Vol.2022-GI-47 No.13
2022/3/18

MC Softmax 探索における局面評価関数の強化学習 —5 五将棋への適用—

岩本裕大[1]　粂川叶[1]　五十嵐治一[1]

概要：MC Softmax 探索は，コンピュータ将棋などのゲーム AI に用いられる探索手法の一手法である．この探索法のための局面評価関数の学習法が 2018 年に五十嵐らにより提案されていたが，実際にコンピュータゲームへ適用しての検証は出来ていなかった．この学習法は探索木内の全ての葉局面や内部ノードを学習対象とすることが可能である．かつ，バックアップ方策による確率的なサンプリングを用いることで，学習パラメータに関するバックアップ評価値の勾配ベクトルの計算を効率化できる．この勾配ベクトルは，回帰，TD(λ)法，方策勾配法，ブートストラップ法（Q 学習）などの計算に共通であり，これらの複数の学習法を同時に行う効率的な複合学習が可能である．本研究では，５五将棋を題材としてこの学習法の検証を実験により行った．結果として，各学習法のサンプリングを用いた学習の正当性と，複数の学習法を同時に適用することによる学習効果の有効性を確認することができた．

キーワード：MC Softmax 探索，方策勾配法，バックアップ方策，将棋

Reinforcement Learning of Evaluation Functions in Monte Carlo Softmax Search - Application to Minishogi -

HIROMASA IWAMOTO[†1]　KANAU KUMEKAWA[†1]
HARUKAZU IGARASHI[†1]

Abstract: MC Softmax search is one of the search methods used in game AI such as computer Shogi. A learning method of state evaluation functions for this search method was proposed by Igarashi et al. in 2018, but it has not been verified by actually applying it to computer games. This learning method can be applied to all leaf nodes and internal nodes in a search tree. In addition, by using probabilistic sampling with backup policies, we can simplify the computation of the gradient vector of backed-up evaluation values with respect to learning parameters. This gradient vector is common to the computation of regression, TD(λ) method, policy gradient method, bootstrapping method (Q-learning), etc., which enables efficient combined learning where these multiple learning methods are performed simultaneously. In this study, we experimentally validated this learning method on the subject of 5x5 Shogi. As a result, we confirmed the validity of learning with sampling of each learning method and the effectiveness of learning by applying multiple learning methods simultaneously.

Keywords: Monte Carlo Softmax Search, Policy gradient reinforcement learning, Back-up policy, Shogi

1. はじめに

一般にチェス，囲碁，将棋などの知的ゲームおいて最善な着手を選択するアルゴリズムには，ゲーム木の生成と末端局面の評価値計算を用いる[1]．すなわち，ゲーム木の展開と子ノードの選択，末端評価値のバックアップ方法などの「探索」と，末端局面の優劣を数量化する「局面評価」の２つの処理から構成されている．

探索については，これまでは伝統的にミニマックス探索の高速化版であるアルファ･ベータ探索が用いられてきた．これらは探索木をすべて展開し，末端局面（以下，葉局面または leaf ノード）の局面評価値をミニマックス演算でバックアップさせる「全幅探索」の考えに基づいていた．しかし，近年では囲碁で考案されたモンテカルロ木探索（Monte Carlo Tree Serach, MCTS）[2] [3]が大成功をおさめている．特に，DeepMind Technologies 社が 2017 年に開発したソフト「Alpha Zero」では，チェスや将棋のアルファベータ探索に基づくチャンピオンソフトに圧勝している[4]．この MCTS は全幅探索ではなく，探索木の探索中に選択あるいは展開すべき子ノードをある種の方策により選択していく「選択方策」と呼ばれる手法の一種である．

我々はこれまでにコンピュータ将棋を題材にして，「モンテカルロ・ソフトマックス探索法」（Monte Carlo Softmax Search, MCSS）と呼ばれる探索法を提案してきた[5] [6]．これは選択探索の一種であり，子ノードの評価値に基づいた確率的な選択方策を特徴としている．MCSS はバックアップの方策も通常のミニマックス演算ではなく，確率的な期待値操作により子ノードの評価値を親ノードの評価値にバックアップさせている．

この操作により，探索木の全ての葉局面の評価結果が着

1 芝浦工業大学
Shibaura Institute of Technology

手決定に貢献する度合いを定量的に計算することができる．したがって，葉局面の評価関数に含まれるパラメータによる指し手の価値（あるいはノードの価値）の勾配ベクトルを厳密に計算することが可能である．しかし，全葉局面に関して勾配ベクトルをすべて計算すると計算量が爆発してしまう．そこで，バックアップの確率に比例したモンテカルロ・サンプリング（MC サンプリング）をルートノードから探索木に沿って末端へ向けて行うことにより，貢献度の高い葉局面を優先的に選んで計算するという効率的な近似法を提案した[6].

上記の勾配ベクトルは局面評価関数の学習に非常に重要な役割を果たしうる．実際，教師あり学習のほか，代表的な強化学習の手法である TD 法，方策勾配法，回帰法，ブートストラッピング法について，この勾配ベクトルと MCSS に基づいた学習法が提案されてきた[6]．この勾配ベクトルを MC サンプリングの手法で計算すれば，内部ノードを含む全ノードの局面を学習データとして局面評価関数の学習に効率良く役立てることが可能である．

本論文では，モンテカルロ・ソフトマックス探索のために考案された上記学習方式を 5 五将棋に適用し，学習則の正当性，MC サンプリングの有効性，複数の強化学習法へ同時適用した際の有効性を調べた結果を報告する．

2. MC Softmax 探索について

2.1 探索の概要

MC Softmax 探索の流れを以下に示す[5] [6].

i) 初期設定：現在の出現局面u_tをノードuに設定
ii) ノード選択：ノードuから選択確率に従って子ノード$v = child(u)$を選択
iii) ノード展開：vが展開済みであればuをvに置き換えてii)へ，未展開であれば一段階だけ子ノードを全て展開
iv) ノード評価：展開した子ノードをそれぞれの局面の評価値を局面評価関数$H_a(s)$により計算
v) バックアップ：ルートノードからvへの経路を逆にたどり，vを含む経路上のノード評価値を更新

i)から v)の処理を繰り返し行うことで，探索木を成長させる．また，本論文では，対局中の自手番の出現局面をu_t，指し手をa_t，相手番の出現局面をv_t，指し手をb_tで表す（図 1 参照）．探索木中のノードや指し手についても図 1 の表記を用いる．tは対局中の時刻ステップ，dは探索木内のノードの深さを表している．

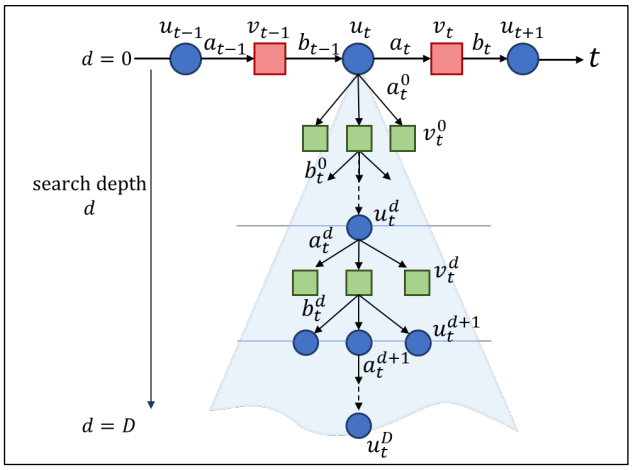

図 1　出現局面と探索木

2.2 ノード選択方策

ii)ノード選択では，ノード選択方策$\pi_a(a|u)$を用いて次に訪問するノードを決定する[5]．自手番の局面u_t^dから指し手a_t^dを選択する確率$\pi_a(a_t^d|u_t^d)$は以下のように Boltzmann 分布で表される．

$$\pi_a(a_t^d|u_t^d) = \exp(Q_a(u_t^d, a_t^d)/T_{\pi a})/Z_{\pi a} \tag{2.1}$$

$$Z_{\pi a} \equiv \sum_{x \in A(u_t)} \exp(Q_a(u_t, x)/T_{\pi a}) \tag{2.2}$$

ここで，$T_{\pi a}$はノード選択温度，$Q_a(u_t^d, a_t^d)$は局面u_t^dにおける指し手a_t^dの評価値，$A(u_t^d)$は局面u_t^dにおける子ノードの集合を表す．相手番の局面v_t^dから指し手b_t^dを選択する確率$\pi_b(v_t^d|b_t^d)$についても同様に以下の式で表される．

$$\pi_b(b_t^d|v_t^d) = \exp(Q_b(v_t^d, b_t^d)/T_{\pi b})/Z_{\pi b} \tag{2.3}$$

$$Z_{\pi b} \equiv \sum_{x \in A(v_t)} \exp(Q_b(v_t, x)/T_{\pi b}) \tag{2.4}$$

ここで，$T_{\pi b}$はノード選択温度，$Q_b(v_t^d, b_t^d)$は局面v_t^dにおける指し手b_t^dの評価値を表す．

2.3 バックアップ方策と評価値の再帰的定義

自手番におけるバックアップ方策$P_a(u_t^d, a_t^d)$は以下のように Boltzmann 分布で表される[5].

$$P_a(a_t^d; u_t^d) = \exp(Q_a(u_t^d, a_t^d)/T_{Pa})/Z_{Pa} \tag{2.5}$$

$$Z_{Pa} \equiv \sum_{x \in A(u_t^d)} \exp(Q_a(u_t^d, x)/T_{Pa}) \tag{2.6}$$

ここで，T_{Pa}はバックアップ温度を表す．相手番のバックアップ方策$P_b(b_t^d; v_t^d)$についても以下のように Boltzmann 分布で表される．

$$P_b(b_t^d; v_t^d) = \exp(Q_b(v_t^d, b_t^d)/T_{Pb})/Z_{Pb} \tag{2.7}$$

$$Z_{Pb} \equiv \sum_{x \in A(v_t^d)} \exp(Q_b(v_t^d, x)/T_{Pb}) \tag{2.8}$$

ここで，T_{Pb}はバックアップ温度を表す．

次に，探索木中のノード評価値を指し手評価値とバックアップ方策を用いて以下のように定義する．$V_a(u_t^d)$は自手番局面u_t^dのノード評価値，$V_b(v_t^d)$は相手番局面v_t^dである．

$$V_a(u_t^d) = \sum_{a_t^d} P_a(a_t^d; u_t^d) Q_a(u_t^d, a_t^d) \tag{2.9}$$

$$V_b(u_t^d) = \sum_{b_t^d} P_b(b_t^d; v_t^d) Q_b(v_t^d, b_t^d) \tag{2.10}$$

将棋の場合，指した後の局面は一意に定まるので，指し手評価値は指した後の局面のノード評価値に置き換えることが出来る．すなわち，

$$Q_a(u_t^d, a_t^d) = V_b(v_t^d) \tag{2.11}$$

$$Q_b(v_t^d, b_t^d) = V_a(u_t^{d+1}) \tag{2.12}$$

が成り立つ．(2.9)~(2.12)を用いると，ノード評価値は以下のように再帰的に書くことができる．

$$V_a(u_t^d) = \sum_{a_t^d} P_a(a_t^d; u_t^d) \sum_{b_t^d} P_b(b_t^d; v_t^d) V_a(u_t^{d+1}) \tag{2.13}$$

$$V_b(v_t^d) = \sum_{b_t^d} P_b(b_t^d; v_t^d) \sum_{a_t^{d+1}} P_a(a_t^{d+1}; u_t^{d+1}) V_b(v_t^{d+1}) \tag{2.14}$$

ただし，末端ノード$(d = D)$では局面評価関数$H_a(u;\omega)$を用いて次のように定義する．ωは局面評価関数で使用されている重みなどのパラメータを表す．

$$V_a(u_t^D) \equiv H_a(u_t^D; \omega) \tag{2.15}$$

指し手評価値についても，(2.9)~(2.12)を用いて以下のように再帰的に書くことができる．

$$Q_a(u_t^d, a_t^d) = \sum_{b_t^d} P_b(b_t^d; v_t^d) \sum_{a_t^{d+1}} P_a(a_t^{d+1}; u_t^{d+1}) Q_a(u_t^{d+1}, a_t^{d+1}) \tag{2.16}$$

3. MC Softmax 探索における強化学習

3.1 学習則について

本論文では，文献[6]で述べた学習則の内，TD(λ)法[7]，PG 期待値法[8][9]，回帰法，Bootstrapping 法[10] (Q 学習)の 4 つの学習則について取り上げる．

TD(λ)法[7]は，代表的な価値ベースの強化学習法であり，状態価値関数をノード評価値$V_a(u_t)$で置き換えた以下の学習則が提案されていた．

$$\Delta\omega_{TD} = \varepsilon \sum_{t=1}^{L_a-1} e^{\lambda}(t)\delta_t \tag{3.1}$$

$$e^{\lambda}(t) = \gamma\lambda e^{\lambda}(t-1) + \nabla_\omega V_a(u_t) \tag{3.2}$$

$$\delta_t \equiv r_{t+1} + \gamma V_a(u_{t+1}) - V_a(u_t) \tag{3.3}$$

ここで，L_aは一局の手番数，εは学習率，γとλは範囲[0,1]の定数，r_tは報酬である．本論文の学習実験では，$\varepsilon = 1, \gamma = 0.98, \lambda = 0.95, r_t =$ (勝利時:1,敗北時:-1)などの値を用いた．$\nabla_\omega V_a(u_t)$の計算法について，3.4 の MC サンプリングを用いる手法を TD(λ)-Sampling 法とする．また，$\nabla_\omega V_a(u_t)$に PV 末端の局面評価関数を用いると TD(λ)-Leaf 法となる．

PG 期待値法[8]は，Williams の REINFORCE [11]を将棋に適用した Softmax ベースの学習法であり，PV 末端局面だけを用いた近似手法として PGLeaf 法[12]がある．PG 期待値法の学習則は，バックアップ方策と指し手評価値を用いて以下のように表される．

$$\Delta\omega_{PGE} = \varepsilon \cdot r \sum_{t=1}^{L_a-1} e_\omega(t) \tag{3.4}$$

$$e_\omega(t) \equiv \nabla_\omega \ln P_a(a_t; u_t) \tag{3.5}$$

$$= \left[\nabla_\omega Q_a(u_t, a_t) - \sum_{a \in A(u_t)} P_a(a; u_t) \nabla_\omega Q_a(u_t, a)\right] \Big/ T_a \tag{3.6}$$

ここで，rは終局時に与える報酬である．本論文の学習実験では，$\varepsilon = 10, r =$ (勝利時:1,敗北時:-1)などの値を用いた．$\nabla_\omega Q_a(u_t, a)$の計算法について，3.4 の MC サンプリングを用いる手法を PGSampling 法とする．

回帰法では，対局結果を用いて学習を行う．勝敗を$r(\sigma) =$ 1(勝ち) or 0(負け)と数値化し，ノード評価値$V_a(u)$を用いて勝利する確率を与える勝利予測関数$P_{win}(u)$を学習する．

$$\Delta\omega_{reg} = \varepsilon \sum_{t=1}^{L_a} [r(\sigma) - P_{win}(u_t)] \nabla_\omega P_{win}(u_t) \tag{3.7}$$

$$P_{win}(u_t) = 1/[1 + \exp(-V_a(u_t)/\tau)] \tag{3.8}$$

$$\nabla_\omega P_{win}(u_t) = (1/\tau)[1 - P_{win}(u_t)] P_{win}(u_t) \nabla_\omega V_a(u_t) \tag{3.9}$$

ここで，τは定数である．本論文の学習実験では，$\varepsilon = 1000, \tau = 600$などの値を用いた．Alpha Zero[4]も回帰による学習を行っているが，$P_{win}(u)$の中では，探索によりバックアップされたノード評価値$V_a(u)$ではなく，局面評価関数$H_a(u)$を用いている．

Bootstrapping 法は，浅い局面の評価を深い探索結果に近づける学習法である．Q 学習は，行動価値関数を次の行動価値関数に近づける学習法であるが，探索木の内部ノードに適用することで Bootstrapping 法と一致する[6]．以後，Boostrapping 法と Q 学習は同一視してまとめて扱うこととする．学習則は以下を用いる．

$$\Delta\omega_{bts} = -\varepsilon \sum\nolimits_{s \in S} [H_a(s) - V_a(s)] \nabla_\omega H_a(s) \tag{3.10}$$

本論文の学習実験では，$\varepsilon = 0.7$などの値を用いた．また，局面$s \in S$は出現局面(root)や PV 上のノードだけでなく，内部ノードも許すことにより，学習データ数を大幅に増やすことが可能である．

3.2 勾配ベクトルについて

前節で述べた学習法の内，TD(λ)法，回帰法ではノード評価値の勾配ベクトル$\nabla_\omega V_a(u_t)$, PG 期待値法では指し手評価値の勾配ベクトル$\nabla_\omega Q_a(u_t)$が計算に必要である．これらの勾配ベクトルは以下のような再帰的な式で表される[5][6].

$$\nabla_\omega V_a(u_t^d) = \sum_{a_t^d} P_a(a_t^d; u_t^d) \cdot [\{Q_a(u_t^d, a_t^d) - V_a(u_t^d)\}/T_{Pa} + 1]\nabla_\omega Q_a(u_t^d, a_t^d) \quad (3.11)$$

$$\nabla_\omega Q_a(u_t^d, a_t^d) = \sum_{b_t^d} P_b(b_t^d; v_t^d)\nabla_\omega V_a(u_t^{d+1}) \quad (3.12)$$

ただし，探索木の末端ノードでは,

$$\nabla_\omega V_a(u_t^D) = \nabla_\omega H_a(u_t^D; \omega) \quad (3.13)$$

である．

3.3 勾配ベクトルの再帰式の相手番を含める拡張

3.2 の勾配ベクトルの再帰式では, 相手番のバックアップ方策$P_b(b_t^d; v_t^d)$はパラメータωに対して固定であり，末端ノードは全て自身の手番局面であるとしていた[5][6]．しかし，実装において末端ノードの手番を揃えようとすると，ノード展開の度に全ての孫ノードを展開する必要があり，学習時と対局時で生成される探索木が乖離してしまうなどの問題があった．そこで本研究では，勾配ベクトルの計算について相手番の局面も学習に取り入れる拡張を行う. つまり，相手番のバックアップ方策をωで微分し，末端ノードが相手局面となることを許す.

本研究の学習実験には三駒関係の評価関数(KPPT)を使用するため，評価値は自身と相手で正負が入れ替わる．そのため，指し手評価値は，指した先の相手番の局面評価値の正負を反転して,

$$Q_a(u_t^d, a_t^d) = -V_b(v_t^d) \quad (3.13)$$

$$Q_b(v_t^d, b_t^d) = -V_a(u_t^{d+1}) \quad (3.14)$$

となる．また，探索木の末端ノード$(d = D)$が相手番の局面であった場合は,

$$V_b(v_t^D) = -H_a(v_t^D; \omega) \quad (3.15)$$

と表される．(3.13)~(3.15)を用い，バックアップ方策$P_b(b_t^d; v_t^d)$もパラメータωの関数として指し手/ノード評価値の勾配ベクトルを計算すると，それぞれ以下の再帰的な関係式によって表される.

$$\nabla_\omega Q_a(u_t^d, a_t^d) = -\nabla_\omega V_b(v_t^d) \quad (3.16)$$

$$\nabla_\omega V_a(u_t^d) = -\sum_{a_t^d} P_a(a_t^d; u_t^d) \cdot [\{-V_b(v_t^d) - V_a(u_t^d)\}/T_{Pa} + 1]\nabla_\omega V_b(v_t^d) \quad (3.17)$$

$$\nabla_\omega V_b(v_t^d) = -\sum_{b_t^d} P_b(b_t^d; v_t^d) \cdot [\{-V_a(u_t^d) - V_b(v_t^d)\}/T_{Pb} + 1]\nabla_\omega V_a(u_t^d) \quad (3.18)$$

ただし，探索木の末端ノードでは，それぞれの手番が末端であった場合において,

$$\nabla_\omega V_a(u_t^D) = \nabla_\omega H_a(u_t^D; \omega) \quad (3.19)$$

$$\nabla_\omega V_b(v_t^D) = -\nabla_\omega H_a(v_t^d; \omega) \quad (3.20)$$

である．これ以降，本論文では，この再帰式を前提に計算や学習実験を行うこととする.

なお, 本研究では三駒関係や NNUE などで用いられる CP による評価値に基づいた式変形を行っているが, DNN などで用いられる予測勝率(範囲[0,1])による評価関数についても，(3.13),(3.14)をはじめとして式の一部を置き換えれば容易に導出が可能である.

3.4 勾配ベクトルの MC サンプリングを用いた近似計算

前節で示した勾配ベクトルの計算式(3.11)～(3.13)は，探索木全体を参照するため，厳密な計算を行うと計算コストが高い．そこで，文献[6]では MC サンプリングによる近似計算を行うことが提案された．本節では，より詳細に近似計算を行う具体的な手順について述べる.

MC サンプリングによる勾配ベクトルの近似計算は,

i) バックアップ方策の確率分布に基づくノード選択
ii) 訪問したノードの情報を利用した勾配ベクトル計算

の 2 つの操作をサンプリング一回とし，これを十分な回数繰り返す．サンプリングの回数は，本研究では探索木のノード数 1000 万程度(探索時間 0.5 秒)に対して 1 万回のサンプリング(所要時間約 0.2 秒)を行っている. このように勾配計算にかかる時間は探索木の生成時間よりも少なく，複数の学習を同時に行う場合も共通の勾配ベクトルを用いるため追加の計算上の負担にはならない.

勾配ベクトル$\nabla_\omega V_a(u)$を求めたいノードをuとすると，i)のノード選択では，バックアップ方策$P_a(a;u)$，$P_b(b;v)$の確率分布に基づく子ノード選択をノードuから末端ノードに到達するまで繰り返す．ここで，サンプリング回数をN_sとし，サンプリングi回目に深さdのとき訪問したノードをx_i^dとする. 末端の深さは$d = D_i$とし，$d = 0$では$x_i^0 = u$である．サンプリングで訪問したノードの集合は$\{x_i^d\}$とする.

次に i)で訪問したノードの経路について，ii)の勾配ベクトル計算を行う．ノード評価値の近似勾配ベクトル$\nabla_\omega V_a'(u)$は以下のような再帰式で求められる.

$$\nabla_\omega V'_a(u) = \frac{1}{N_s}\sum_{i=1}^{N_s} X_{Va}(x_i^0) \tag{3.21}$$

$$X_{Va}(x_i^d) = -\left[\{-V_b(x_i^{d+1}) - V_a(x_i^d)\}/T_{Pa} + 1\right]X_{Vb}(x_i^{d+1}) \tag{3.22}$$

$$X_{Vb}(x_i^d) = -\left[\{-V_a(x_i^{d+1}) - V_b(x_i^d)\}/T_{Pb} + 1\right]X_{Va}(x_i^{d+1}) \tag{3.23}$$

ただし，探索木の末端ノード$(d = D_i)$では，

$$X_{Va}(x_i^{D_i}) = \nabla_\omega H_a(x_i^{D_i};\omega) \tag{3.24}$$

$$X_{Vb}(x_i^{D_i}) = -\nabla_\omega H_a(x_i^{D_i};\omega) \tag{3.25}$$

となる．なお実際の計算においては，ノード評価値$V_a(x), V_b(x)$はシグモイド関数を用いて$[0,1]$の範囲に写像して計算している．

指し手評価値の近似勾配ベクトル$\nabla_\omega Q'_a(u,a)$についてもサンプリングで訪問したノードの集合を用いて計算するが，この集合は指し手aを選択しなかった経路も含むため，それを除外して勾配ベクトルの計算を行う．

$$\nabla_\omega Q'_a(u,a) = \frac{1}{N_{Qa}}\sum_{i=1}^{N_s} X_{Qa}(x_i^1) \tag{3.26}$$

$$X_{Qa}(x_i^1) = -\mathbb{I}_{\{x_i^1 = v(a;u)\}}X_{Vb}(x_i^1) \tag{3.27}$$

$$N_{Qa} = \sum_{i=1}^{N_s}\mathbb{I}_{\{x_i^1 = v(a;u)\}} \tag{3.28}$$

ここで，$\mathbb{I}_B$は真偽値を返す式Bの値が真のときに 1，偽のときに 0 を返す指示関数である．つまり，ノードx_i^1が局面uでaを指した後の局面と一致する場合のみ勾配ベクトルの計算を行っている．

また，PG 期待値法(3.6)の[　]内の第二項については，指し手それぞれについて計算を行うことなく，以下のようにまとめて近似計算を行うことが出来る．

$$\sum_{a\in A(u_t)} P'_a(a;u_t)\nabla_\omega Q'_a(u_t,a) = \frac{1}{N_s}\sum_{i=1}^{N_s}\left[-X_{Vb}(x_i^1)\right] \tag{3.29}$$

3.5 Bootstrapping 法と MC サンプリング

Bootstrapping 法（Q 学習）を他の学習法と組み合わせる方法を考える．Bootstrapping 法では，TD(λ)法，PG 期待値法，回帰法のような勾配ベクトルの計算は必要でない．しかし，Bootstrapping 法が探索木全体を学習対象としていることから，MC サンプリングのノード選択の際に訪問したノードの局面について学習を行う方式が考えられる．

ただし，学習のための評価関数の特徴量の計算にはそれなりのコストがかかる（KPPT+KKPT のパラメータ数は 5 五将棋で約 730 万，本将棋で約 4 億）．全ての訪問ノードについて学習すると計算コストが過大となってしまうため，訪問毎に一定確率で学習を行うこととする．本論文の実験では，訪問した局面の学習を行う確率を 0.02，局面での学習率を 0.01 とした．

4. 実験による検証

4.1 実験の進め方

3 章で述べた学習法について，5 五将棋[13]を用いて学習実験と評価実験を行う．各学習則を単独で用いる場合については，サンプリングを用いた学習法と，単体局面(root や PV)を用いた学習法を学習させ，それぞれを評価実験で比較しサンプリングの効果を確認する．MC サンプリングを用いると複数の学習法を組み合わせて同時に適用できるため，4 つの学習法の全ての組み合わせについても学習実験を行い，それぞれについて評価実験を行う．実験を行う学習法の組み合わせは以下の通りとなる．

- 単独の学習
  1. TD(λ)法：TD(λ)-Sampling 法，TD(λ)-Leaf 法
  2. PG 期待値法：PGSampling 法，PGLeaf 法
  3. 回帰法：サンプリング利用(∇V)，出現局面のみ(∇H)
  4. Bootstrapping 法：探索木全体をサンプリングして学習，PV 上の局面を学習，出現局面のみ学習の 3 種．
- 複数の学習法の同時適用
  2 種類の組み合わせ：6 通り
  3 種類の組み合わせ：4 通り
  4 種類の組み合わせ：1 通り

実験には，筆者の開発したコンピュータ将棋ソフトの芝浦将棋 Softmax を 5 五将棋用に改造した「芝浦少将」[14]を用いる．

4.2 学習実験

学習実験では，各学習法について対局をしながら学習の計算を行い，1 局ごとに局面評価関数のパラメータを更新した．対局は 2000 局行い，対局相手には学習前のソフトと学習中のパラメータを参照するソフトを用いた．対局時の持ち時間は 1 手あたり 0.5 秒としている．学習時のハイパーパラメータは，3.1 で述べた値を基本的に用いているが，学習条件によってはパラメータの調整を行っている．詳しい実験条件の設定などは文献[15]に記した．

単独で学習した場合の勝率の推移を図 2～図 5 に示す．ただし，図中では対局 100 局毎の学習前のソフトに対する勝率の区間平均をプロットしている．

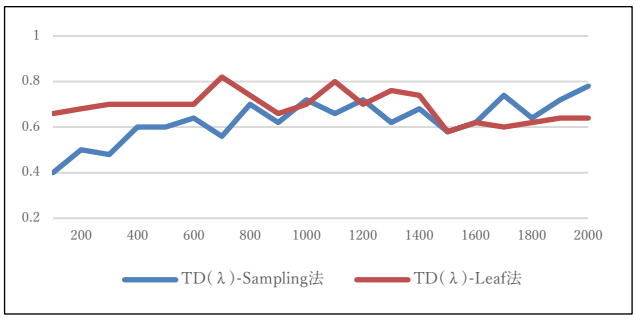

図 2　TD(λ)法の学習中の勝率の推移

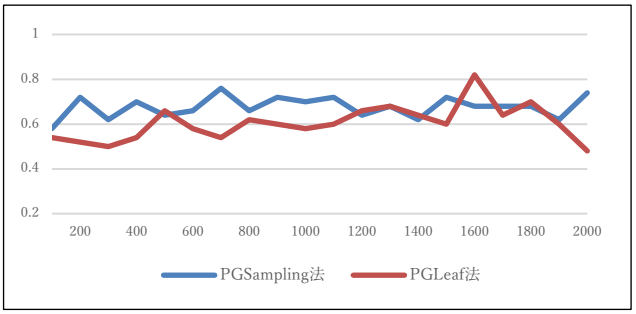

図 3　PG 期待値法の学習中の勝率の推移

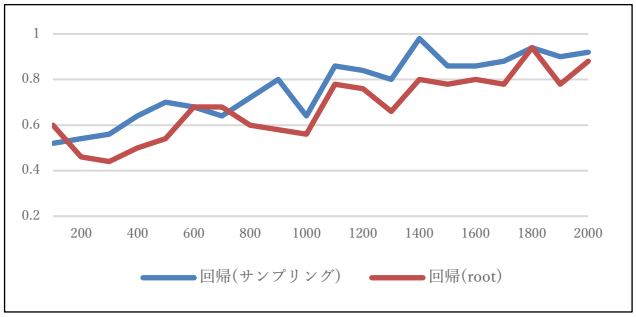

図 4　回帰法の学習中の勝率の推移

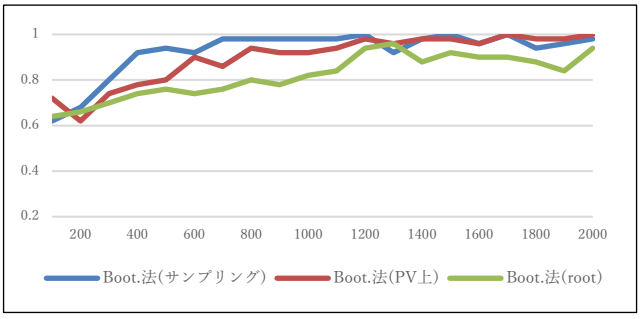

図 5　Bootstrapping 法の学習中の勝率の推移

複数の学習法を組み合わせた場合の学習実験の結果を図 6～図 8 に示す．ただし，図 8 では比較対象として，サンプリングによる Bootstrapping 法(図 5)と，回帰法と Bootstrapping 法の同時学習(図 6)の学習中の勝率の推移もプロットしている．

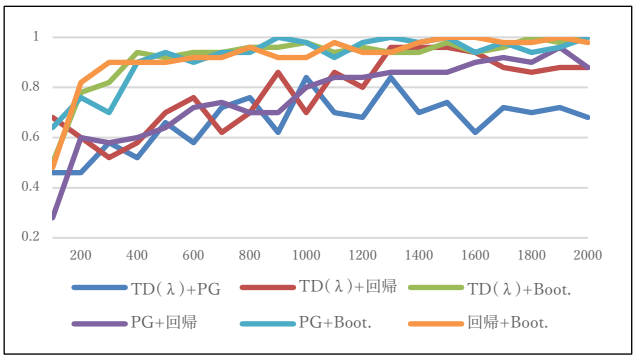

図 6　2 種類の学習法の同時学習での学習中の勝率の推移

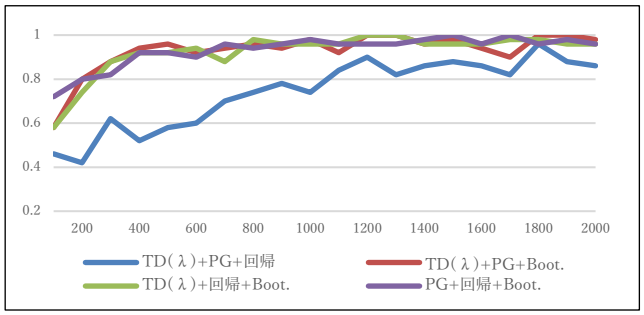

図 7　3 種類の学習法の同時学習での学習中の勝率の推移

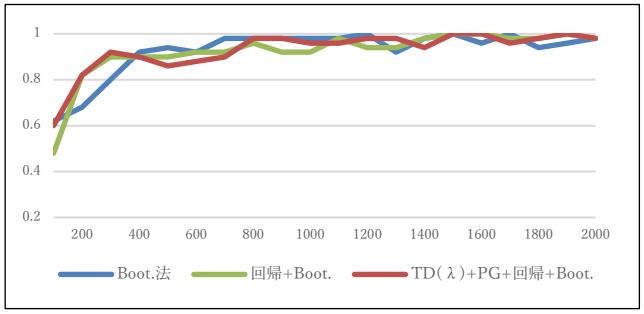

図 8　4 種類の学習法の同時学習での学習中の勝率の推移

4.3 評価実験

評価実験では，学習後のソフトを対局させて，その勝率から学習法の評価を行った．それぞれの対局は先後を均等に 100 局行い，持ち時間は 1 手あたり 0.5 秒とした．

まず，単独の学習法の場合の学習前のソフト（芝浦少将）との対局結果を表 1 に示す．数値は本研究のサンプリングを用いたソフトと従来法によるソフトの勝敗である．

表 1　各学習法の学習前ソフトとの勝率

	サンプリングを使用		PV 上の局面を学習する従来法	
学習法	勝	負	勝	負
TD(λ)法	89	11	*78	22
PG 期待値法	90	10	*55	45
回帰法	100	0	**100	0
Bootstrapping 法	99	1	99	1
	-	-	**95	5

*PV 末端局面のみを学習．**出現局面のみを学習

TD(λ)法，PG 期待値法，回帰法について，本研究であるサンプリングを用いて学習したソフトと，PV 上の局面を学習する従来法によるソフトとの直接対局の結果を表 2 に示す．

表 2　従来法との直接対局の勝率

学習法	勝	負
TD(λ)法	**53**	47
PG 期待値法	**90**	10
回帰法	**66**	34

サンプリングを用いた Bootstrapping 法については，従来法が 2 種類ある(PV 上の局面または出現局面のみを学習する場合)ので，それぞれとの対局結果を表 3 に示す．

表 3　Boot.法の学習局面が異なるソフトとの勝率

学習局面	勝	負	対戦相手
サンプリング	**80**	20	PV 上の局面
サンプリング	**89**	11	出現局面(root)のみ

表 4〜表 6 に，複数のサンプリングを用いた学習法を同時に実行したソフトの対局結果を示す．ただし，対戦相手はその組み合わせに含まれる部分的な組み合わせでサンプリングを用いて学習したソフトである．学習法の組み合わせについて，1.〜4.の数字は学習の種類を表しており，1.が TD(λ)法, 2.が PG 期待値法, 3.が回帰法, 4.が Bootstrapping 法である．●はその学習法を用いていることを表す．

表 4　2 つの学習法を同時に行う学習の勝率

学習法						対局相手			
1.	2.	3.	4.	勝	負	1.	2.	3.	4.
●	●			46	**54**	●			
				46	**54**		●		
●		●		**86**	14	●			
				51	49			●	
●			●	**92**	8	●			
				54	46				●
	●	●		**86**	14		●		
				54	46			●	
	●		●	**91**	9		●		
				53	47				●
		●	●	**82**	18			●	
				49	51				●

表 5　3 つの学習法を同時に行う学習の勝率

学習法						対局相手			
1.	2.	3.	4.	勝	負	1.	2.	3.	4.
●	●	●		**88**	12	●			
				86	14		●		
				51	49			●	
				75	25	●	●		
				45	**55**	●		●	
				42	**58**		●	●	
●	●		●	**95**	5	●			
				95	5		●		
				53	47				●
				82	18	●	●		
				39	**61**	●			●
				54	46		●		●
●		●	●	**91**	9	●			
				80	20			●	
				59	41				●
				80	20	●		●	
				52	48	●			●
				53	47			●	●
	●	●	●	**94**	6		●		
				81	19			●	
				51	49				●
				73	27		●	●	
				58	42		●		●
				61	39			●	●

表 6　4 つの学習法を同時に行う学習

学習法						対局相手			
1.	2.	3.	4.	勝	負	1.	2.	3.	4.
●	●	●	●	**88**	12	●			
				93	7		●		
				84	16			●	
				57	43				●
				90	10	●	●		
				83	17	●		●	
				54	46	●			●
				82	18		●	●	
				66	34		●		●
				69	31			●	●
				81	19	●	●	●	
				54	46	●	●		●
				55	45	●		●	●
				51	49		●	●	●

4.4 実験の考察

4.2, 4.3 の実験結果から考察すると以下のように纏められる：

① 図 2～図 5，表 1 から，4 つの学習法について，MC サンプリングを用いた学習法が実際に可能であることが分かった．MC サンプリングを用いた学習が学習前のソフトに対して 89%~100%の勝率となっており，棋力が向上したことがわかる．

② 表 2,表 3 から，MC サンプリングを用いた学習法が，従来法（PV の末端や経路上の局面あるいは root 局面のみを学習）よりも効果があった．それぞれの学習法での直接対決では，従来法に対して 53%～90%の勝率となった．

③ 表 6 では 4 つの複合学習に対する単独の学習法は，Bootstrapping 法が 43%, 回帰法が 16%, TD(λ)法が 12%, PG 期待値法が 7%の勝率となっていた．したがって，サンプリングを用いた 4 つの学習法の学習効率の順序は，Bootstrapping 法，回帰法，その次に TD(λ)法と PG 期待値法が同程度で並んでいると考えられる．

④ 表 4～表 6 より，サンプリングを用いた学習法を複数同時に適用すると，最終的な棋力向上についての効果が高い．多くの学習法を同時に実行するほど棋力が高くなる傾向が全般的にみられる．

⑤ 表 6 より，4 つの学習法全てを同時に行う学習は，単独では最も強かった Bootstrapping 法よりも 57 勝 43 敗と勝ち越した．この差は格段に大きいわけではないが，学習開始 400 局ぐらいまでの学習初期においては複合学習の方が学習効率が良く，少ない学習回数で早く棋力が向上している．

5. まとめ

本論文では，モンテカルロ・ソフトマックス探索のために考案された MC サンプリングを用いた学習方式を 5 五将棋に適用し，学習実験を行った．その結果，学習則の正当性，MC サンプリングの有効性，複数の強化学習法へ同時適用した際の有効性を示すことができた．また，本事例では，サンプリングによる勾配ベクトルの計算時間は探索木生成程度の時間で済むことが実験的に示された．

今後は，本将棋や他のゲームへの適用や，2 人対戦ゲームだけではなく，一般的なエージェントの方策学習の問題へ理論の枠組みを拡張することを予定している．さらに，これまでは，探索が完了して生成された探索木に対して，勾配ベクトルの計算や学習を行っていたが，高速化のために探索を行いながら勾配ベクトルを計算する方式も考案中である．

参考文献

[1] 小谷善行，岸本章宏，柴原一友，鈴木 豪．ゲーム計算メカニズム，コロナ社，2010, 190p.
[2] Rémi Coulom. Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search. Computers and Games, 5th International Conference, CG 2006, Turin, Italy, May 29-31, 2006. p. 72–83.
[3] Gelly, S., Wang, Y., Munos, R., Teytaud, O.. Modification of UCT with Patterns in Monte-Carlo Go. Technical report, INRIA, 2006.
[4] Silver, D., et al.. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. SCIENCE, 2018, Vol 362, Issue 6419, p. 1140-1144.
[5] 桐井杏樹，原悠一，五十嵐治一，森岡祐一，山本一将．確率的選択探索の将棋への適用．ゲームプログラミングワークショップ 2017 論文集, 2017, vol.2017, p.26-33.
[6] 五十嵐治一，森岡祐一，山本一将. MC Softmax 探索における局面評価関数の学習．ゲームプログラミングワークショップ 2018 論文集, 2018, vol.2018, p.212-219.
[7] Sutton, R. S. and Barto, A. G.. Reinforcement Learning: An Introduction. MIT Press, 1998, 344p.
[8] 五十嵐治一，森岡祐一，山本一将．方策勾配法による静的局面評価関数の強化学習についての一考察．ゲームプログラミングワークショップ 2012 論文集, 2012, vol.2012, no.6, p.118-121.
[9] 五十嵐治一，森岡祐一，山本一将．方策勾配法による局面評価関数とシミュレーション方策の学習．第 30 回ゲーム情報学研究発表会，情報処理学会研究報告, 2013, vol.2013-GI-30, no.6, p.1-8.
[10] Veness, J., Silver, D., Uther, W. and Blair, A.. Bootstrapping from Game Tree Search. Advances in Neural Infomation Processing Systems 22, 2009.
[11] Williams, R. J. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. Machine Learning, 1992, vol8, p.229-256.
[12] 森岡祐一，五十嵐治一．方策勾配法とαβ探索を組み合わせた強化学習アルゴリズムの提案．ゲームプログラミングワークショップ 2012 論文集, 2012, vol.2012, no.6, p.122-125.
[13]“５五将棋”．電気通信大学 伊藤毅志研究室．http://minerva.cs.uec.ac.jp/cgi-bin/uec55shogi/wiki.cgi?page=5%B8%DE%BE%AD%B4%FD,（参照 2022-02-20).
[14]“MCSS_55Shogi”．https://github.com/tanuki12hiromasa/MCSS_55Shogi,（参照 2022-2-21).
[15] 岩本裕大, MC Softmax 探索における局面評価関数の強化学習―５五将棋への適用―．芝浦工業大学大学院修士論文, 2022.