将棋の局面評価関数におけるディープラーニングの利用

和田悠介[†1]　五十嵐治一[†1]

概要：コンピュータ囲碁ではディープラーニングが有効であることが分かり，コンピュータチェスにおいても局面評価関数の学習に利用されてきている．その適用例として，Deep Pink と Giraffe がある．前者はビット列で表現された盤面情報を入力とする教師付き学習を，後者は特徴量で表現された盤面情報を入力とする強化学習を用いている．本研究では，これらの手法をコンピュータ将棋へ適用した学習実験を行ったので，その結果を報告する．

キーワード： コンピュータ将棋，評価関数，ディープラーニング

Deep Learning for Positional Evaluation Function of Shogi

YUSUKE WADA[†1]　HARUKAZU IGARASHI[†1]

Abstract: Deep Learning has been proved to be very effective in computer Go and was applied to learn positional evaluation functions in computer chess. Deep Pink and Giraffe are ones of such examples. Deep Pink used supervised learning whose input are bit data representing piece positions. Giraffe used reinforcement learning whose input are characteristics extracted from piece positions. In this paper, we applied the two kinds of learning methods to computer shogi and report the results of our experiments.

Keywords: Computer shogi, Evaluation function, Deep learning

1. はじめに

近年，深層ニューラルネットワーク（DNN:Deep Neural Network）を用いた機械学習手法，ディープラーニングがコンピュータ囲碁やコンピュータチェスで成果を挙げている．コンピュータ囲碁では，深層畳み込みニューラルネットワークを打ち手選択と局面評価に用いたプログラム「AlphaGo」[1]がプロ棋士を超える棋力となった．またコンピュータチェスでは，DNN を教師あり学習に用いたプログラム「Deep Pink」[2]と強化学習に用いたプログラム「Giraffe」[3]がある．Deep Pink の局面評価関数には，盤面をビット列で表現した入力層の DNN が用いられ，教師あり学習により学習が行われた．一方，Giraffe の局面評価関数では，入力層において人間の知識を利用した特徴量が用いられた．また，特徴量を低レベル層で混ぜず，別々に伝播させるネットワーク構造を提案し，これに TD-Leaf(λ)法[4]による強化学習を適用した．

しかし，コンピュータ将棋では大規模なニューラルネットワークを用いた局面評価関数の研究報告はそれほど行われていない．ニューラルネットワークを用いた非線形評価関数に対し TD(λ)法を適用した研究[5]があるが，ニューラルネットワークのサイズが非常に小さい．また，評価関数に 3 層パーセプトロンと類似した構造を持つプログラム「習甦」[6]では，Bonanza メソッド[7]として用いられている棋譜との不一致度に加え，勝率予測と勝敗との負の対数尤度，および深さの異なる探索結果に対する分散の線形和を最小化させる学習を行っているが，ネットの層数が少なく，本格的なディープラーニングとは言えない．

そこで，本研究ではコンピュータチェスにおけるディープラーニングを用いた研究である Deep Pink と Giraffe の手法を将棋に適用することを試みた．

2. チェスにおけるディープラーニングの利用

2.1 Deep Pink：教師あり学習

図 1 に Deep Pink のモデルを示す．

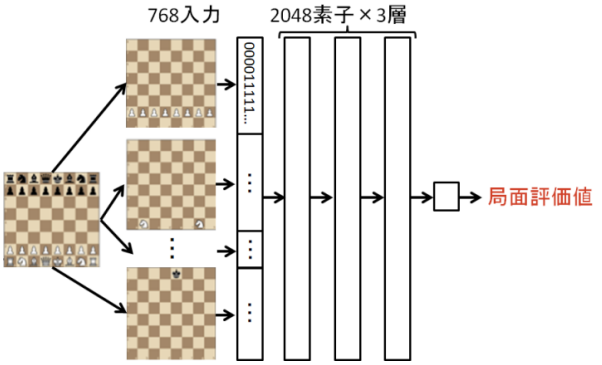

図 1　Deep Pink の DNN モデル

Figure 1　DNN model of Deep Pink

Deep Pink では，チェスにおける 6 種類の駒それぞれに対し，盤面のある升目にその駒があれば 1，なければ 0 とし

†1 芝浦工業大学
Shibaura Institute of Technology.

た 64×6×2（白番と黒番）の計 768 ビット列を抽出し DNN の入力層とする．次の中間層は 3 層で，それぞれ 2048 素子になっており，活性化関数には ReLU を用いている．最後の出力層でスカラーの局面評価値が出力される．また，これらは全結合層であり，それぞれにバイアス項が存在する．

Deep Pink の学習方式では，確率的勾配降下法によって以下の損失関数の最小化を行う．

$$L(p,q,r) = -\frac{1}{m}\sum_{i=1}^{m}\Big[\log\Big(S\big(f(q_i)-f(r_i)\big)\Big) + \kappa\log\Big(S\big(f(p_i)+f(q_i)\big)\Big) + \kappa\log\Big(S\big(-f(q_i)-f(p_i)\big)\Big)\Big] \quad (1)$$

損失関数の引数 p，q，r はそれぞれ親局面，親局面で教師が指した後の局面（教師局面），親局面でランダムに選択した手を指した後の局面（ランダム局面）の集合である（図 2）．図 2 の例では，（親局面 A，教師局面 A，ランダム局面 A），（親局面 A，教師局面 A，ランダム局面 B），（親局面 A，教師局面 A，ランダム局面 C）という 3 つの局面集合ができる． i がある 1 局面を指す添え字で，$f(x)$がネットワークの出力，すなわち局面の評価値である．また，m はバッチサイズ，κ は定数，S はシグモイド関数である．教師局面とランダム局面の評価値の差を大きくすることで教師局面の評価を高め，親局面と教師局面の評価値が等しくすることで，評価値変動の少ない滑らかな局面推移となる．

Deep Pink における実際の実装とは，式(1)の評価値の符号がそれぞれ異なるが，これは黒番である局面を白番から見たものに反転して計算を行っているためである．

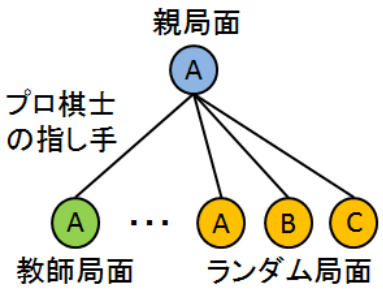

図 2　Deep Pink の学習方式で用いる局面の例

Figure 2　Positions used by Deep Pink for learning

Deep Pink の学習棋譜には，FICS（Free Internet Chess Server）[8]からダウンロードしたアマチュアプレイヤの 1 億棋譜が用いられ，m=2000，κ=1.0，学習率を 0.03 として学習実験が行われた．ただし，学習率は経過時間と共に減少させる．この学習の結果，同じ python で書かれたチェスプログラム「Sunfish」[9]に 3 割程度勝利したことが報告されている[3].

2.2　Giraffe：強化学習

Giraffe の入力層の出力ベクトルは，Deep Pink のような駒位置のビット列ではなく，手番，駒の個数，駒の可動性，攻め駒と守り駒のマップといった盤面の特徴を 0 から 1 の実数で表現されている（図 3）．ただし，攻め駒と守り駒のマップには Stockfish[10]の駒価値が利用されている．特徴表現の総数は 363 であり，Deep Pink と比較するとかなり小さく，評価値計算にかかる時間が短縮できる．

ネットワークモデルも Deep Pink のような全結合層ではなく，特徴を 3 種類に分けて別々に最初の中間層へ伝播させ，その後全結合層へ伝播させる．これは，特徴表現をモデルの高いレベルで混ぜた方が効果的であり，また過学習を防ぐための工夫である．出力層では tanh を用いることにより[-1, 1]の値が出力される（図 4）．中間層の素子数は 3 種類の特徴表現の個数に一定の割合を掛けることで定められる．ただし，図 3 中の 64 素子は固定値である．

特徴	内容	表現	個数
手番	現在の手番	0，1	1
キャスリング	先後のキングでキャスリングが行われたか	0，1	8
駒の個数	盤上の駒(キングを除く5種類)の個数	0～1	10
駒の有無と位置	駒の有無と、もしあるならどの座標にあるか	0～1	96
駒の移動性	先後のクイーン(8方向)，ルーク(4方向)，ビショップ(4方向)の移動できる場所の数	0～1	48
	先後のルーク，ビショップ，ナイトが安全に移動できる場所の数	0～1	12
攻め駒と守り駒のマップ (Stockfishの駒価値を利用)	キングを除く、各駒の位置に利いている駒の中で、駒価値の最も小さい駒の価値	0～1	60
	それぞれの升目(64個)に利いている駒の中で、最も価値の小さい駒の価値	0～1	128

図 3　Giraffe の特徴量

Figure 3　Feature representations of Giraffe

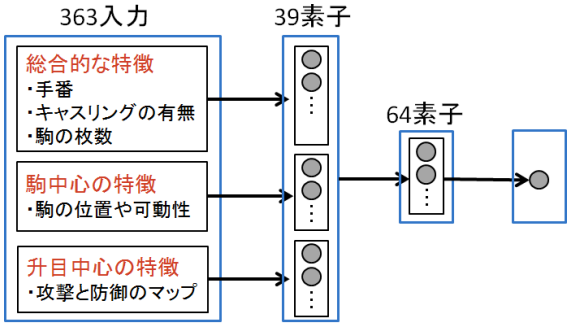

図 4　Giraffe の DNN モデル

Figure 4　DNN model of Giraffe

このモデルに対し，以下の TD-Leaf(λ)法の更新式を適用する[4].

$$\omega := \omega + \alpha\sum_{t=1}^{N-1}\nabla J(x_t,\omega)\left[\sum_{j=t}^{N-1}\lambda^{j-t}d_t\right] \quad (2)$$

$$d_t = J(x_{t+1},\omega) - J(x_t,\omega) \quad (3)$$

ωはモデルの重み，αは学習率，$\nabla J(x_t,\omega)$は局面 x_t におけるモデルの勾配であり，d_t はその局面における TD 誤差である．Giraffe では，初期局面から終局までを自己対局で行うのではなく，用意した棋譜の途中局面を 1 手ランダムに進めた後に 12 手の自己対局を行う．学習に用いる棋譜はコンピュータの対局棋譜であり，総局面数は 500 万局面である．途中局面から 1 手ランダムに進めることは，学習局面を増やす工夫である．Giraffe は 1 イテレーションで 500 万局面から 256 局面ランダムに選択して上記の TD-leaf(λ)法を適用し，これを 6 万イテレーション行った．その結果，STS（Strategic Test Suite）[11]の成績では Stockfish 等のチェスプログラムと同等程度となり，レーティング[12]では International Master レベルの棋力を実現した．

3. 将棋への適用のための変更点

3.1 Deep Pink の場合

Deep Pink の DNN モデルを将棋に適用する上で変更した点は，入力層の大きさである．将棋では盤面が 81 マス，駒が 14 種類，持ち駒が 7 種類存在するため，81×14×2（先手と後手）＋ 7×2 で計 2282 個の入力層となる．また，持ち駒については 0 と 1 のビット表現ではなく，それぞれの枚数をそのまま特徴量とした．さらに，入力層の増大に伴って中間層の素子数を 4096 に変更した．図 5 に将棋に適用した最終的なモデルを示す．また，学習方式は式(1)と同一のものを使用した．

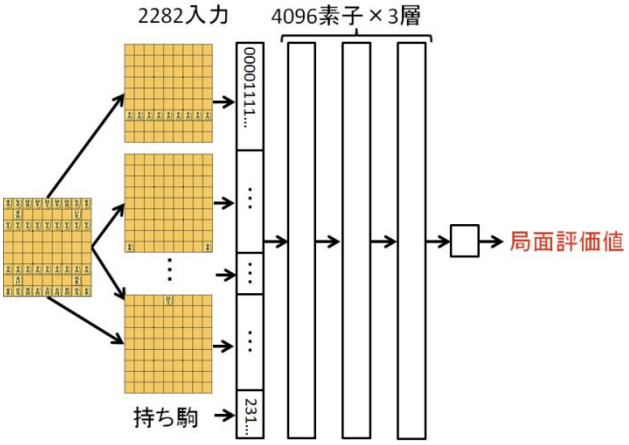

図 5　将棋に適用した Deep Pink の DNN モデル
Figure 5　DNN model of Deep Pink applied to Shogi

3.2 Giraffe の場合

Giraffe の特徴量を将棋に適用する上で，いくつか変更の必要があった．例えば，駒の個数は，チェスでは終盤に近づくにつれ少なくなるため，進行度の役割を担う．しかし，将棋では持ち駒を打つことによって盤面に駒を増やすことができるため，事情が異なる．そこで，技巧[13]の進行度を利用した．また，攻め駒と守り駒のマップには，Bonanza の駒価値を歩の価値が 100 になるように調整して用いた．最終的に盤面から抽出する特徴量の個数の合計は 640 となった．図 6，図 7 に将棋に適用した Giraffe の特徴量と DNN モデルを示す．　また，学習方式は式(2)と同一のものを使用した．

特徴	内容	表現	個数
手番	現在の手番	0，1	1
進行度	局面の進行度	0～1	1
駒の有無と位置	駒の有無と、もしあるならどの座標にあるか	0～1	202
持ち駒	7種類の持ち駒の枚数	0～1	14
駒の移動性	先後の香車, 角, 飛車, 馬, 龍の移動できる場所の数	0～1	80
	先後の香車, 桂馬, 銀, 金, 角, 飛車, 馬, 龍が安全に移動できる場所の数	0～1	48
攻め駒と守り駒のマップ (Bonanzaの駒価値を利用)	玉を除く、各駒の位置に利いている駒の中で、駒価値の最も小さい駒の価値	0～1	132
	それぞれの升目(81個)に利いている駒の中で、最も価値の小さい駒の価値	0～1	162

図 6　将棋に適用した Giraffe の特徴量
Figure 6　Feature representations of Giraffe applied to Shogi

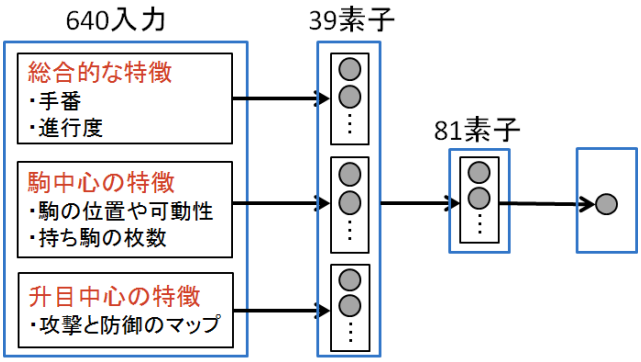

図 7　将棋に適用した Giraffe の DNN モデル
Figure 7　DNN model of Giraffe applied to Shogi

4. Deep Pink の学習方式の適用結果

4.1 学習実験の環境

将棋に適用した Deep Pink の学習方式を Python で実装し学習を行った．本実験における局面データは，プロの棋譜 62454 局に存在する約 700 万局面から親局面，教師局面，ランダム局面をそれぞれ抽出し使用した．ランダム局面は，図 2 の例において，親局面から 5 手ランダムに進めることで作成した．つまり，1 局面につき，（親局面，教師局面，ランダム局面）のデータセットが 5 つできる．最終的にデータセットの総数は 3200 万となった．しかしながら，この 1 割である 320 万セットをテストセットとして使用したため，学習データセットには 2880 万セットを使用した．

学習データセットを使用して，約 23000 エポックの学習を行った．ただし，1 エポックは学習データセット 1 周分

を指し，それを終えるごとに学習データセットをシャッフルした．また，損失関数のパラメータは m=1000，κ=1.0 とし，学習率は 0.01 で固定した．

4.2 損失関数の値の推移

4.1 の学習実験における，学習時の損失関数の値（Train loss）とテスト時の損失関数の値（Test loss）の推移を区間数 1000 の区間平均で図 8，図 9 に示す．学習エポック数が増えるにつれ Train loss が減少していることがわかる．また，それに伴って Test loss も減少しているため，一般性の高い結果が得られたことがわかる．

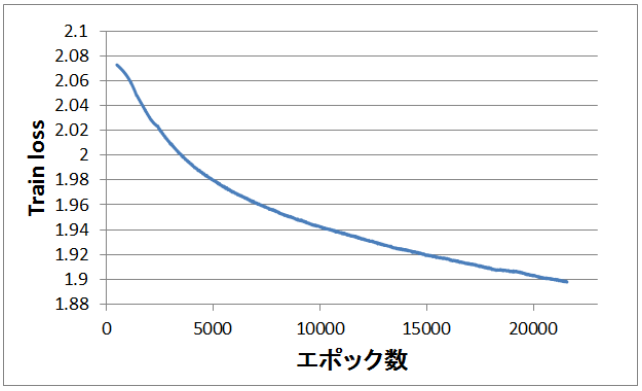

図 8　エポック毎の Train loss の推移

Figure 8　Change of Train loss per epoch

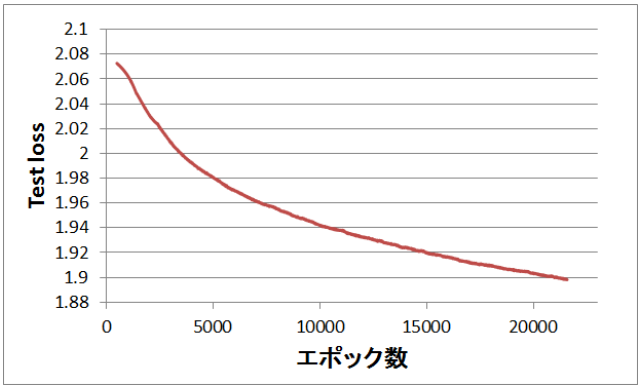

図 9　エポック毎の Test loss の推移

Figure 9　Change of Test loss per epoch

4.3 棋譜との一致率

一致率の計算は，学習で用いた局面において 1 手探索を行い，求められた最善応手とプロ棋士の指し手が同一のものだった場合に一致とみなし，10 万局面分の計算を行った．

未学習の評価関数と学習後の評価関数を使用して，教師として用いた局面との一致率を計算した．未学習時は約 6.91%であった一致率が，学習後は約 15.9%まで上昇した．一致率の上昇が約 15.9%に留まったのは，序盤の定跡形は学習できているものの，中盤以降の複雑な局面における一致率が低かったためと考えられる．さらなる一致率の向上には，エポック数を増やした学習実験を行うことや，ネットワーク構造の変更が必要であると考える．

4.4 探索エンジンへの組み込み

学習後の評価関数を Stockfish 8 ベースの自作プログラムに実装し，芝浦将棋 Jr.[14]との対局を行った．自作プログラムへ実装したのは，芝浦将棋 Jr.への実装が複雑だったためである．なお，対局実験で使用した芝浦将棋 Jr.は，2015 年第 3 回将棋電王トーナメントのバージョン[a]である．

対局では，学習後のプログラムは芝浦将棋 Jr.に対して勝つことができなかったが，対局時の局面に学習の効果が表れた．学習前は飛車先を受けずに駒損をしてしまう場面が序盤から見られたが（図 10），学習後はプロ棋士の定跡と似た形を指し，簡単な駒損はなかった（図 11）．

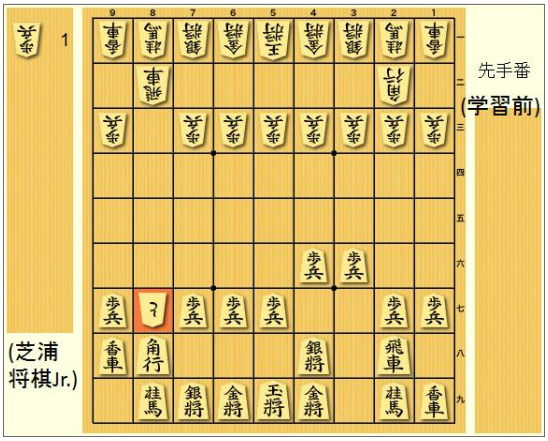

図 10　学習前の対局で現れたある局面

Figure 10　A position that appeared in a game of before learning

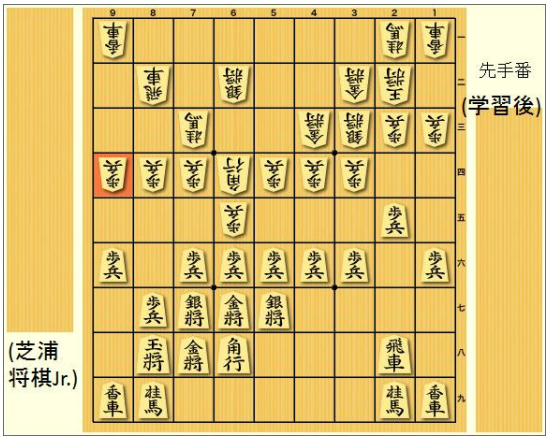

図 11　学習後の対局で現れたある局面

Figure 11　A position that appeared in a game of after learning

しかしながら，学習後のプログラムも中盤以降に駒損する手を多く指した．これは，中盤以降の局面の学習が不十分

a）28 チーム中 16 位．Bonanza 6.0 の評価関数を利用．

であることや，入力層のビット列で駒の価値が考慮されていないことが原因だと考えられる．また，評価関数の計算に時間を費やし，深い探索が行えないことにも原因があると考えられる．これらの対策には，教師として用いる棋譜を増やして大規模な学習を行うこと，入力層を 0 と 1 のビット列ではなく，駒の価値を正規化した-1〜1 の実数値列を用いることや，GPU を利用して評価関数の計算を高速化することが考えられる．

5. Giraffe の学習方式の適用結果

5.1 学習実験の環境

将棋に適用した Giraffe の学習方式を C++で実装し，学習実験を行った．本実験では，プロの棋譜 62454 局から抽出した約 700 万局面からランダムに 1000 局面取り出し，Giraffe の手法と同様にそれぞれ 1 手ランダムに進めた後に 12 手の自己対戦を行った．この 1000 局面の自己対戦による学習を 1 イテレーションとして 6000 イテレーションの学習を行った．なお，式(2)における学習率 α を 1.0，λ を 0.7 とした．

5.2 誤差の推移

5.1 の学習実験における誤差の推移を図 12 に示す．この誤差は，(2)式において，12 手の自己対戦で計算された λ と TD 誤差の積の和を 1000 局面分計算し，その平均をとったものである．図 12 からイテレーション数の増加と共に誤差が減少し，学習が行われたことがわかる．しかしながら，最終的な誤差は約 0.07 に留まり，さらなる減少は見られなかった．

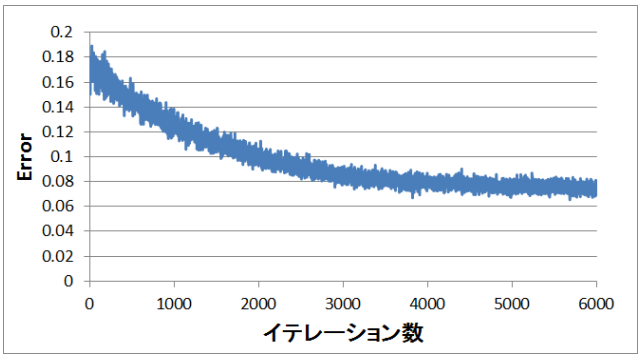

図 12　イテレーション毎の誤差の推移

Figure 12　Transition of Error per iteration

5.3 局面の定性的評価

本実験では 4.4 同様，学習後の評価関数を用いたプログラムと芝浦将棋 Jr.の対局を行い，出現した局面の評価を行った．対局は 1 スレッドの 1 手 10 秒で 100 局行い，結果は学習後のプログラムが約 35%勝利した[b]．図 13 に対局に現れた局面の 1 つを示す．先手が学習後のプログラム，後手が芝浦将棋 Jr.である．序盤では，Deep Pink の学習方式を将棋に適用した場合と異なり，プロ棋士の対局で出現するような定跡形にならず，常に中住まいといった玉を中心に囲う形を好んで指した．また，序中盤での不自然な駒損も少なかった．これは，図 6 の特徴量に駒価値を加えた効果だと考えられる．

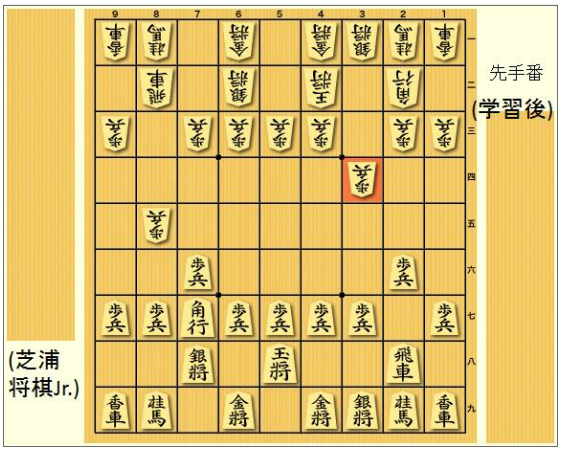

図 13　学習後の対局で現れたある局面

Figure 13　A position that appeared in a game of after learning

6. おわりに

本研究では，コンピュータチェスにおけるディープラーニングを利用した先行研究をコンピュータ将棋へ適用した．

Deep Pink の学習方式(教師あり学習)を将棋に適用した場合，序盤はプロ棋士と同様の定跡を指したが，中盤以降の駒損で形勢を損なうことが多かった．Giraffe の学習方式(強化学習)を将棋に適用した場合，定跡は学習しないが異なる独自の形を好んで指し，序中盤の不自然な駒損は見られなかった．

今後，将棋に適用した Deep Pink の学習方式では，より多くの棋譜を用いた大規模な学習を行うことや，駒価値が自然に学習できる特徴量の工夫を行うことが必要である．将棋に適用した Giraffe の学習方式では，チェスにはない囲いや手筋といった将棋特有の特徴を，特徴量やネットワーク構造を変更することで学習できるようにすることが必要である．また，さらにチェスにおける他の先行研究(DeepChess[15]等)を将棋に適用し実験を行うことで，ディープラーニングにおけるチェスと将棋の違いがより理解できると考える．

b) 芝浦将棋 Jr.の探索部分と自作した Stockfish 8 ベースの探索部分とは性能が異なるので，評価関数同士の直接的な比較にはなっていない．後者の探索性能の方がかなり上である．

参考文献

[1] Silver, D. et al.. Mastering the game of Go with deep neural networks and tree search. Nature 529, 2016, p. 484–489.

[2] Bernhardsson, E.. “Deep learning for... chess”, https://erikbern.com/2014/11/29/deep-learning-for-chess/, 2014.

[3] Lai, M.. Giraffe: Using Deep Reinforcement Learning to Play Chess. Master’s Thesis, Imperial College London, 2015.

[4] Baxter, J., Tridgell, A., and Weaver, L.. Tdleaf (lambda): Combining temporal difference learning with game-tree search. arXiv preprint cs/9901001, 1999.

[5] 松本慶太, 鈴木豪, 小谷善行. ニューラルネットワークを利用した TD（λ）法に基づく将棋の評価関数の学習. GPW2001 論文集, 2001, p. 176-178.

[6] 竹内章. コンピュータ将棋における大局観の実現を目指して. 人工知能学会誌, 2012, p. 443-448.

[7] Hoki, K., Kaneko, T.. Large-Scale Optimization for Evaluation Functions with Minmax Search. Journal of Artificial Intelligence Research, 49, 2014, p. 527-568.

[8] “Free Internet Chess Server”. http://www.freechess.org/, (参照 2017-10-09).

[9] Ahle, T. D.. “Sunfish.”. https://github.com/thomasahle/sunfish, (参照 2017-10-09).

[10] “Home - Stockfish - Open Source Chess Engine”. https://stockfishchess.org/, (参照 2017-10-09).

[11] Corbit, D., Swaminathan. “Strategic Test Suite”. https://sites.google.com/site/strategictestsuite/, (参照 2017-10-09).

[12] “CCRL (Computer Chess Rating Lists)”. http://www.computerchess.org.uk/ccrl/, (参照 2017-10-09).

[13] 出村洋介. “将棋ソフト「技巧」”. https://github.com/gikou-official/Gikou, (参照 2017-10-09).

[14] 和田悠介他. 「芝浦将棋 Jr.」とは. http://denou.jp/tournament2015/img/PR/ShibauraShogiJr.pdf, (参照 2017-10-09)

[15] David, O. E., Netanyahu, N., and Wolf, L.. DeepChess. End-to-End Deep Neural Network for Automatic Learning in Chess. ICANN 2016, 2016, p. 88-96.

