モンテカルロ碁における統計的手法による特徴の学習

門脇聡広, 村松正和
電気通信大学情報工学専攻

概 要

モンテカルロ碁におけるシミュレーションに囲碁の特徴や知識を用いることで, シミュレーションの精度が向上し, プログラムの強さに影響を与えることが知られている. そこで本稿では, シミュレーションの精度向上を目指して, 囲碁における特徴を学習する新たな手法を提案する. 松井ら [2] と同様の確率モデルを用いて, 最尤法を用いた目的関数の設計を行い, 提案する最適化手法を用いてパラメタ推定を行った. 得られたパラメタの評価を行い, 提案する手法の有効性を確認することができたが, 既存の手法に比べてパラメタの精度は劣る結果であった.

Using the maximum likelihood method in learning patterns for Monte Carlo Go

Satohiro KADOWAKI, Masakazu MURAMATSU
Department of Computer Science, the University of Electro-Communications

Abstarct

It is well-known that using simple patterns or knowledges of Go in the simulation of Monte Carlo Go enhances the strength of the program. Choosing appropriate patterns or knowledges is important in developing strong computer Go programs. This paper presents a new approach of learning such patterns and knowledge. Specifically, we use the same model as Matsui et al. proposed, and estimate the parameter by using maximum likelihood method. Although our method has the statistical meaning, we could not prove the superiority of our method against the existing methods by our preliminary numerical experiments.

1 はじめに

モンテカルロ碁において, ランダム・シミュレーションの精度がプログラムの強弱に大きな影響を与えることが知られている. Crazy Stone の作者である Rémi Coulom は, Elo Rateing の概念と少数化-最大化アルゴリズムを用いた学習手法 [1] を用いてシミュレーションの精度向上に成功した. 松井らは, ランダム・シミュレーションにおいて用いる各種特徴を確率モデルで表し, 誤差関数の最小化を用いた棋譜からの学習を行った [2].

本研究では, 松井らと同様の確率モデルを, より自然な統計的手法である最尤法を用いてパラメタ推定を行うことにより, 良い特徴を抽出することを試みた. そのパラメタの精度を測るために, テストセットを用いて検証を行った. またパラメタを用いた対局実験を行い, そのパラメタの有効性を確認した.

2 目的関数の設計と最尤法

ある局面 q が与えられた時, その局面でのある合法手 i_q に対する評価関数 $\gamma_q(i_q)$ を以下のように定義する [1].

$$\gamma_q(i_q) = \prod_{k=1}^{K} l_k(i_q).$$

ただし $l_k(i_q)$ とは局面 q での特徴 k の特徴関数であり, 以下のように定義される.

$$l_k(i_q) = \begin{cases} x_k & (\text{特徴 } k \text{ がある}) \\ 1 & (\text{特徴 } k \text{ がない}). \end{cases}$$

ここで $\boldsymbol{x} = (x_1, \cdots, x_K)^T > \boldsymbol{0}$ は K 次元のパラメタであり, その成分 x_k が大きいほど特徴 k が重要であることを表す. 以上の定義から, ある局面 q におけるある合法手 i_q が打たれる確率 $p_q(i_q)$ を以下のように定義する [2].

$$p_q(i_q) = \frac{\gamma_q(i_q)}{\sum_{j=0}^{M} \gamma_q(i_{q,j})}.$$

このような確率モデルのもとで, パラメタ $\boldsymbol{x}$ を推定することを考える. プロ棋士の棋譜における局面 q において打たれた手を i_q^* とし, これが大量に観測されていると考える. このような場合, 統計学においては, 最尤法によるパラメタ推定の考え方が自然とされ, 広く用いられている. 本稿ではこれを適用することを考える. 上記の確率モデルに対する尤度最大化問題は, 次のように定式化される.

$$\begin{cases} \max & \prod_{q=1}^{S} p_q(i_q^*) \\ \text{s.t.} & \boldsymbol{x} > \boldsymbol{0} \end{cases}$$

ここで標本である局面の数は S である. この最適化問題は以下の等価な問題に変形できる.

$$\begin{cases} \min & f(\boldsymbol{x}) = -\sum_{q=1}^{S} \log p_q(i_q^*) \\ \text{s.t.} & \boldsymbol{x} > \boldsymbol{0} \end{cases} \tag{1}$$

(1) を最適化することによって, 最適なパラメタを得ることを目標にする.

3 目的関数の微分

(1) を最適化する際には, 目的関数 $f(\boldsymbol{x})$ を微分して降下方向を求めなければならない. 以下で $f(\boldsymbol{x})$ の偏微分を計算する.

$$\begin{aligned} (\nabla f(\boldsymbol{x}))_k &= -\frac{\partial}{\partial x_k} \sum_{q=1}^{S} \log p_q(i_q^*) \\ &= -\sum_{q=1}^{S} \frac{1}{p_q(i_q^*)} \cdot \frac{\partial}{\partial x_k} p_q(i_q^*). \end{aligned}$$

ここで, $\frac{\partial}{\partial x_k} p_q(i_q^*)$ は以下のように計算される.

$$\begin{aligned} \frac{\partial}{\partial x_k} p_q(i_q^*) &= \frac{\partial}{\partial x_k} \frac{\gamma_q(i_q^*)}{\sum_{j=1}^{M} \gamma_q(i_{q,j})} \\ &= \frac{-(\frac{\partial}{\partial x_k} \sum_{j=1}^{M} \gamma_q(i_{q,j})) \cdot \gamma_q(i_q^*)}{(\sum_{j=1}^{M} \gamma_q(i_{q,j}))^2} \\ &\quad + \frac{\frac{\partial \gamma_q(i_q^*)}{\partial x_k}}{\sum_{j=1}^{M} \gamma_q(i_{q,j})}. \end{aligned}$$

さらに, $\frac{\partial \gamma_q(i_q)}{\partial x_k}$ は以下のように計算される.

$$\frac{\partial \gamma_q(i_q)}{\partial x_k} = \begin{cases} \prod_{s \neq k}^{K} l_{s,q}(i_q) & (\text{局面 } q \text{ において合法手 } i_q \text{が特徴 } k \text{ を持つ}) \\ 0 & (\text{局面 } q \text{ において合法手 } i_q \text{が特徴 } k \text{ を持たない}) \end{cases}$$

以上のように求められる $\nabla f(\boldsymbol{x})$ を用いて最適化を行う.

4 内点法的共役勾配法

上記の最適化問題は, 統計的観点からは自然なものであるが, 計算を行う際にはいくつか問題点がある. 一つは, 局面 q の数 S および特徴の数 K が非常に多いことにより, ヘッセ行列の計算は非常に困難なことである. もう一つは非負条件が存在するため, 制約付き最適化問題となっていることである. 制約付き最適化問題は, 一般に制約無しのものよりも解くのが困難である.

そこで, $\mu > 0$ を適当な実数として, (1) の問題に対して以下のようなバリア問題を考える.

$$\begin{cases} \min & g_\mu(\boldsymbol{x}) \stackrel{\text{def}}{=} f(\boldsymbol{x}) - \mu \sum_{j=1}^{K} \log x_j \\ \text{s.t.} & \boldsymbol{x} > \boldsymbol{0} \end{cases}$$

このバリア問題を, 図 1 のように解いていく.

入力: 目的関数 f, ∇f, 初期点 $\boldsymbol{x}^1$,
初期探索方向 $\boldsymbol{d}^0 = \boldsymbol{0}$,
初期バリアパラメタ $\mu_1 > 0$
Step0: $k = 1$ とする.
Step1: $\boldsymbol{x}^k$ が最適解に十分近ければ終了.
Step2: $\nabla g_{\mu_k}(\boldsymbol{x}^k)$ を $\nabla f(\boldsymbol{x}^k)$ を用いて計算する.
Step3: $\boldsymbol{d}^k = -\nabla g_{\mu_k}(\boldsymbol{x}^k) + \frac{\|\nabla g_k(\boldsymbol{x}^k)\|^2}{\|\nabla g_{\mu_{k-1}}(\boldsymbol{x}^{k-1})\|^2} \boldsymbol{d}^{k-1}$ を計算する.
Step4: $\boldsymbol{x} = \boldsymbol{x}^k$, $\boldsymbol{d} = \boldsymbol{d}^k$ として直線探索を行い, 解を α^k とする.
Step5: $\boldsymbol{x}^{k+1} \leftarrow \boldsymbol{x}^k + \alpha^k \boldsymbol{d}^k$.
Step6: $\mu^{k+1} < \mu^k$ を定める.
Step7: $k \leftarrow k + 1$ として **Step1** へ.

図 1 内点法的共役勾配法のアルゴリズム

このアルゴリズムは, 基本的には共役勾配法の探索方向を用いているが, 同時に徐々にバリアパラメタを減少させるという内点法的な考えも利用し

ているので, ここでは内点法的共役勾配法と呼ぶことにする. この方法は内点法と違い, ヘッセ行列を必要としないので, 計算量的に有利となっている.

5 実験

5.1 実験内容

特徴ベクトルは対称性を考慮した 3×3 パターン, アタリ (抜き, 逃げとなる位置), アタリにする手, 盤端からの距離, 1 手前の手から斜めの位置, 隣の位置, 2 手前の手から斜めの位置, 隣の位置, 飛びの位置, 桂馬の位置で構成した.

パラメタの精度の指標として, 一致率を用いる. 一致率は全サンプル中に, プロの打った手の評価関数の値が上位 N 手以内に含まれる確率である. またフィルタリングの性能についての指標として, 占有率を使う. 占有率とは全サンプル中で, 上位 N 手の評価関数の値の和が総スコアに比べてどの程度の割合かを示す.

特徴ベクトルの初期点は, 全て 1.0 として最適化を行った. テストセットには, 最適化に使用したプロの棋譜とは別のプロの棋譜から 100,000 局面を抽出したものを用いた. このテストセットを用いて, 一致率, 占有率を測定した.

また, 通常の最適化の後にフィルタリングという操作を行った. フィルタリングとは最適化終了後に, ある閾値を下回るような価値の低い特徴を, 特徴無しとして再び最適化を行うことで, 有用な特徴を選び出し, そのパラメタを決定することである. ただしシミュレーションなどにフィルタリング後のパラメタを使用する際には, 除かれた特徴には非常に小さな重みを与えることとした.

5.2 一致率

図 3 はプロの棋譜から 50,000 局面, 100,000 局面, 200,000 局面, 300,000 局面を抽出して最適化した後のパラメタを用いた一致率 50,000, 100,000, 200,000, 300,000 である.

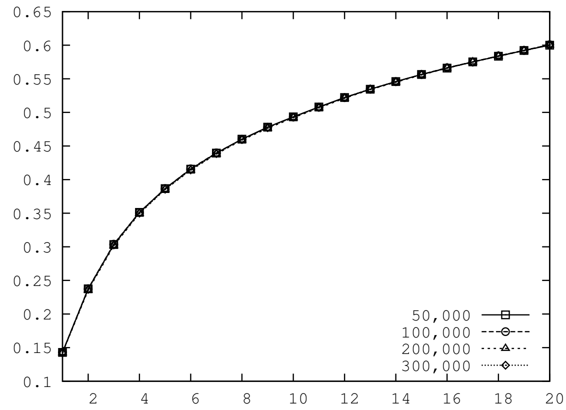

図 3 局面数に応じた一致率

この図から, 訓練集合の数を増やしても一致率の変化はごく僅かであり, 50,000 局面の最適化でも十分な効果が得られることが分かる. 図 4 は 300,000 局面で最適化を行った際のパラメタを用いた一致率 NotFiltering, フィルタリングを 1, 2, 3 回行った際の一致率 Filtering1, 2, 3 である.

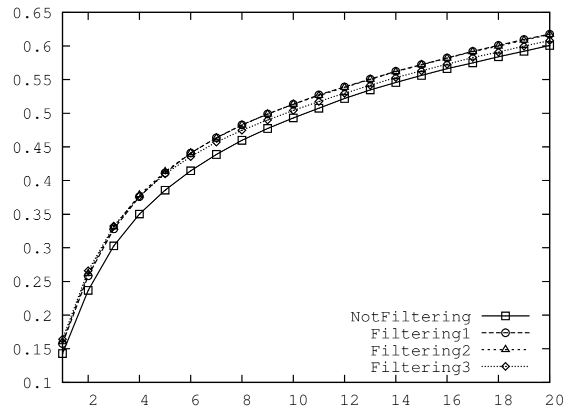

図 4 フィルタリング数に応じた一致率

この図から, フィルタリングを行うことで, フィルタリングをしないものより良い予測を与えていることが分かる.

5.3 占有率

図 5 は 300,000 局面で最適化を行ったパラメタを用いた占有率 Possession, フィルタリングを 1, 2, 3 回行った際の占有率 PossessionFilter1, 2, 3 である.

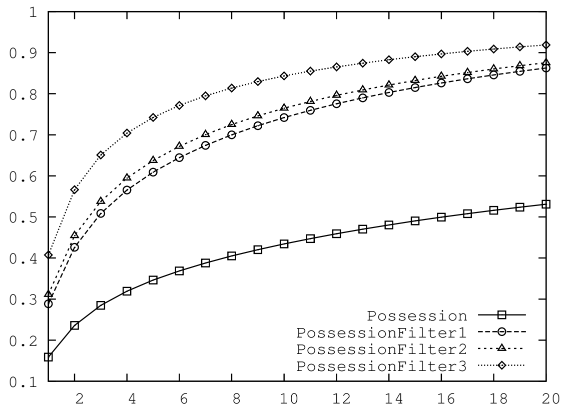

図 5 フィルタリング数に応じた占有率

この図から, フィルタリングを行えば行うほど占有率は上がり, フィルタリング回数を増やす毎に上位 20 位内の手から選択される確率が上がることが分かる.

5.4 対局実験

実験で用いるプログラムには UCT を実装し, 1 手毎に 10,000 回のシミュレーションを行った. シミュレーションは 3×3 パターン, アタリ, 1 手前の距離による重み付けを行った.

実験としては, 上記の特徴のパラメタを従来手法（約 1 万棋譜から現れたパターンと実際に打たれたパターンを計測して計算される頻度を基にした重み付け）と今回の提案手法との比較で行った. 19 路盤でコミは 6.5 目, 対局数は 200 である.

以下表 1 にその結果を載せる. フィルタリング数は, フィルタリングを行った回数毎のパラメタを用いたプログラムを表す. 上限, 下限はそれぞれ 95%信頼区間の上限, 下限を表す.

表 1 従来手法との対局結果

フィルタリング数	勝率	上限	下限
1 回	61.00 %	67.80 %	53.87 %
2 回	68.00 %	74.40 %	61.05 %
3 回	69.00 %	75.33 %	62.09 %

6 考察

一致率において, 既存の手法 [1] は上位 1 手で約 35%, 上位 20 手で約 85%を主張している. それに対して図 3, 4 より, 我々の一致率は既存の手法を上回ることが出来なかった. 特徴の数として, まだ少ないのが影響しているものと思われる. また, 最適化に用いる局面数にはあまり影響されないことが分かった. 最適化にかかる時間は, 局面数に大きく左右される. 局面数が少なくて済むということから, 計算時間の面で有利になりうることが考えられる.

また図 4 から, フィルタリングを行うことで有用な特徴を抽出し, その中で良いパラメタを与え得るということが分かる. しかしフィルタリングの効果は 1, 2 回でほぼ変わらず, 3 回で上位 5 手程度は他のフィルタリング数に比べ良い結果となっているが, それ以降は下回っている. フィルタリングの回数を単純に上げるだけでは, 上手くは行かないと思われる. しかし図 5 の結果から, フィルタリングの回数を上げると, 局面毎に計算される上位と下位の手とのスコアの差が大きくなり, シミュレーション中では上位の手が打たれやすくなると思われる. 一致率が大きく変わらない為, 表 1 からフィルタリングの回数が多い程, より良いシミュレーションを行えたと思われる.

7 おわりに

本研究では, 特徴ベクトルを求める新たな手法を提案した. 統計的に自然な考え方である最尤法が, 効果的なパラメタを与える可能性があることを確認した. しかし, 一致率では既存の手法に対して上回ることが出来なかった.

今回行った特徴の最適化に用いた特徴数は, 既存の手法の特徴数に比べてまだ少ない. 今後の課題として, 特徴の数を増やすことでどれだけ一致率が上がるのかを実験する必要がある.

参考文献

[1] Rémi Coulom.Computing Elo Ratings of Move Patterns in the Game of Go, In Computer Game Workshop, Amsterdam, The Netherlands(2007).

[2] 松井利樹, 野口陽来, 土井佑紀, 橋本剛. 囲碁における勾配法を用いた確率関数の学習, IPSJ SIG Technical Report, pp.33-40, 2009.