能を持つことを示した[9]. TD(λ)-MC 法とは, ある局面の静的な評価値をその局面から MC シミュレーションによって派生した末端局面の勝敗 (勝ちを 1, 引き分けを 0.5, 負けを 0 とする) の平均と比較することで, 静的な評価値を更新していくという手法である.

3. 提 案 手 法

本研究では将棋の評価関数の学習に, 自己対戦棋譜を利用した Self-Training を適用する. 自己対戦棋譜を学習に用いることの有効性は将棋よりも探索空間の小さいゲームにおいては示されているが[7][10], 将棋の評価関数の学習に有効であるかは分かっていない. また, 前述の通り自己対戦棋譜は熟練者の棋譜ほど信頼できるものではない. そこで本研究では自己対戦棋譜を半教師あり学習におけるラベルなしデータとして用いることで, 自己対戦棋譜の中でもある程度信頼できる局面のみを選択的に学習する. 半教師あり学習の具体的手法として Self-Training を用いたのは, Self-Training は前述の通り対象とする学習器をブラックボックスとして適用できるため, 既存の学習器に適応しやすいと考えたからである.

提案手法は自己対戦棋譜と熟練者の棋譜の両方を用いるという点で自己対戦棋譜を利用した他の研究とは異なる. Self-Training のラベルなしデータとしては, 人間同士の対局では現れない局面が数多く出現するようなものであればよいのだが, 容易に大量の棋譜を生成することができるという理由で自己対戦棋譜を用いた.

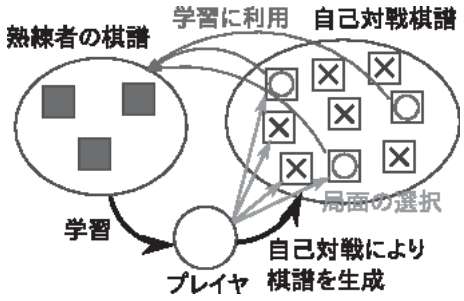

図 2 提案手法の流れ

3.1 将棋における Self-Training の適用

Self-Training を将棋に適用する場合, ラベルありデータは熟練者の棋譜, ラベルなしデータは自己対戦棋譜である. 実際の学習の流れを図 2 及び以下に示す.

(1) 熟練者の棋譜を用いて評価関数の学習を行う.
(2) この評価関数を用いて, 自己対戦により生成した最善手が未知の局面の指し手を決める.
(3) 生成した棋譜のうち, ある程度高い信頼度で指し手が決まる局面の棋譜を学習データセットに含める.
(4) 再び学習を行う. 1 から 4 を繰り返す (今回は行っていない).

将棋の評価関数の学習に提案手法を適用することが, 定性的にはどのようなことを意味するのかを説明する. まず熟練者の棋譜によって評価関数の学習を行うことで, 人間同士の対局でよく現れる特徴に対する重みを学習する. 次にその評価関数を用いて自己対戦棋譜中の信頼度の高い局面を選ぶ段階で, 熟練者の棋譜での学習の結果重要だと判断された特徴を持つ局面が選ばれる. そのような局面においては, そこに現れている他の特徴に関しても同じく重要であるという仮定を置き, 重みを更新していく. 自己対戦棋譜を使用している以上, ここでいう他の特徴の中には人間同士の対局では現れにくい局面, 特徴が含まれていると考えられ, 結果としてこのような特徴に対しても十分に学習が行われるようになることが期待できる. またこれとは逆に, ある特徴が複数回現れて, その度に評価がばらばらになるようなことがあれば, その特徴の重要度は低いと判断することもできる. 以上が提案手法の考え方である.

3.2 信頼度の基準

指し手を選ぶ際に, もし最終的に選ばれることになる評価値の最も高い手と他の合法手の評価値の間に大きな差がない場合, 学習に用いた棋譜数やそれらを学習した順番などの些細な要因によって指し手が容易に変化してしまうことが考えられる. つまり, そのような局面で選ばれた手が真に最善である確率は低い. そこで本研究では以下のようにして信頼度の高い局面を選ぶ.

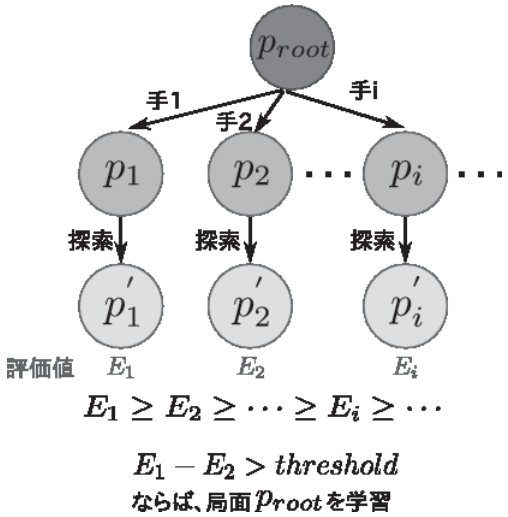

図 3 最善手と次善手の評価値差に着目した局面の分類

(1) ある局面における全ての合法手に対して, そこから学習の深さより一段浅い探索を行う. 末端評価は熟練者の棋譜のみを用いて学習した評価関数を用いて行う.
(2) それによって得られた評価値をそれぞれの手の評価値として, 一番よいと判断された手と, 二番目によいと判断された手の評価値を求める.
(3) これらを比べてある閾値以上の差があったとき, この局面は高い信頼度で指し手が決まったとする.

提案手法の流れを図 3 に示す. このようにすることでプレイヤが自信を持って手を指した局面のみを学習でき, 真の最善手を学習できる確率が高くなると考えられる.

4. 評　　価

4.1 評 価 方 法

自己対戦棋譜から抽出した信頼度の高い局面に加え, 熟練者の棋譜を 30,000 棋譜学習させたプログラムについて, 信頼度の閾値を変えてベースラインプログラムとの対戦実験を行った. 結果を図 5 に示す. 学習にはベースラインプログラムと同様深さ 6 のボナンザメソッド[11] を用いた. 学習に用いる自己対戦棋譜は, 深さを 6 に設定したベースラインプログラム同士で対戦を行うことで生成した. ベースラインプログラムと提案手法の対戦実験は, 持ち時間は一手 10 秒とし, 先手・後手を交互に入れ替えて 200 戦行った. また, 各条件

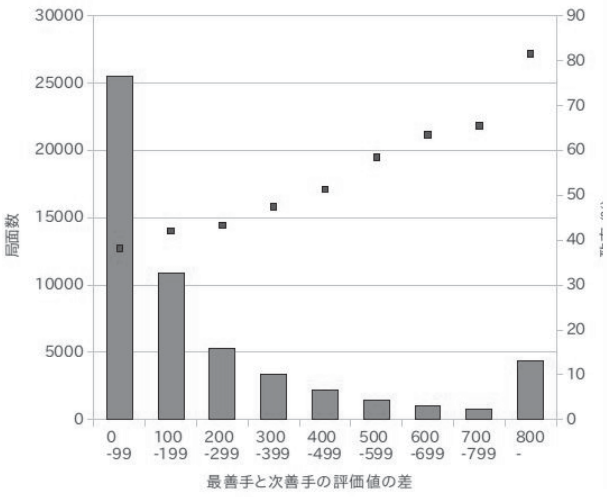

図 4 評価値の差と一致率

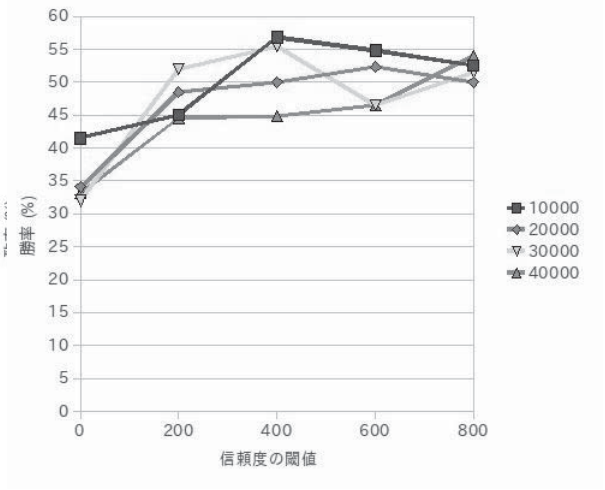

図 5 棋譜数を揃えた場合の対戦結果

における指し手の一致率を, 熟練者の棋譜を 500 棋譜用いて計算した.

対戦実験は棋譜数を揃えて学習を行った場合と局面数を揃えて学習を行った場合の二通り行った. 棋譜数を揃えて学習を行う場合, 閾値によって自己対戦棋譜中で実際に使われる局面数が異なる. 各条件において, 実際にどれくらいの割合の局面が学習に使われているか調べた.

4.2 閾値の決定

まず, 信頼度の閾値をどれくらいの値するのが妥当かという調査を行った. 熟練者の棋譜を 500 棋譜用いて, 最善手と次善手の評価値の差と指し手の一致率の関係, 及び各評価値の差についてどれくらいの数の局面が現れるのかを調べた. 一致率は将棋プログラムの性能評価で一般的に用いられる指標であり, この値がベースラインプログラムを上回るような自己対戦棋譜のみを用いるようにすれば性能向上が見込めると予想し, このような実験を行った.

実験では東京大学近山・田浦研究室で開発された将棋プログラム「激指」[1] を使用した. 一致率を算出する際には, 熟練者の棋譜を 3 万棋譜使って学習したプログラム (ベースラインプログラム) を用いた. また, 学習は深さ 6 のボナンザメソッドにより行った.

評価値の差と指し手の一致率の関係を図 4 に示す. ベースラインプログラムの一致率を上で用いた 500 棋譜に対して調べた結果, 42.5%という結果が得られた. これに対して, 最善手と次善手の評価値の差が 100 から 199, 200 から 299 となる局面での一致率はそれぞれ 42.1%, 43.3%となった. つまり信頼度の閾値が 200 以上 (激指の評価関数で歩 1 枚得程度の点数) であればベースラインプログラムの一致率を越えるため, そのような場合は性能向上が見込めると予想し, まず閾値 200, 及び比較のために閾値 0 で学習を行ったものを用いてベースラインプログラムとの対戦実験を行うことにした. しかし, 閾値 200 で性能が向上するというのはあくまで予想であり, 他の値との比較も行う必要があると考え, 最終的には閾値の値は 0, 200, 400, 600, 800 の 5 通りで対戦実験を行うことにした.

4.3 自己対戦棋譜数を揃えた対戦実験

実験結果を図 5, 表 1, 表 2 及び図 6 に示す. 閾値 0 に注目すると, 使用する自己対戦棋譜数が 1 万の場合と比べて 2 万以上の場合では勝率が 5%以上下がっている. このことから, 自己対戦棋譜をそのまま学習に用いるだけでは性能向上は見込めず, 何らかの選別を行う必要があるという予想が正しいものだったと言える. また, どの場合においても閾値 0 の場合が一番勝率が低くなっていることから, 閾値以下の局面を不要であると考え捨てるという手法が有効であったと予想される. しかし, これだけでは「自己対戦棋譜は将棋の場合有効ではなく, 他の閾値に比べ閾値 0 の場合の勝率が低いのは単に自己対戦棋譜中で学習に使用される局面数が多く, 悪い影響を多めに受けた」という可能性を排除しきれない. そのためこれに関する結論は 4.4 で出すことにする.

閾値に関して一つ言える事は, 閾値を大きくするほど結果が良くなるわけではないということである. このことから, 最善手と次善手の評価値の差が大きすぎるような局面も何か良くない要素を含んでいるということになるが, その原因ははっきりしていない. 考え

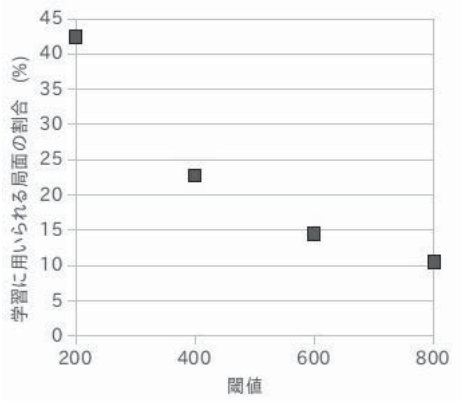

図 6 自己対戦棋譜中の学習に用いられる局面の割合の変化

られることとしては, 最善手と次善手の評価値の差が大きいような局面というのは大駒が絡んだ局面であることが多く, そのような局面を学習データセットに追加すると学習局面が偏ってしまい, 大駒に関係しない特徴に対して学習がうまく行われないという可能性がある.

自己対戦棋譜についても同じで, 単に多く使えば性能が上がるというわけではないという結果が得られた. このことから, 現状の自己対戦棋譜中の信頼度の高い局面の選び方は完璧ではなく, 学習すべきではない局面をある程度選んでしまっているということが考えられる. ただ, 閾値 800 のときは自己対戦棋譜数が 4 万のときの勝率が最も高くなっていることから, 利用する棋譜数をもっと増やした場合に閾値 800 においては勝率が上がるということも考えられる.

グラフに関して, 全体的にぶれが大きいということが言える. 特に自己対戦棋譜が 3 万棋譜で閾値 600 のときの勝率が不自然に下がっていることが分かる. このことから, 対戦実験の回数が 200 というのは少なく, 勝率がまだ安定していないと考えられる.

勝率については, 閾値 400, 自己対戦棋譜数 1 万のときに最大で 56.8%という結果を得ることができた. これは p 値 0.5 の二項検定の有意水準である 57.5%には届いていないが, 勝ち越すことはできている.

一致率については, 特に規則性を見出せず, 棋譜数が多いほど一致率がわずかに高い程度だった. 本研究では, 自己対戦棋譜を用いて人間の指し手には現れにくい特徴を獲得することを目的とした学習方法を適用したので, 熟練者の棋譜を用いて算出した一致率が特に向上しなかったのは妥当な結果であると言える. これらの結果から言えることは, 熟練者の棋譜との指し手の一致率が必ずしも性能を反映しているわけではないということである. 指し手の一致率が現在将棋プログラムの性能評価に広く用いられている中で, このような結果が得られたのは重大なことであると言える.

4.4 局面数を揃えた対戦実験

4.1 で述べた通り, 棋譜数を揃えた対戦実験では同じ棋譜数を用いても実際に学習に使われる局面の数は閾値によって異なる. そこで, 閾値ごとに学習に使用する局面数を揃えて学習を行った場合の対戦実験を行った.

対戦実験の結果を図 7, 表 3, 表 4 に示す. なお, 用意した自己対戦棋譜数の不足により, 閾値 800 は 50 万局面, 閾値 600 は 75 万局面までしかデータが取れていない. 対戦実験の結果を見ると, 局面数 25 万のときは数が少なすぎて結果がばらけてしまっているが, 局面数 50 万のときは閾値が高い順に並んでいる. さらに局面数が 75 万,100 万のときは中間の値である閾値 400 が最も高い勝率を記録しており, これは 4.3 の結果とも合致する. 局面数が 75 万のときに閾値 0 の勝率が高すぎるのは明らかに不自然であり, より多くの実験を試す必要がある.

ここで 4.3 で保留した「閾値以下の局面を不要と考えて切り捨てるという本手法は妥当であったか」とい

表 1 棋譜数を揃えた場合の対戦結果

	0	200	400	600	800
10,000	41.5	45	56.8	54.8	52.5
20,000	34.1	48.5	50	52.3	50
30,000	32	52	55.5	46.5	51.5
40,000	33.2	44.6	44.9	46.5	54

表 2 棋譜数を揃えた場合の一致率の比較

	0	200	400	600	800
10,000	44.0	42.8	42.8	43.0	44.0
20,000	44.8	45.2	42.0	43.0	43.4
30,000	42.8	43.6	44.6	43.0	44.6
40,000	45.4	45.8	43.6	43.6	44.4

表 3 局面数を揃えた場合の対戦結果

	25 万	50 万	75 万	100 万
0	52.3	40.7	50.5	54.3
200	50.8	46.7	46.2	50
400	44.5	50.5	53.5	50.8
600	49	51.3	49.5	-
800	45.7	52.8	-	-

表 4 局面数を揃えた場合の一致率の比較

	0	200	400	600	800
25 万	43.4	43.2	43.4	45.4	43.2
50 万	44.6	43	44.4	43.6	43.0
75 万	43.8	43.6	44.6	42.2	44.2
100 万	43.0	44.8	45.2	44.2	44.2

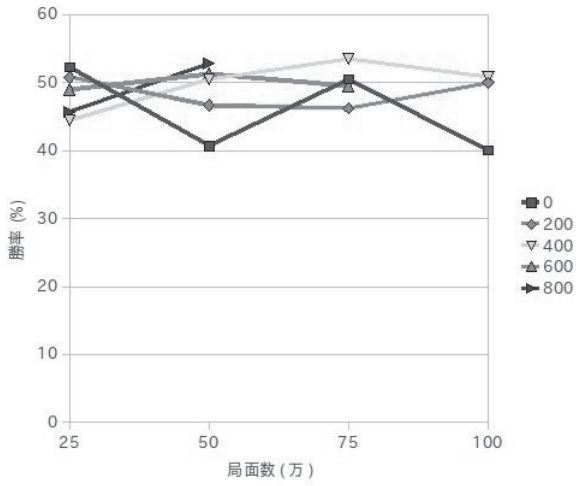

図 7 局面数を揃えた場合の対戦結果

う問題だが, 局面数を揃えて学習した場合においても閾値を設けた場合の方が勝率が高くなっていることから, これは有効であったと考えられる.

5. おわりに

本研究では自己対戦棋譜を用いた Self-Training を将棋の評価関数に適用し, 人間の棋譜のみでは十分学習できない特徴を学習することで, 評価関数を改善した. 評価として熟練者の棋譜のみを使って学習を行ったプログラムとの対戦実験を行った. また, 各種条件における指し手の一致率を調べた. 対戦実験の結果, 有意な勝率ではないにしても最大で 56.8%という勝率を得ることができた. また, 指し手の一致率は既存のプログラムと大きな差はなかった.

今後の課題としては, まず閾値及び自己対戦棋譜数に対して勝率が単調増加になっていない原因を追求するということが挙げられる. そのために, まず閾値と自己対戦棋譜数をもっと増やした場合に勝率がどう変化するのか, また実際に学習に使われている局面にどのような特徴が現れているのかという調査を行う.

さらに, 今回は自己対戦棋譜を生成するときの探索の深さを 6 としていたが, これを変化させた場合の勝率の変化も調査する予定である. また繰り返し学習に関しても今回は行わなかったので, 一回の学習に関する調査が終了したら今度は繰り返し回数に対して性能がどのように変化するのかを調べてみる予定である.

もう一つ重要な事として, 自己対戦棋譜中の局面の選び方が挙げられる. 実験の結果, 現在行っている信頼度の高い局面の選び方が必ずしも最善ではないという結論が得られた. そのため, より正確に局面の信頼度を求める方法を考える必要がある. また, 本研究では熟練者の棋譜に現れにくい局面を得ることが目的で自己対戦棋譜を使用しているが, 提案手法は自己対戦棋譜以外の棋譜にも適用可能であり, この目的をもっと直接的に達成できるような棋譜の選別方法を導入し, 現在行っている方法との比較を行うということも考えている.

参 考 文 献

1) 将棋プログラム「激指」. http://www.logos.t.u-tokyo.ac.jp/~gekisashi/index.html.
2) Jonathan Baxter, Andrew Tridgell, and Lex Weaver. TDLeaf(λ): Combining temporal difference learning with game-tree search. *In Proceedings of the 9th Australian Conference on Neural Networks (ACNN-98)*, pp. 168–172, 1998.
3) Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. *Seventh IEEE workshops on application of computer vision (WACV/MOTION'05) - Volume 1*, pp. 29–36, 2005.
4) R.S.Sutton and A.G.Barto. *Reinforcement Learning:An Introduction*. MIT Press, 1998.
5) David Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. *In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, pp. 189–196, 1995.
6) Xiaojin Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.
7) 柿木義一. 5 五将棋における評価関数の自動学習, 2008.
8) 小町守. 半教師あり学習チュートリアル, 2008.
9) 大崎泰寛, 柴原一友, 但馬康宏, 小谷善行. TD(λ)-MC 法を用いた評価関数の強化学習. 第 12 回 ゲームプログラミング ワークショップ 2009, pp. 36–43, 2007.
10) 築地毅, 小谷善行. 自己対局による兄弟局面学習における汎用的制御の有効. 第 14 回 ゲームプログラミング ワークショップ 2009, pp. 127–134, 2009.
11) 保木邦仁. 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 2006, pp. 78–83, 2006.