特徴の生成を組み合わせた機械学習

佐 藤 佳 州[†1,†2] 高 橋 大 介[†1]

近年，ゲームプログラミングの分野において機械学習は急速に発展しており，評価関数の重みの決定，探索深さの調整，モンテカルロシミュレーションにおける指し手の予測など，幅広い課題に対して有効であることが示されている．現在のゲームプログラミングにおける機械学習は，あらかじめ評価項目（特徴）を用意し，プロの棋譜などを基にその重みを学習するというものが主流である．このような手法は，人間では調整できないような膨大な数の特徴に対しても，自動的に適切な重みを算出できるという利点がある．一方で，学習に用いる特徴自体は人間が手動で設計する必要があり，性能を決定する大きな要因となっている．現在，特徴の設計に関しては，人手による試行錯誤的な調整が行われているが，機械学習に有効な特徴を手動で生成することは一般的に非常に困難な問題である．本論文では，この問題を解決するため，特徴の自動生成と機械学習の重み付けを組み合わせることにより，有効な特徴を生成する手法を提案する．具体的には，機械学習の反復計算の過程に特徴の生成を組み込むことで，性能の向上を目指す．実験の結果，提案手法が現在ゲームプログラミングの分野で用いられている各学習手法において有効であることを示した．

A Machine Learning Method Combined with a Feature Generation

Yoshikuni Sato[†1,†2] and Daisuke Takahashi[†1]

Recently, machine learning has progressed rapidly in the field of game programming. It has been shown to be effective against a wide range of issues, for example, determining the weights of the evaluation function, adjustment of search depth and a move selection in Monte-Carlo simulation. Current machine learning methods in game programming use pre-assessment features and learn its weight based on the training set such as records of professional players. Such methods have the significant advantage to be able to calculate appropriate weight for a large number of features that human cannot adjust. On the other hand, it is necessary for human to design the feature manually, and it becomes the major factor determines the performance of programs. Currently, features are manually adjusted by trial-and-error, however, it is very difficult to generate the effective features for machine learning. In this paper, we propose a method of generating the effective features by combining automatic feature generation and machine learning to solve this problem. In our method, we improve performance by including feature generation in the process of the repeated calculation of machine learning. The experimental results show the effectiveness of our approach for each learning method that recently used in game programming.

1. はじめに

ゲームプログラミングの分野において，機械学習は近年急速に進歩している．近年の代表的な成功例としては，将棋の評価関数のパラメータ調整[1]や，実現確率による探索深さの調整[2]などがあげられる．これらの課題は，従来人間がゲーム固有の知識を用いて手作業で調整を行っていた部分であるが，現在ではプロの棋譜を教師とした学習により，適切な値を自動で調整できるようになっている．これらの学習手法は，現在ではトップレベルの強さを持つ多くの将棋プログラムで採用されるほど普及している．

また，近年囲碁の分野ではでのモンテカルロ木探索[3]が大きな注目を集めている．この手法では，ゲームのシミュレーション（playout）が性能の決定に大きな影響を及ぼすことが知られている．この playout 中の指し手の選択においても各種の学習手法が使用されており，成功を収めている[4][5]．

このように現在では，機械学習はゲームプログラミングにおける幅広い課題に対して有効であることが示されている．

現在のゲームプログラミングの分野における機械学習は，あらかじめ評価項目（特徴）を用意し，プロの棋譜などを基にその重みを学習するというものである．

†1 筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering, University of Tsukuba
†2 パナソニック株式会社先端技術研究所
Advanced Technology Research Laboratories, Panasonic Corporation

このような学習手法は，人間では調整できないような膨大な数の特徴に対しても，自動的に適切な重みを計算できるという大きな利点がある．一方で，学習に用いる特徴自体は人間が手動で設計する必要があり，プログラムの性能を決定する大きな要因となっている．機械学習に有効な特徴を手動で生成することは非常に困難な問題であり，現在は試行錯誤的に調整が行われている．ただし，このような手法は，特徴の選択にゲーム固有の専門的な知識が必要になる，また，特徴セットを用意するたびに評価を行う必要があるため非常に多くの時間を要する，といった問題を有していた．このような背景から，学習に用いる特徴自体の生成・選択も機械で自動的に行うことのできる手法の実現が望まれている．

本論文では，この問題を解決するため，特徴の生成と機械学習の重み付けを組み合わせることにより，有効な特徴を生成する手法を提案する．

2. 関 連 研 究

機械学習における特徴の自動生成，自動選択は古くから数多くの研究が行われてきた分野であり，ゲームの分野においても研究が行われている[6)7)8)]．機械学習において特徴選択を行う主な目的は，予測精度の改善と，不必要な特徴の削減による効率化の 2 点である．

これらの手法は大きく分けて，特徴のテンプレートを固定してその中から有効な特徴を選択することを目的としたものと，特徴のテンプレートを固定せず学習の目的に応じて有効な特徴自体を生成することを目的としたものに分けられる．

2.1 特徴の自動選択

特徴の自動選択は，ゲームプログラミングに限らず，機械学習の分野では広く研究されているテーマである．特徴のテンプレートを固定した特徴選択は大きく分けて，filter 法，wrapper 法，embedded 法が存在する[9)]．filter 法は，学習を行う前処理として，特徴の有効さを示す指標に基づき，あらかじめ特徴選択を行う手法である．一般的にこの手法は高速であるが，精度では後述する wrapper 法や embedded 法に劣る．wrapper 法とは，特徴の部分集合（特徴セット）を作成し，実際に学習を行った上で特徴セットを評価する手法である．評価の方法としては Cross Validation などが用いられる．この手法は，実際に学習と評価を行った上で特徴セットを作成するため予測精度は高くなるが，場合によっては非常に時間を要するという問題が存在する．embedded 法は，学習に特徴選択を組み込む手法であり，L1 正則化[10)] による特徴選択なども embedded 法に相当する．embedded 法は学習と同時に特徴選択が行われるため高速であり，精度も良いため広く用いられている．

2.2 特徴の自動生成

一方，近年のゲームプログラミングの分野では，用意された特徴テンプレートからの選択ではなく．特徴そのものの生成に関する研究が数多く行われている．

文献 6) では，実現確率に類似した考え方を用いることにより，ゲームに有効と考えられる，高次の組み合わせ特徴の生成を可能としている．また，文献 7) では，カーネル法を用いることにより，人間があらかじめ複雑な特徴を用意しなくても．高次な特徴量空間での学習を可能にしている．また，文献 8) のようにゲーム特有の知識を用いない General Game Playing に向けた特徴の生成に関する研究も存在する．

ただし，これらの手法により生成された特徴は，精度面，速度面などに問題があることも多く，一般的に広く利用されるまでには至っていないものが多い．また，これらの手法は，複雑な特徴を作ることにはある程度成功しているが，その妥当性を保証する手法とはなっていない．

2.3 現在のゲームプログラミングにおいて利用される特徴とその課題

現在ゲームプログラミングの分野でよく用いられている特徴の生成は，大きく分けて，以下の 3 つに分類される．

(1) ゲームの知識に基づく特徴
(2) 単純な位置関係の組み合わせによる特徴
(3) 人間の棋譜で頻出するパターンを抽出した特徴

（1）の特徴は，例えば将棋の指し手を考えた場合，「駒の損得」「王手」「成る手」といったものである．これらの特徴は，一般的に学習において有効であることが多いが，深いゲームの知識が必要となり，数多くの特徴を用意することも困難である．

膨大な数のパラメータを調整できるという機械学習のメリットを考えたとき，重要となるのは（2），(3）の特徴である．(2）の特徴としては，例えば Bonanza の評価関数で用いられている 3 駒間の関係などが存在する[11)]．(3）の特徴は，モンテカルロ木探索による囲碁において用いられることが多い[4)]．現在は，上記 (1)〜(3) のような特徴を用い，必要に応じて正則化を組み合わせた学習を行うのが一般的である．

また，どのような特徴を使用するかによって学習結果には大きな差が生じるが，その特徴の取捨選択は現在は試行錯誤により行われることが多い．この特徴選択の部分は，現在のプログラムの性能を決定する大きな要因になっており，この部分を自動で適切に調整可能とすることは大きな課題である．

本論文では，この問題を解決するため，特徴の生成と機械学習の重み付けを組み合わせることにより，有効な特徴の選択・生成を行う手法を提案する．また，近年ゲームプログラミングの分野で成功を収めた学習手法に提案手法を適用し，その効果を検証する．

3. 提 案 手 法

前節までに述べたように，ゲームプログラミングにおける機械学習では，特徴をあらかじめ生成しておき，その特徴の重みを求めるといったように，特徴の生成と機械学習による重み付けは完全に切り離されている．機械学習の種類によっては，L1 正則化による特徴の自動選択等も行われているが，このような手法は，用意された特徴を絞り込むためのものであり，新たな特徴が生成されることはないため，十分な特徴生成・選択手法とは言えない．

本論文では，以上の問題点を解決するため，特徴の生成と機械学習の重み付けを組み合わせることにより，有効な特徴を生成する手法を提案する．現在の機械学習では，プロの棋譜を教師として，その指し手に近づけるように特徴の重みを調整している．本提案手法では，この重みの調整に加え，特徴の生成も同時に行い学習の精度を高めることを目的とする．

具体的には，機械学習による重み付けの過程で，正例（指し手，局面）よりも良い評価を返した負例が存在する場合，その正例および正例よりも良い評価を返した負例から特徴の抽出を行う．従来の機械学習手法は，このような例の場合，正例の評価を上げる，あるいは負例の評価を下げるように特徴の重みを調整していた．しかし，そもそも正例と負例を分離するために有効な特徴が用意されていない場合，どのような学習手法を使って重みを調整しても精度を上げることは不可能である．本提案手法は，このような点に着目し，重みの調整と特徴の生成を同時に行うことで，機械学習の性能向上を目的とする．以下に具体的な手順を示す．

(1) 棋譜中の頻出パターンなど従来の特徴を用いて，機械学習による重み付けを行う
(2) 機械学習による重みの反復計算中で，予測が当たらなかった局面において，正解手（局面），および正解手よりも良い評価を返した不正解手から，新たな特徴を抽出する
(3) (2) で抽出された特徴のうち，出現頻度が一定以上のものを新たな特徴セットとして追加する
(4) 生成された特徴を含めて重みの学習を継続する (2)〜(4) を繰り返し

以上の手順により，特徴の生成，機械学習による重み付け，特徴の取捨選択を同時に行う．提案手法では，機械学習による重み付けの結果をフィードバックすることにより，学習に必要な特徴の生成を可能にしている．本提案手法のポイントは正解手よりも良い評価を返した不正解手（局面）のみから特徴を抽出する点である．このような処理により，平易な特徴のみで学習可能な局面からは無駄な特徴を抽出せず，逆に複雑な局面のみから有効な特徴を抽出することを可能としている．

なお，本論文中では，特徴のテンプレートは固定した条件により実験を行う．提案手法では，特徴のテンプレートを固定する必要はないが，関連研究でも述べたように，高度な特徴の生成はそれ自体が非常に難しい課題となっている．そのため，本論文中では特徴のテンプレートを固定した条件のもとで，特徴の生成と学習を組み合わせるという提案手法の有効性を検証する．

また，本提案手法は，基本的に有効な特徴を生成する（増やす）ことにより，性能の改善を目指す手法であるため，正則化による特徴の選択との併用が可能である．この場合，提案手法により必要と判断され生成された特徴が，正則化の効果によって選択（削除）されることになる．以降の実験においても必要に応じて，提案手法と正則化は併用して用いることとする．

4. 実験対象の学習手法

本論文では，将棋を題材に以下の 3 種類の機械学習について提案手法を適用し，有効性を検証する．

(1) モンテカルロ木探索における Elo rating を用いた指し手の予測
(2) 実現確率探索における Logistic 回帰を用いた指し手の予測
(3) Bonanza の学習手法を用いた評価関数の学習

以下で，それぞれの学習手法について，簡単な説明を行う．

4.1 モンテカルロ木探索における Elo rating を用いた指し手の予測

モンテカルロ木探索は近年，主に囲碁の分野で大きな成功を収めた手法である．将棋など他のゲームへの適用に関する研究も数多く行われており[12)13)]，現在ゲームプログラミングの分野で最も注目を集めている手法の一つである．

モンテカルロ木探索は，乱数を用いたゲームのシミュレーション（以下 playout）と，木探索を組み合わせた手法となっている．モンテカルロ木探索では，playout 中の指し手の選択に知識を導入することで，性能を大幅に向上させることができることが知られている[4)5)]．この中でも文献 4) で示される，Elo rating を用いた指し手の選択は，囲碁および将棋において有効であることが示されており，モンテカルロ木探索において一般的に使用される手法の一つとなっている．

4.2 Elo rating による指し手の予測

Elo rating による指し手の予測は，試合の勝敗を予測するモデルの一つである Bradley-Terry モデルに基づく．あるプレイヤ i が γ_i という正の値（レーティング）を持つとき，1 から n 人までのプレイヤの中で i $(1 \leq i \leq n)$ が勝利する確率は式 (1) で表される．

$$P(i \text{ wins}) = \frac{\gamma_i}{\sum_{j=1}^{n} \gamma_j} \quad (1)$$

このモデルはチーム間の勝敗予測に拡張可能である．あるチームのレーティングは構成プレイヤのレーティングの積で表現される．例えば，プレイヤ 1, 2, 3 で構成されるチームのレーティングは，$\gamma_1\gamma_2\gamma_3$ となる．

将棋の指し手には,「駒の損得」,「王手」や「逃げる手」など様々な特徴がある．これらの特徴を個々のプレイヤと見なし，各特徴の選択されやすさをプレイヤの強さ（レーティング）と考えることで，上記のモデルを適用できる．この場合，指し手は複数の特徴からなるチームと考えることができる．このモデルを利用して playout 中の指し手の選択を行うことを考えた場合，指し手の選択確率はチームが勝つ確率と対応する．

各特徴のレーティング γ_i は，式 (2) の反復計算により求める[3]．

$$\gamma_i \leftarrow \frac{W_i}{\sum_{j=1}^{N} \frac{C_{ij}}{E_j}} \quad (2)$$

ここで，N は学習局面の数，W_i は特徴 i を持つ手が選択された回数，C_{ij} は局面 j における特徴 i が属する指し手の他の特徴のレーティング，E_j は全指し手のレーティングを表す．

4.3 Bonanza の学習手法を用いた評価関数の学習

ゲームプログラミングにおいて，評価関数の機械学習は古くから研究されていた課題であるが，将棋において初めてその課題に成功したのが Bonanza である．Bonanza の評価関数の学習手法は Bonanza メソッドとも呼ばれ，PV（Principal Variation）の生成と評価関数の重みの学習を繰り返し行うことを特徴としている．Bonanza の学習では，具体的には式 (3) の目的関数を最適化する．

$$J(P, v) = \sum_{p \in P} \sum_{m=2}^{M_p} T_p[\xi(p_m, v) - \xi(p_1, v)] + \lambda g(v) + \|w\| \quad (3)$$

P は学習対象の局面集合，M_p は局面 p における合法手数，p_1 は記譜中で実際に指された手，p_m はそれ以外の手，$\xi(p_m, v)$ は p_m を選択した際の探索結果の評価値を示す．また T_p は損失関数，$\lambda g(v)$ は拘束条件，$\|w\|$ は正則化項を示す．上記の式では，L1 正則化を用いている．本論文中でも，Bonanza の手法を用いた評価関数の学習では L1 正則化を用いる．

4.4 実現確率探索における Logistic 回帰を用いた指し手の予測

実現確率探索（実現確率による探索打ち切りアルゴリズム）[2] は鶴岡によって提案された手法である．実現確率探索では，プロの棋譜を基にした学習により，その手がどの程度の確率で指されそうか（遷移確率）を求め，その遷移確率によって探索の深さを制御するというものである．

実現確率の学習は，従来は単純に実際の棋譜における選択確率を用いていたが，近年は logistic 回帰による予測が良い結果を得ており，主流となっている[14]．

この手法では，n 個の特徴が存在し，i $(1 \leq i \leq n)$ 番目の特徴の値を x_i とすると，特徴の値 $(x_1, x_2, ..., x_n)$ を持つ指し手の遷移確率 p は式 (4) で表される．

$$p(x_1, x_2, ..., x_n) = \frac{1}{1 + \exp(-\sum_{i=1}^{n} w_i x_i)} \quad (4)$$

ロジスティック回帰による実現確率探索では，プロの棋譜から各特徴が選択される割合を求める代わりに，式 (4) における各特徴の重み w_i を学習により求める．学習の際には，プロの棋譜において実際に指された手を正例，指されなかった手を負例とすることで，各特徴の重み w_i を推定する．本論文中では，重み w_i の算出には LIBLINEAR[15] を提案手法用に変更したものを用いている．具体的には式 (5) の最適化を行う．

$$\min_{w} \|w\|_1 + C \sum \log(1 + e^{-y_i w^T x_i}) \quad (5)$$

上記の式では，L1 正則化を行っている[⋆1]．以降の実験においても，ロジスティック回帰では L1 正則化を用いる．

5. 実　　験

本論文では，前節で説明した 3 つの学習手法に対して提案手法を適用し効果を検証する．以降では，まず実験に使用する特徴の説明を行い，その後実験内容について説明する．

5.1 実 験 条 件

5.1.1 実験に用いる特徴

本論文の実験において，使用する指し手の特徴および評価関数の特徴を以下に示す．

指し手の特徴

- 駒の損得（SEE）
- 王手
- 成る手
- 駒の取り返し
- 逃げる手
- 移動元，移動先の駒の絶対位置
- 移動元，移動先の周囲 3×3 の駒の配置のパターン
- 利きの関係に基づく駒の位置関係のパターン

評価関数の特徴

- 駒割
- 自玉，相手玉との位置関係
- 3×3 の駒の配置のパターン
- 利きの関係に基づく駒の位置関係のパターン

利きの関係に基づく駒の位置関係のパターンは，利

⋆1 LIBLINEAR のオプションは「-s 6」

きの重なりのある 2 駒以上の駒の位置関係のパターンを表す特徴である．具体的には，あるマスに着目した時，そのマスに利きのあるすべての駒の位置関係を特徴として利用する．利き関係に基づく駒の位置関係の例を**図 1** で示す．

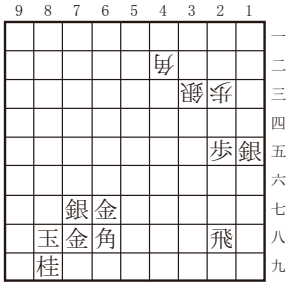

図 1 利きに基づく駒の配置のパターンの例

図 1 において，例えば 2 四の地点に着目すれば（▲２五歩，▲１五銀，▲６八角，▲２八飛，△２三歩，△３三銀，△４二角）というパターンが，７七の地点に着目すれば（▲８九桂，▲７七銀，▲６七金，▲７八金，▲６八角，▲８八玉）というパターンが抽出される．この方法では，部分的ではあるが，かなり多くの数の駒の位置関係も表現でき，かつ利きの情報に基づいているため，意味のあるパターンを生成しやすいという利点がある．さらに利き情報を持つデータ構造を使用しているプログラムでは，非常に高速にパターンを生成することが可能である．

なお，3×3 の駒の配置のパターン，利き関係に基づく駒の位置関係パターンはハッシュを用いて実装している．この方法では，特徴の数が膨大になっても高速な処理が可能という利点がある．

5.1.2 実 験 内 容

本実験では，従来手法と提案手法において，一致率（正解手を当てた確率）の比較，及び対局による評価を行った．従来手法としては，頻出パターンにより特徴セットを生成する手法を用いている．なお提案手法，及び従来手法ともに，L1 正則化等の一般的な機械学習で用いられている特徴選択は行うものとする．

対局実験では，以下の 2 つ種類の評価を行う．

(1) 頻出パターンを用いたプログラムとの対局実験（生成された全特徴を用いた場合）
(2) 特徴セットの数を制限した条件において，考えられる全特徴を用いたプログラムとの対局実験

（1）の実験では，大きな特徴セット（特徴テンプレート）を用いて，現在よく用いられている頻出パターンを利用したプログラムとの対局により提案手法の有効性を検証する．(2) の実験では，小さな特徴セットを用いて，提案手法を用いて特徴を生成した場合に，考えうる全特徴を用いた場合と比較してどの程度の性能が得られるかを検証する．具体的には，(1) の実験では，先に説明したすべての特徴を用いて実験を行い，(2) の実験では，3×3 の駒の位置関係パターン，利きに基づく駒の配置のパターンを除いた特徴テンプレートを用いて実験を行う．

各手法とも，学習にはプロの棋譜 8,000 局を用い，一致率の評価には学習用の棋譜とは別の棋譜 2,000 局を用いた．また，対局実験は定跡で 24 手目まで進めた局面（同一局面は除く）から開始し，先後を入れ替えて 200 セット，計 400 局の対局を行った．

5.2 モンテカルロ木探索における Elo rating を利用した指し手の予測

モンテカルロ木探索における，Elo rating を用いた指し手の予測に提案手法を適用した効果を検証した．実験にはアマチュア 1 級程度のモンテカルロ木探索によるプログラムを用いた[⋆1]．このプログラムは基本的に文献 13) の手法に基づいたものであり，UCT (Upper Confidence bounds applied to Trees)[16] に progressive widening 等の UCT の一般的な改良手法を組み合わせている．末端での評価関数の利用，静止探索等の利用などは行っていない．

提案手法および従来手法（頻出パターン）を用いた場合の指し手の一致率と生成された特徴数を**表 1** に，対局実験の結果を**表 2** にそれぞれ示す．対局実験の思考は 1 手 10,000 playout とした．

表 1 一致率および特徴数（モンテカルロ木探索）

手法	一致率	特徴数
頻出パターン	0.332	329,768
提案手法	0.344	4,416,460

表 2 提案手法の頻出パターンを用いたプログラムに対する勝率（モンテカルロ木探索）

手法	勝敗（勝率）
提案手法	250 勝 142 敗 8 分 (0.637)

実験結果から，提案手法が一致率による評価および対局実験による評価において，頻出パターンを用いた従来手法を上回っていることがわかる．特に，対局実験による評価では，提案手法が従来手法を大きく上回る結果となった．また，表 1 からわかるように，本実験では L1 正則化等による特徴の絞り込みは行っていないため，特に提案手法では特徴数が非常に多くなっている．

次に，特徴セットを限定した条件における，全特徴を用いたプログラムとの対局実験の結果を**表 3** に示す．特徴数の割合は全特徴を用いた場合の特徴数に対する各種法を用いた場合の特徴数の割合を示す．

全体の特徴数が少ないこともあり，対局結果に大きな違いはなかったものの，頻出パターンは全特徴を用いた場合に負け越し，提案手法は全特徴を用いた場合

⋆1 強さは次の一手問題の結果等から推定

表 3 全特徴を用いたプログラムに対する勝率（モンテカルロ木探索）

手法	勝敗（勝率）	特徴数の割合
頻出パターン	185 勝 206 敗 9 分 (0.473)	0.522
提案手法	207 勝 190 敗 3 分 (0.521)	0.769

に勝ち越すといった結果となった．この結果から，頻出パターンでは，必要な特徴を生成し切れていない可能性があると考えられ，逆に提案手法では，全特徴を用いた場合と同等以上の性能を得ており，十分な特徴を生成できていると考えることができる．

5.3 実現確率探索における logistic 回帰を用いた指し手の予測

実現確率探索における Logistic 回帰を用いた指し手の予測に提案手法を適用し，効果を検証した．

実験には，アマチュア四段程度の通常探索（alpha-beta 法ベース）のプログラムを用いた．探索部分は PVS（Principal Variation Search）をベースとし，各種枝刈り（null move pruning, history pruning, futility pruning）を行っている．また，評価関数については，3 駒間の関係を Bonanza の学習手法により求めたものを利用している．本実験では，このプログラムの探索部分に実現確率探索を適用し，実験を行う．

一致率，及び生成された特徴数の実験結果を**表 4** に，対局実験の結果を**表 5** に示す．対局実験は 1 手 1,000 万ノードとした．

表 4 一致率および特徴数（Logistic 回帰）

手法	一致率	特徴数
頻出パターン	0.329	362,570
提案手法	0.335	537,015

表 5 提案手法の頻出パターンを用いたプログラムに対する勝率（実現確率探索）

手法	勝敗（勝率）
提案手法	194 勝 155 敗 51 分（0.555）

実験結果から，一致率及び対局実験の評価において，提案手法が従来手法を上回っており，logistic 回帰を用いた実現確率探索においても提案手法は有効であると考えられる．ただし，提案手法の導入による効果はモンテカルロ木探索での実験よりも小さかった．値が非ゼロの特徴数は，5.2 節の実験と比較して大幅に少なくなっていることがわかる．これは，logistic 回帰の学習過程で，L1 正則化を併用しているためである．

特徴セットを限定した条件における，全特徴を用いたプログラムとの対局実験の結果を**表 6** に示す．

表 6 全特徴を用いたプログラムに対する勝率（実現確率探索）

手法	勝敗（勝率）	特徴数の割合
頻出パターン	162 勝 191 敗 47 分 (0.458)	0.630
提案手法	175 勝 166 敗 59 分 (0.513)	0.688

本実験の結果は 5.2 節のモンテカルロ木探索の実験と同様の傾向となった．この実験においても，頻出パターンを用いた手法は，全特徴を使用したプログラムに劣っており，頻出パターンのみでは，必要な特徴生成し切れていないと考えられる．

5.4 Bonanza の学習手法を用いた評価関数の学習

Bonanza の学習手法を用いた評価関数の学習について提案手法を適用し，効果を検証した．

実験に用いたプログラムは，基本的には 5.3 節のものと同様であるが，探索部分は通常の PVS とし，評価関数を実験用に差し替えている．今回の実験では実験を簡単にするため，学習における PV（Principal Variation）の生成は静止探索のみで行った．

一致率および特徴数の結果を**表 7** に，対局実験の結果を**表 8** に示す．対局実験における思考は 1 手 1,000 万ノードとした．

表 7 一致率および特徴数（評価関数）

手法	一致率	特徴数
頻出パターン	0.343	639,742
提案手法	0.350	1,130,519

表 8 提案手法の頻出パターンを用いたプログラムに対する勝率（評価関数）

手法	勝敗（勝率）
提案手法	212 勝 166 敗 22 分 (0.560)

実験の結果，一致率，対局実験の結果とも提案手法が従来手法を上回っており，評価関数の学習において提案手法は有効であると考えられる．

特徴セットを限定した場合における，全特徴を用いたプログラムの対局結果を**表 9** に示す．

表 9 全特徴を用いたプログラムに対する勝率（評価関数）

手法	勝敗（勝率）	特徴数の割合
頻出パターン	176 勝 200 敗 24 分 (0.468)	0.676
提案手法	178 勝 180 敗 42 分 (0.497)	0.783

実験の結果，5.2 節，5.3 節の実験と同様の傾向の結果が得られた．これらの結果から，提案手法は少なくとも比較的小さな特徴セットの調整においては，特徴数を抑えつつ全特徴を生成した場合とほぼ同等の性能を得ることができると考えられる．

5.5 局面の進行度に応じた一致率の変化

前節までの実験の結果，特にモンテカルロ木探索における Elo rating を用いた指し手の予測において提案手法の効果が大きかった．本節では，詳細な分析を行うため，Elo rating を用いた指し手の予測における一致率を進行度別に求めた．結果を**図 2** に示す．

ここでの進行度とは，ある時点の手数が，その対局の総手数を 10 分割したとき，どの範囲に属するかを意味している．図 2 では，評価用棋譜での一致率に加え，学習用棋譜での一致率も示している．実験結果か

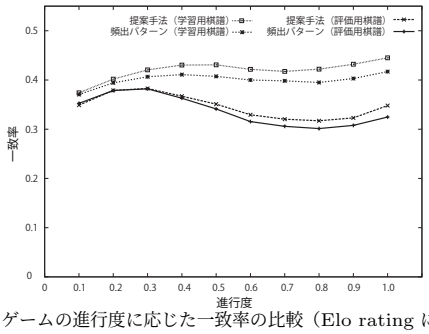

図 2 ゲームの進行度に応じた一致率の比較（Elo rating による指し手の予測）

ら，序盤では従来手法（頻出パターン）を用いた場合と提案手法にほとんど差がないことがわかる．一方で，終盤に近づくに従って，提案手法が従来手法を上回っていくことがわかる．

この結果から，終盤のような複雑な局面を，単純な頻出パターンによる特徴で表現することは難しく，そのような局面において，提案手法が有効に働いている事がわかる．特にモンテカルロ木探索による将棋の場合は，終局条件（詰み）が絡むため，終盤の予測の向上は効果が大きいと考えられ，提案手法が特に効果を発揮した一つの理由と考えられる．

5.6 提案手法と従来手法の実例による比較

本節では，Elo rating による指し手の予測において，提案手法により生成された特徴を用いた場合と，従来手法（頻出パターン）を用いた場合の比較例を示す．**表 10** および**表 11** は，**図 3** の A 図，B 図において Elo rating による指し手の予測を行った際の上位 5 手を示したものである．

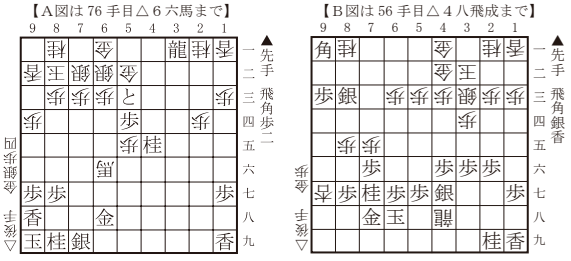

図 3 提案手法を用いた指し手の予測と頻出パターンを用いた指し手の予測の比較（A 図，B 図）

表 10 提案手法を用いた指し手の予測と頻出パターンを用いた指し手の予測の比較１（A 図）

オーダー	提案手法	頻出パターン
1	▲７七歩（0.227）	▲８八銀（0.258）
2	▲７七角（0.183）	▲８八角（0.221）
3	▲８八角（0.155）	▲７七歩（0.139）
4	▲８八銀（0.146）	▲７七角（0.127）
5	▲７七金（0.122）	▲７七金（0.108）

表 11 提案手法を用いた指し手の予測と頻出パターンを用いた指し手の予測の比較２（B 図）

オーダー	提案手法	頻出パターン
1	▲５八銀打（0.247）	▲５八銀引（0.256）
2	▲５八香　（0.135）	▲７九玉　（0.194）
3	▲５八銀引（0.154）	▲５八銀打（0.176）
4	▲７九玉　（0.140）	▲５八香　（0.168）
5	▲５八飛　（0.130）	▲５八飛　（0.132）

A図では，頻出パターンによる予測では，▲８八銀が最上位に上がっている．実際にはこの手は危険な手といえるが，頻出パターン中にはそれを表現するための特徴が存在しないため，高い評価となっている．一方，提案手法による予測ではより安全な▲７七歩が最上位となっている．提案手法により生成した特徴と頻出パターンにより生成した特徴を比較すると，提案手法の場合には頻出パターンの特徴には存在しない（▲９九玉，▲８八銀，△６六馬），(▲９九玉，▲６八金，▲８八銀，▲８九桂，△６六馬）といった特徴が生成されており，玉と接する▲８八銀が△６六馬でピンされることの危険性を評価可能となっている．

B図の場合にもA図と同様，頻出パターンによる予測では，▲５八銀引や▲７九玉といった危険な指し手が上位にあがっている．しかし，提案手法の場合には，頻出パターンには存在しない（▲６八玉，▲５八銀，▲６七歩，▲５七歩，△４八龍）や，(▲７九玉，▲７八金，△４八龍）といったような危険度を認識可能な特徴が生成されており，より適切な予測が可能となっている．

A 図，B 図のように提案手法では，特に終盤における，玉の危険度が高いような局面において有効に働いている．このような利点が，モンテカルロ木探索で最も良い結果を得た一つの要因と考えられる．

6. 今後の課題

本論文における実験では，特徴の生成を機械学習の重み付けの内部に組み込む手法を提案し，実験においてその有効性を示した．

今後の課題としては，特徴のテンプレートを固定しない特徴生成手法との組み合わせがあげられる．今回は生成する特徴として，3×3 の駒のパターン，利きが関連する駒のパターンといった特徴テンプレートを固定した形で実験を行った．これらの特徴は高速で実用的なものであるが，表現力には限界がある．

提案手法では生成する特徴のテンプレートは固定されている必要はなく，例えば文献 6) 等の手法を用いることにより，より複雑な特徴を取り扱うことも可能である．学習過程で正しい評価を得られていないサンプルは複雑な局面であると考えることも可能であり，従来提案された高度な特徴生成手法を組み合わせることは性能を大きく改善する可能性がある．

その他の課題としては，囲碁など他のゲームにおける提案手法の有効性の検証があげられる．本実験でも，対象とした学習手法によって提案手法の効果の大きさには大きな違いがあった．現在のゲームプログラミングの分野では，将棋に限らず，多くのゲームで様々な機械学習の手法が取り入れられており，提案手法がどのようなゲームのどのような学習手法を用いた場合に有効かを検証することは大きな意義があると考えられる．

7. おわりに

本論文では，特徴の生成と機械学習による重み付けを組み合わせた手法を提案した．具体的には，機械学習の反復計算の途中で，正しい評価を得られていないサンプルから新たな特徴を生成し，重みの調整と特徴の生成を同時に行うことで，有効な特徴セットを作成した．

提案手法を現在ゲームプログラミングで用いられている各種学習手法に適用した結果，指し手の一致率および対局実験において従来手法を上回り，提案手法の有効性を示した．今回の実験では，特にモンテカルロ木探索中の指し手の予測で提案手法が高い効果を示した．これはモンテカルロ木探索では長手数のシミュレーションを行うため，一致率・予測率の差が小さい場合にも結果に影響を及ぼしやすいことなどが理由として考えられる．

また，実現確率探索における logistic 回帰による指し手の予測，及び Boannza の手法による評価関数の学習における実験では，提案手法を L1 正則化による特徴の選択と組み合わせることにより，特徴数を抑えつつも有効な特徴を生成できることを確認した．

現在，ゲームプログラミングは，評価関数の作成や探索深さの調整など，従来人間がゲームの専門知識を利用して調整を行っていた多くの部分で機械学習が可能となっており，特徴の生成はゲームプログラミングにおいて人間の知識が必要な唯一の要素となりつつある．この部分を自動調整可能にすることは大きな課題であると同時にゲームプログラミングにおいて最も進歩する余地の大きい部分ともいえ，今後の課題に上げたような点を解決することにより，ゲームプログラミングのさらなる性能向上に貢献したいと考えている．

参考文献

1) 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，pp.78–83 (2006).
2) Tsuruoka, Y., Yokoyama, D. and Chikayama, T.: Game-Tree Search Algorithm Based On Realization Probability, *ICGA Journal*, Vol. 25, No.3, pp.145–152 (2002).
3) Coulom, R.: Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search, *In 5th International Conference on Computer and Games*, Turin, Italy, pp.72–83 (2006).
4) Coulom, R.: Computing Elo Ratings of Move Patterns in the Game of Go, *ICGA Journal*, Vol.30, No.4, pp.198–208 (2007).
5) Gelly, S., Wang, Y., Munos, R. and Teytaud, O.: Modification of UCT with Patterns in Monte-Carlo Go, Technical Report 6062, INRIA, France (2006).
6) 矢野友貴，三輪誠，横山大作，近山隆：ゲーム構成要素を組み合わせた特徴の最適化，ゲームプログラミングワークショップ 2010 論文集，pp.15–22 (2010).
7) 末廣大貴，畑埜晃平，坂内英夫，瀧本英二，竹田正幸：カーネル法を用いたコンピュータ将棋の評価関数の学習，ゲームプログラミングワークショップ 2010 論文集，pp.23–27 (2010).
8) Kaneko, T., Yamaguchi, K. and Kawai, S.: Automatic Feature Construction and Optimization for General Game Player, *The 6th Game Programming Workshop, IPSJ Symposium Series 2001*, Vol.14, pp.25–32 (2001).
9) Guyon, I. and Elisseeff, A.: An introduction to variable and feature selection, *J. Mach. Learn. Res.*, Vol.3, pp.1157–1182 (2003).
10) Ng, A. Y.: Feature selection, L1 vs. L2 regularization, and rotational invariance, *In Proceedings of the twenty-first International Conference on Machine Learning(ICML '04)*, New York, NY, USA, ACM, pp.78– (2004).
11) 保木邦仁：Bonanza – The Computer Shogi Program. http://www.geocities.jp/bonanza_shogi/.
12) 竹内聖悟，金子知適，山口和紀：将棋における，評価関数を用いたモンテカルロ木探索，ゲームプログラミングワークショップ 2010 論文集，pp. 86–89 (2010).
13) 佐藤佳州，高橋大介：モンテカルロ木探索によるコンピュータ将棋，情報処理学会論文誌，Vol.50, No.11, pp.2740–2751 (2009).
14) 鶴岡慶雅：最近のコンピュータ将棋の技術背景と激指，情報処理， Vol.49, No.8, pp.982–986 (2008).
15) Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R. and Lin, C.-J.: LIBLINEAR: A Library for Large Linear Classification, *Journal of Machine Learning Research*, Vol.9, pp.1871–1874 (2008).
16) Kocsis, L. and Szepesvari, C.: Bandit based Monte-Carlo Planning, *In Proceedings of 15th European Conference on Machine Learning(ECML)*, Springer, pp.282–293 (2006).