局面の組合せを用いた囲碁評価関数の学習

万代悠作[1,3,a)]　金子知適[2,4]

概要：本研究では局面の組合せを利用した囲碁評価関数の学習法について提案する. 深層学習において, 複数の入力を持つネットワークは近年盛んに研究されており, 様々な応用例が提案されている. そのような複数の入力を持つネットワークの学習では, 教師例の数を二乗のオーダーで増やすことが可能である. 本研究では既存の囲碁評価関数である局面価値関数と, ランキング学習の手法である RankNet を組合せることで局面の組合せによって学習を行うことができる学習手法を提案する.
実験では組合せを用いた学習手法と既存の局面価値関数を学習精度, 対局時の強さの点で評価したが, 既存手法を改善することはできなかった.

Learning evaluation functions of Go from combination of states

Yusaku Mandai[1,3,a)]　Tomoyuki Kaneko[2,4]

Abstract: This paper proposes a new learning method for evaluation functions of Go, which learns from combination of states. Recently a number of new neural network architectures are proposed and some of them take multiple feature vectors as input. Such neural networks are called 'Siamese network' and there are many applications of the networks. One of the advantages of the neural networks is that the number of training instances can be increased thanks to the combination of inputs. In this paper we proposed a new training method for evaluation functions of Go by combining the existing training method for state-value function and RankNet, which is framework of learning to rank.
We compare the performance of our proposal and the existing value function training method, however, the results showed that no significant improvements are archived in terms of accuracies, losses, and strength in matches.

1. はじめに

近年の人工知能の発展には, 探索手法の向上とならんで深層学習 (deep learning) 技術の進化が背景にある. 深層学習によって以前は困難だと考えられてきた囲碁の評価関数の学習が成功し, 人間と同等以上の性能を発揮した [1]. その評価関数の学習においては人間の熟達者が残した棋譜を用いて評価関数を学習し, 一定の強さをもつエージェントを作成したのちに強化学習によってさらに性能のよい評価関数を学習している.

一方で, 近年様々な深層学習の応用研究がなされており, そのうち複数の入力を持つ深層ニューラルネットワーク (DNN) の研究が近年注目されている. 特に入力の数が二つの DNN は Siamese ニューラルネットワークと呼ばれ, 古くから研究されており, 最初期の例では署名の検証の研究において提案されている [2]. 近年では画像の類似度の判定 [3] や, テキストの類似度判定 [4] などで利用されている. ゲームにおいて Siamese ネットワークを用いた先行研究として, チェスの評価関数を Siamese ネットワークを用いて作成した DeepChess [5] が存在する.

いずれの例でも, 入力が二つ存在するネットワークの訓練では, N 個の教師データを組合せを用いることで $O(N^2)$

[1] 東京大学大学院総合文化研究科
Graduate School of Arts and Sciences, The University of Tokyo
[2] 東京大学大学院情報学環
Interfaculty Initiative in Information Studies, The University of Tokyo
[3] 日本学術振興会特別研究員
JSPS Research Fellow
[4] 国立研究開発法人科学技術振興機構さきがけ
JST, PRESTO
[a)] mandai@graco.c.u-tokyo.ac.jp

に増やすことができ, 既存の知見が十分に蓄積されていない領域でも教師あり学習を効率的に行える可能性がある.

本研究では上記のような, 二つの入力を持つ DNN を利用し, 囲碁の評価関数を作成することを目的とする. 具体的には入力として二つの局面を受け取り, どちらがどれだけ優れているを判定する DNN の学習を目的とする. このような特徴を持つ DNN の先行研究として, ランキング学習 (learning to rank) で用いられている RankNet [6] が挙げられる. RankNet は入力を二つ受け取る Pairwise な学習手法であり, そのどちらが優れているかどうかを学習することが可能である.

実験では学習に用いる棋譜の数を変化させて DNN を訓練し, 正答率, 交差エントロピー損失, そして対戦成績という側面から性能を評価したが, 既存手法と比較して大きく改善することはできなかった.

2. 関連研究

2.1 AlphaGo

AlphaGo は Google DeepMind が開発した囲碁プログラムであり, モンテカルロ木探索と DNN による評価関数によって次の着手を決定している. AlphaGo は 2015 年 10 月に行われた, 当時欧州王者である Fan Hui 二段との対局において五勝〇敗の成績を残している. また改良されたと思われる AlphaGo は 2016 年に行われた, 当時最も強い棋士の一人である Lee Sedol 九段との対局で四勝一敗の成績を収めた.

AlphaGo の評価関数は 1) 既存の棋譜からの方策関数の学習 を行った後, 2) 自己対局による価値関数の学習 を行うことによって作成されている. 方策関数の学習では既存の棋譜から (局面, 着手) のペアを教師例として用いて次の行動の確率を学習する. 価値関数の学習では自己対局による棋譜から (局面, 終端局面の勝敗) のペアを教師例としてある局面の勝率を学習する. この学習は非常に大規模であり, 既存の棋譜は 3000 万局面, 自己対戦から抽出した局面も同程度の局面数を持っている.

2.2 DeepChess

ゲームの評価関数を局面の組合せを用いて行った先行研究として DeepChess [5] がある. DeepChess はコンピュータチェスプログラムであり, 本稿ではそれが用いている DNN を指す. DeepChess は入力として局面対を受け取り, そのどちらがより好ましい局面かを二項分類問題として出力する. 入力を二つ受け取るため, DeepChess は図 1 のような二叉のネットワークである. 図中の長方形は全結合層を表しており, 書かれている数字の数だけ入力を受け付ける.

DeepChess では重みの初期値を事前に自己符号化器 (autoencoder) で調整してから学習を行っている. 図 1 中の

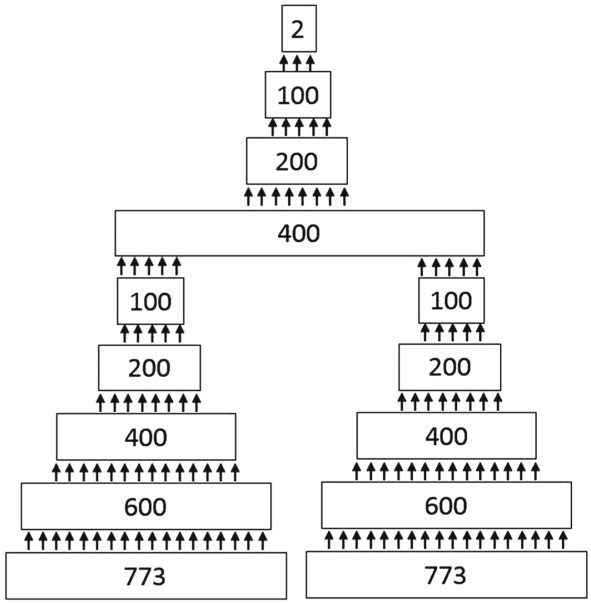

図 1 DeepChess のネットワーク構成
Fig. 1 Network architechture of DeepChess

$773 \to 600 \to 400 \to 200 \to 100$ の部分ネットワークを積層自己符号化器 (stacked autoencoder) として学習している.

自己符号化器によって重みの初期値を得た後, 学習を次のとおりに実行する. まず, 用いる棋譜に含まれる局面を白番から見た勝ち局面集合 W と 負け局面集合 L に分割する. その後 W, L から一局面ずつ w, l を乱択抽出し, 順番をランダムに入れ替えてネットワークに入力する. つまり確率 0.5 で (w, l), 0.5 で (l, w) の順番でそれぞれの下部ネットワークに入力する. ネットワークは勝ち局面がどちらであるかを正しく判定するように重みを更新する.

DeepChess の学習では $|W|, |L|$ ともに 百万程度であり, 手番の対称性なども考慮した場合, 組合せを考えた場合にはおよそ 2×10^{12} 通り程度作成することができる. 実験ではテスト集合における予測精度は 98% 程度であると報告されている. 対戦においては学習した DNN を使った alphabeta 探索により局面を評価している. α, β を値ではなく局面として保持し, 探索して訪れた新たな局面と, α, β それぞれとを学習したニューラルネットワークで比較する. 比較した結果が α より良ければ alpha-cut, β より悪ければ beta-cut が行われる.

2.3 RankNet

RankNet [6] はランキング学習 (learning to rank) の手法の一つである. ランキング学習は入力されたクエリに対してそれぞれのクエリの好ましさを出力するような学習器を構築することが目的である. ランキング学習の手法は入力する特徴ベクトルの数によって分類されることが多く, 単一の入力の場合 Pointwise, 二つの場合 Pairwise, そして

リストの場合 Listwise と呼ばれる.

RankNet はランキング学習に DNN を用いた手法で, Pairwise なランキング学習に分類される. RankNet には二つのクエリ U_i, U_j のそれぞれの特徴ベクトル x_i, x_j が入力される. 入力されるクエリ U_i, U_j において U_i が U_j より好ましいという事象を $U_i \triangleright U_j$ と表記する. RankNet は特徴ベクトルを実数値に写像する DNN f を用いて, それぞれのクエリのスコア $s_i = f(x_i)$, $s_j = f(x_j)$ を計算する. このとき, 事象 $U_i \triangleright U_j$ である確率 $P_{ij} = \Pr(U_i \triangleright U_j)$ をシグモイド関数とスコアを用いて

$$P_{ij} = \frac{1}{1 + e^{-\sigma(s_i - s_j)}}$$

であると仮定する. ここで σ はパラメータである. この確率を用いて, 交差エントロピー損失関数 C を以下のように表記できる:

$$C = -\bar{P_{ij}} \log P_{ij} - (1 - \bar{P_{ij}}) \log(1 - P_{ij}). \qquad (1)$$

ここで $\bar{P_{ij}}$ は U_i が実際に U_j よりも優れている確率である. 学習時には U_i, U_j に加えて $S_{ij} \in \{1, -1\}$ を教師例として入力する. このとき $U_i \triangleright U_j$ であるならば $S_{ij} = 1$, $U_j \triangleright U_i$ であるならば $S_{ij} = -1$ とする. そうすると, 既知の確率 $\bar{P_{ij}}$ は S_{ij} を用いて $\bar{P_{ij}} = \frac{1}{2}(1 + S_{ij})$ と計算できる. よって損失関数は

$$C = \frac{1}{2}(1 - S_{ij})\sigma(s_i - s_j) + \log(1 + e^{-\sigma(s_i - s_j)}) \qquad (2)$$

となる.

RankNet の学習とは DNN f の学習であり, 二つの入力 x_i, x_j から $s_i = f(x_i), s_j = f(x_j)$ を計算した後, s_i, s_j と S_{ij} を用いて損失 C を計算し, 誤差伝搬法によって f の重みを更新する.

3. 提案手法

局面の組合せを利用した RankNet を用いる囲碁評価関数の学習手法を提案する. つまり, 二つの局面を受け取って, どちらの局面が優れているかを出力するネットワークの学習法について提案する.

まずはじめに用意した棋譜集合を手番の勝利局面集合 W, 手番の敗北局面集合 L に分割し, それぞれの集合を訓練集合 $(W_{\text{train}}, L_{\text{train}})$ とテスト集合 $(W_{\text{test}}, L_{\text{test}})$ に分割する. 学習の際には, $W_{\text{train}}, L_{\text{train}}$ からランダム一つずつ局面 w_i, l_j を抽出し, 順番もランダムに入れ替えた局面対と教師例 (x_i, x_j, S_{ij}) を作成する. ここで x_i, x_j は局面の特徴ベクトルであり, どちらかが勝利局面 w_i を表す特徴ベクトルで, もう一方が敗北局面 l_j を表す特徴ベクトルである. ここで, x_i が w_i に対応するならば $S_{ij} = +1$, そうでないならば $S_{ij} = -1$ となる. その後, それぞれの入力のスコア s_i, s_j を DNN を用いて計算し, スコアから損失 C ((2) 式) を求め, 誤差伝搬法によって DNN の重みを更新する.

学習した DNN は局面の特徴ベクトルから実数値へと写像する関数 f となり, なおかつ二つの局面 s_i, s_j に関して s_i が s_j よりも優れているならば $f(s_i) > f(s_j)$ であると期待することができる. よってこの f そのものを評価関数として利用できると期待できる.

4. 実験

実験はすべて九路盤で行った. 実装には python 3.5 を, 深層学習のフレームワークとして chainer 2.1.0 を使用した.

実験は RankNet を用いた DNN の学習の性能評価, 学習した DNN の対戦における強さの測定, 及び終盤における予測性能について行った.

4.1 使用した棋譜

学習には山下宏氏によって作成されたコンピュータ囲碁プレイヤによって生成された棋譜を用いた. この囲碁プレイヤは CGOS の BayesElo で 2,900 程度の棋力を持つ. また対局において, 一方が投了した後には目数差が最大になるような方策に切り替えて着手を行い, 死石をすべて打ち上げ, 双方がパスを選択するまで行動選択を行っている. コミは 7.0 目であり, よって引き分けが起こりうる. 学習に用いる際には一つの対局からランダムに一局面のみを抽出したものを用いている.

4.2 ネットワークの学習

RankNet に用いる DNN は, 特徴ベクトルを実数値に写像するものであれば任意のものを使用できる. 本実験では [1] のネットワークを用い, 局面の特徴も同様のものを抽出して使用した. よって入力される特徴は $9 \times 9 \times 49$ のテンソルとなる.

実験では棋譜の多寡による性能の変化を調べるため, 学習に用いる棋譜の数を 10,000, 100,000, 及び 200,000 に制限して学習を行った.

4.2.1 事前学習

事前学習を用いずに DNN f の重みを正規乱数を使って初期化した場合, 学習が全く進まなかったため, DNN f を [1] と同様の手法で学習した. 学習に用いる棋譜の数に応じて三つの Value network を作成した. 以降, ValueNet-10k, ValueNet-100k, 及び ValueNet-200k と記述する.

学習では最適化手法に SGD を用い, 初期学習率を 0.003 とし, 1,000,000 イテレーションごとに 0.5 を学習率に乗じている. バッチサイズは 32 で, validation には三つのネットワークに共通の 10,000 局面を用いた. 用いた棋譜数ごとの学習曲線を図 2 に示す.

RankNet の初期値として用いる重みは, validation loss が最も小さかった epoch の重みを用いた. 最小の loss と

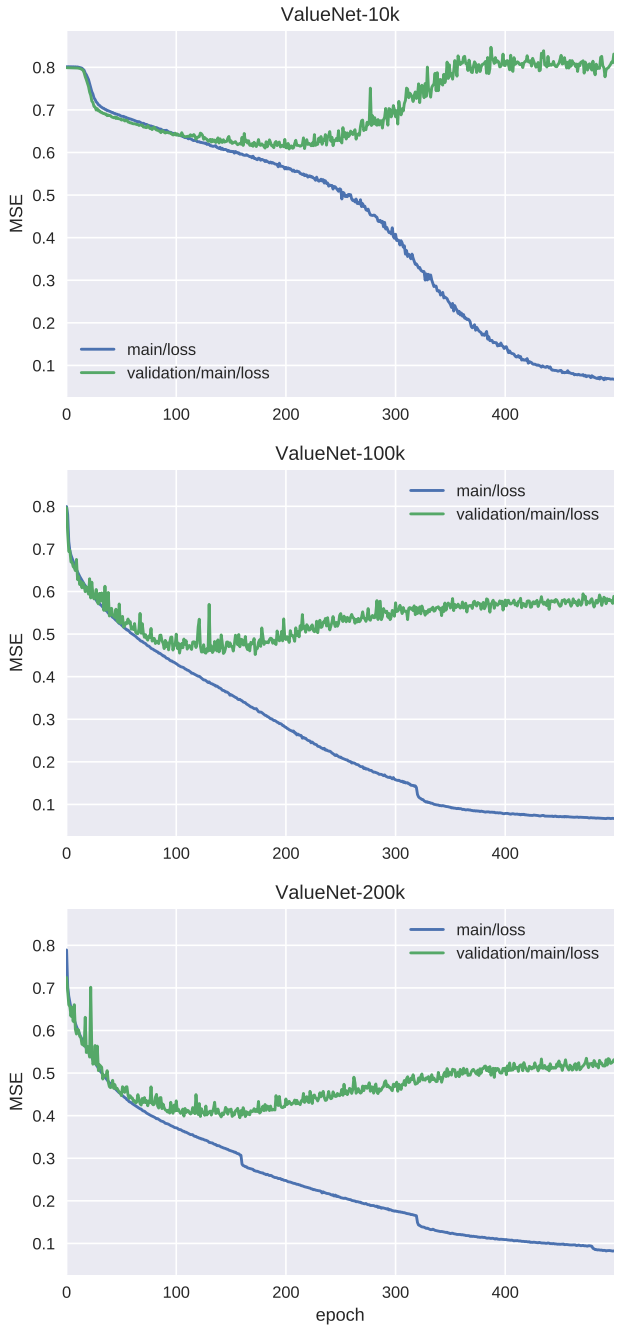

図 2 Value network の学習曲線. 縦軸: 平均二乗誤差, 横軸: epoch 数

Fig. 2 Learning curve of Value network. Vertical axes: Mean Squared Error, Horizontal axes: #epochs

表 1 それぞれの ValueNet の最良 epoch

Table 1 Best epochs of each ValueNet

	epoch	validation loss	loss
ValueNet-10k	203	0.61	0.56
ValueNet-100k	172	0.45	0.32
ValueNet-200k	158	0.40	0.31

そのときの epoch を次の表に示す.

なお, DeepChess では自己符号化器を利用してネットワークの重みの初期値を決定していたが, 今回用いた RankNet が要求するネットワークが特徴から実数値へ写像するものであるということ, 囲碁においては局面価値関数として DNN を用いる研究 [1], [7], [8] があることなどを考慮して自己符号化器ではなく, 価値関数を教師あり学習で学習し, それを重みの初期値とする方法を採用した.

表 2 学習に使用した局面数の内訳

Table 2 Breakdown of positions used in training, validation, and test

	black win	white win	black lose	white lose
train 10k	1894	2118	2219	1781
train 100k	18571	21441	21571	18403
train 200k	37047	42691	43298	36667
validation	1888	2194	2090	1821
test	500	500	500	500

4.2.2 RankNet の学習

提案する手法によって学習がうまく行えるかどうかを実験した. 初期の重みとして前節で述べた三つの Value network を用いた. 初期値として用いた Value network に対応してそれぞれ RankNet-10k, RankNet-100k, 及び RankNet-200k と記述する.

学習には SGD を用い, 初期学習率として 0.01 を与え, epoch ごとに 0.99 を学習率に乗じた. バッチサイズは 16 とし, ランダムに作成した 1,000,000 の局面対を 1 epoch とし, 100 epochs の学習を行った. すなわち 1 epoch あたり 62,500 イテレーション行う.

Validation には, 訓練集合とは別に用意した局面集合からランダムに作成した 10,000 局面対を用いて計測した. また正答率は Value network が出力した値が高い方の局面が勝利局面集合に含まれるならば正解としたときの正答率を計算している.

学習に用いた局面の内訳は表 2 のとおりである. 前述のとおり生成された棋譜は引き分けが存在するため, 勝利局面集合と敗北局面集合の要素数の和は対応する局面の数と一致しない.

図 3, 4 に学習経過における交差エントロピー損失並びに正答率の変化の様子を示す. Validation が大きく振動しているのは, 前述のとおり validation に用いた局面が epoch ごとに異なっていることが原因として考えられる. 図から, RankNet の学習は進行しているようにみえるが, いずれの訓練集合を用いた学習でも validation の性能が上がらず, 過学習をしていると考えられる.

図 5 は, 勝利, 敗北それぞれのテスト局面全 1,000 局面ずつのすべての組合せ (1,000,000 通り) で性能評価を行った結果である. 評価は表 1 の ValueNet と, 100 epoch 目の RankNet を対象に行なった. 図から分かるとおり, いずれの RankNet も学習を行う前の ValueNet の性能と比較して悪化した.

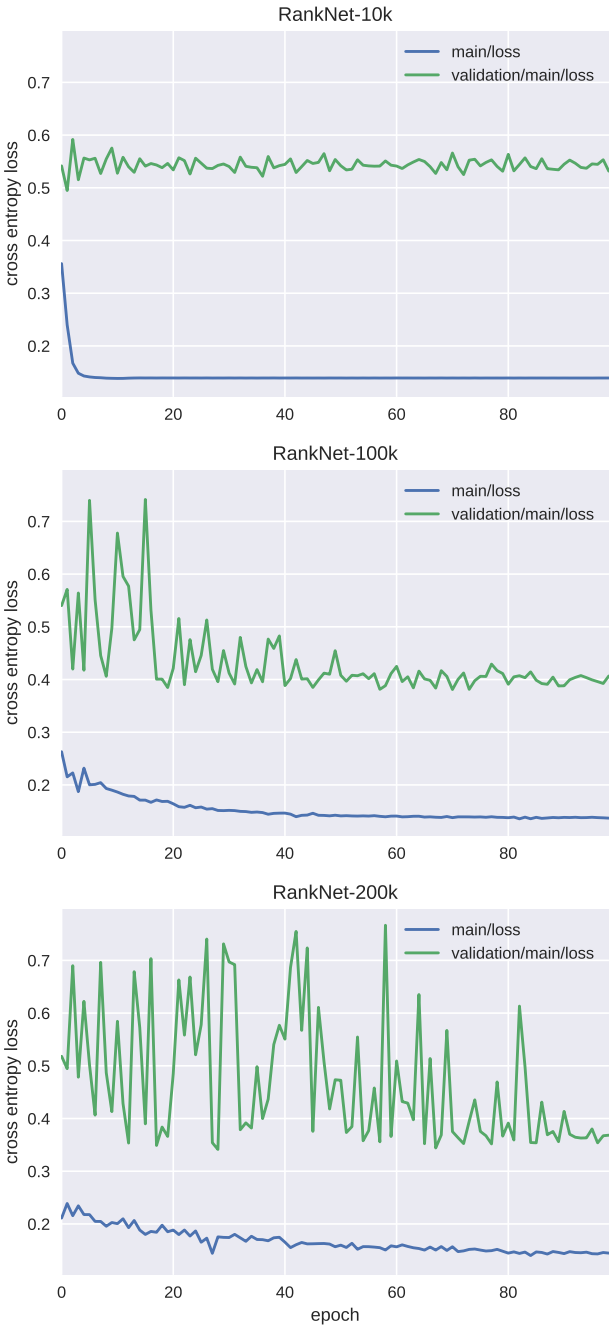

図 3 RankNet の学習曲線. 縦軸: 交差エントロピー損失, 横軸: epoch 数

Fig. 3 Learning curve of RankNet. Vertical axes: cross entropy loss, Horizontal axes: #epochs

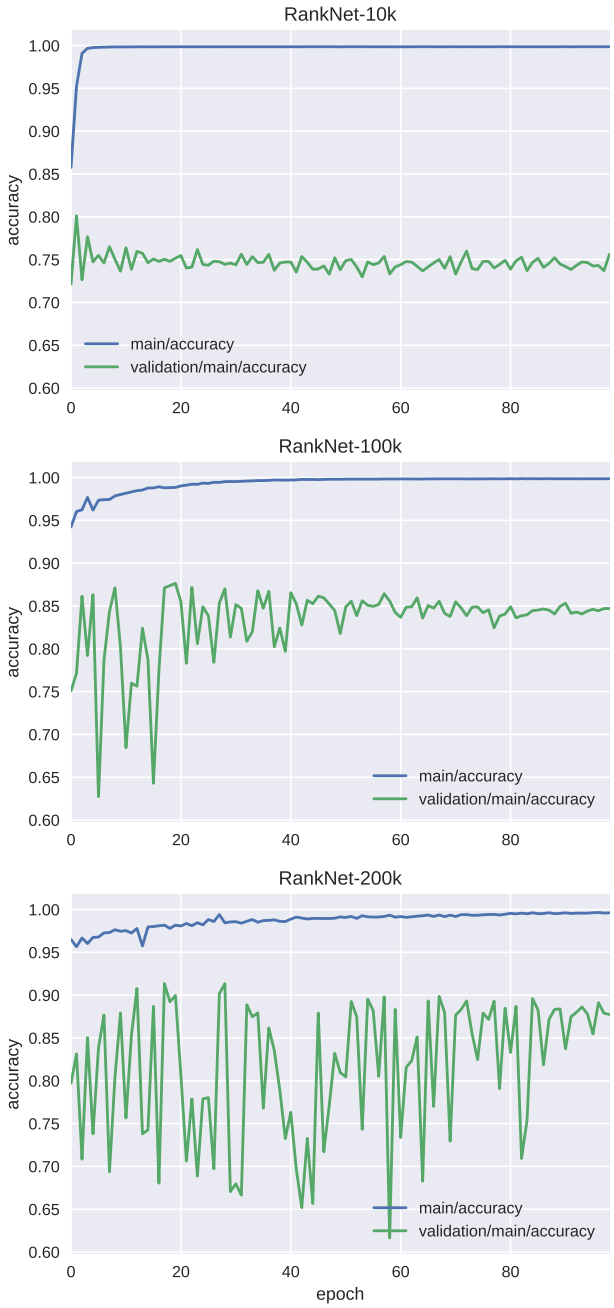

図 4 RankNet の学習曲線. 縦軸: 正答率, 横軸: epoch 数

Fig. 4 Learning curve of RankNet. Vertical axes: accuracy, Horizontal axes: #epochs

4.3 対戦実験

学習した DNN を用いて着手を決定するプレイヤーと, gnugo 3.8 とを対戦させ, 性能を測定した. gnugo のレベルはデフォルト値 (10) を用いた. コミは教師例と合わせるために 7.0 に設定した.

DNN プレイヤーの局面 s における着手決定は, s のすべての後続局面のうち, 最大の値を出力する局面に導く行動を決定的に選択する. このとき局面の対称性 (鏡像, 回転の組合せ計 8 通り) も考慮に入れている.

対戦に用いた DNN の重みは, ValueNet の場合表 1 の重みを, RankNet の場合は 100 epoch 目の重みを用いた. 対戦した結果を表 3 に示す. いずれの RankNet も, 同数の棋譜を使用した ValueNet と比較して成績が悪化する結果となった.

4.4 終盤の勝敗予測

最後に, 学習対象を終盤のみに限定した場合の勝敗予測について実験した. 重みの初期値として ValueNet-200k を用い, 表 4 の局面を用いて学習を行った. ここで, ある対局を三等分した場合の最後の局面集合を終盤としている. 学習に用いた局面の内訳を表 4 に示す.

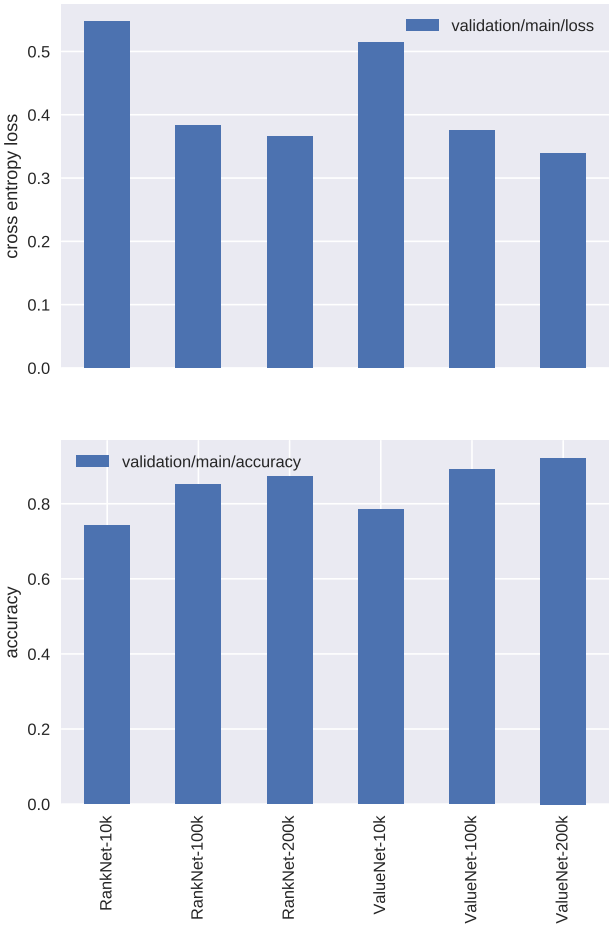

図 5 学習器ごとの test 局面での性能評価
Fig. 5 Test evaluation of each trained neural network

表 3 GnuGo 3.8 との対戦成績. 括弧内は標準誤差
Table 3 Match results against GnuGo 3.8. Figures inside parentheses indicate standard errors

	Win ratio (%)	Score
RankNet-10k	0 (0.0)	-76.6 (1.1)
RankNet-100k	0 (0.0)	-73.5 (3.1)
RankNet-200k	16.7 (8.8)	-58.4 (7.0)
ValueNet-10k	0 (0.0)	-74.3 (2.2)
ValueNet-100k	25 (7.2)	-46.2(5.7)
ValueNet-200k	60 (8.9)	-16 (6.2)

表 4 学習に使用した局面の内訳 (終盤のみ)
Table 4 Breakdown of positions used in training and test (Endgames only)

	black win	white win	black lose	white lose
train	12315	14160	13967	11669
test	637	690	659	571

学習は 4.2.2 章で述べた手法と同様に行った. 10 epochs の学習を行い, その重みを RankNet-200k_endgame とする. その後 test 局面のすべての組合せである 1,632,210 局面対に対して交差エントロピー損失と正答率を計算し, 性能を評価した. 結果を表 5 に示す.

表 5 終盤局面における評価
Table 5 Perforamnce of networks in endgames

	test loss	test accuracy
RankNet-200k_endgame	0.166	0.993
ValueNet-200k	0.175	0.995

交差エントロピー損失に関しては RankNet が ValueNet よりもよい性能を示した. 正答率については ValueNet の成績が有意に高くなった.

5. 考察

全体を通して, RankNet を用いた学習手法は成果を上げることができなかった. 終盤のみを学習対象にした実験では, 損失関数の面では学習によって性能が向上したが, 大きく改善することはなかった.

原因として考えられるのは囲碁の学習の困難さが挙げられる. AlphaGo の DNN 作成には非常に大規模な棋譜集合を用いており, 今回用いた棋譜の数では不足していた可能性がある.

その他, RankNet に入力する曲面の入力の仕方を工夫することも考えられる. 今回はどちらかが必ず勝利曲面でもう一方が必ず敗北曲面であったが, 両方とも勝利局面もしくは敗北局面を入力するということも考えることができる.

6. おわりに

本研究では囲碁において二つの局面の優劣を判定する DNN の構築を目的とし, 構築にあたって RankNet を用いたランキング学習の枠組みを用いた. 実験結果からは既存の局面価値関数以上の性能を出すことは叶わなかった. また終盤のみを対象とした学習ではわずかに既存手法よりも優れた結果を残すことができた.

今後の研究として, 囲碁において序盤のみ, 中盤のみといったより容易な学習についての知見を得ることが考えられる. また近年提案された curriculumn learning の知見を活かし, 学習する局面そのものを学習することにより評価関数の学習を促進するといった研究が考えられる.

謝辞

本研究にあたり, 棋譜生成のためのプログラムを使用をご快諾くださった山下宏氏に深く感謝いたします. この研究の一部は，JSPS 科研費 17J09685, 16H02927 と JST さきがけの支援を受けています.

参考文献

[1] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T. P., Leach, M., Kavukcuoglu, K., Graepel, T. and Has-

sabis, D.: Mastering the game of Go with deep neural networks and tree search, *Nature*, Vol. 529, No. 7587, pp. 484–489 (online), DOI: 10.1038/nature16961 (2016).

[2] Bromley, J., Bentz, J. W., Bottou, L., Guyon, I., LeCun, Y., Moore, C., Säckinger, E. and Shah, R.: Signature Verification Using A "Siamese" Time Delay Neural Network, *IJPRAI*, Vol. 7, No. 4, pp. 669–688 (online), DOI: 10.1142/S0218001493000339 (1993).

[3] Chopra, S., Hadsell, R. and LeCun, Y.: Learning a Similarity Metric Discriminatively, with Application to Face Verification, *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2005), 20-26 June 2005, San Diego, CA, USA*, IEEE Computer Society, pp. 539–546 (online), DOI: 10.1109/CVPR.2005.202 (2005).

[4] Hu, B., Lu, Z., Li, H. and Chen, Q.: Convolutional Neural Network Architectures for Matching Natural Language Sentences, *CoRR*, Vol. abs/1503.03244 (online), available from ⟨http://arxiv.org/abs/1503.03244⟩ (2015).

[5] David, O. E., Netanyahu, N. S. and Wolf, L.: DeepChess: End-to-End Deep Neural Network for Automatic Learning in Chess, *Artificial Neural Networks and Machine Learning - ICANN 2016 - 25th International Conference on Artificial Neural Networks, Barcelona, Spain, September 6-9, 2016, Proceedings, Part II* (Villa, A. E. P., Masulli, P. and Rivero, A. J. P., eds.), Lecture Notes in Computer Science, Vol. 9887, Springer, pp. 88–96 (online), DOI: 10.1007/978-3-319-44781-0 (2016).

[6] Burges, C. J. C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N. and Hullender, G. N.: Learning to rank using gradient descent, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005* (Raedt, L. D. and Wrobel, S., eds.), ACM International Conference Proceeding Series, Vol. 119, ACM, pp. 89–96 (2005).

[7] Tian, Y. and Zhu, Y.: Better Computer Go Player with Neural Network and Long-term Prediction, *CoRR*, Vol. abs/1511.06410 (online), available from ⟨http://arxiv.org/abs/1511.06410⟩ (2015).

[8] Wu, T., Wu, I., Chen, G., Wei, T., Lai, T., Wu, H. and Lan, L.: Multi-Labelled Value Networks for Computer Go, *CoRR*, Vol. abs/1705.10701 (online), available from ⟨http://arxiv.org/abs/1705.10701⟩ (2017).