Vol.2018-GI-40 No.4
2018/6/29

手札提出時期によるコンピュータ大貧民プログラムの分類

但馬 康宏[1,a)]

概要：コンピュータ大貧民の研究においては，提出手の特徴によりプログラムを分類する研究がなされている．本研究では，1 ゲームの中でカード提出が行われた時期に注目し分類を試みる．大貧民においては，1 ゲームは 1 デッキのカードを使い切ることにより終了するので，各カードが提出された時期を要素値とするベクトルで 1 ゲームを表現できる．このベクトルをオードエンコーダおよびフィードフォワードネットにより学習させることにより，ゲームの特徴が得られるかを実験を通じて検討する．

Classification of Daihinmin programs based on move timing of hands

Tajima Yasuhiro[1,a)]

1. はじめに

コンピュータ大貧民においては，モンテカルロ法を用いたアルゴリズムの研究 [1] およびシミュレーション効率向上のための相手モデルの検討 [2] やヒューリスティック戦略の検討 [3] などアルゴリズムの直接的な発展の他に，空場におけるパスの効果 [4] などゲームの分析にかかる研究も盛んに行われている．その中でもプログラムの分類に関する研究は，n-gram 統計を用いたもの [5] ，決定木を用いたもの [6] などが行われている．

分類基準として提出手の n-gram 統計を用いた研究 [5] では，提出手の 1-gram および 2-gram の出現率をベクトルとし，その類似度からプログラムのグルーピングを行うことにより良い分類が得られることが示されている．また，決定木を用いた分類の研究 [6] では，手札の状況や提出に要する時間などを選択変数として決定木を作成することによりプログラムの分類を行っている．

本研究では，1 ゲームにおいて各プログラムが提出した時期を要素値として持つベクトルを考え，1 ゲームにおける 1 プログラムの提出状況を 1 つのベクトルで表現することにより分類および特徴の抽出が行えるかを実験的に確かめる．分析には，オートエンコーダとフィードフォワードネットを用いて実験を行った．

[1] 岡山県立大学
111, Kuboki, Soja, Okayama 719–1197, Japan
[a)] tajima@cse.oka-pu.ac.jp

2. ゲームの表現

大貧民における 1 ゲームは，ジョーカー 1 枚を含む 1 デッキのカードをすべて使い切ることにより終了する．したがって，すべての提出は 1 ゲーム中の提出時期において全順序をなす．さらに，一度に複数枚の提出については，その提出の中の各カードについてランクやスートで順序付けることにより，すべてのカードは 1 ゲーム中の提出時期において全順序をなす．

以上より，1 ゲームにおいて最初に提出されたカードの値を 1/53 とし，最後に提出されたカードの値を 1 とし，その他のカードは提出時期の順序で値が増えるようにすることで 53 次元のベクトルで 1 ゲームを表現することができる．複数枚を同時に提出する場合は，その提出で使われているカードの順位の平均値に対応する値をそれぞれのカードが持つものとする．さらに，ゲームの最後に大貧民のプレイヤーが提出できずに残した手札は，最後に提出されたカードとして扱う．このように 1 ゲームを表現したベクトルを以後ゲームベクトルと呼ぶ．

次に，あるプレイヤーに注目した 1 ゲームの表現を考える．ゲームベクトルのうち，注目したプレイヤーの初期手札の部分だけ値を残し，初期手札でないカードの部分，すなわち注目したプレイヤー以外のプレイヤーが持つカードの部分をすべて 0 としたベクトルをその注目プレイヤーの

提出ベクトルと呼ぶ．同様にゲームベクトルにおいて，注目したプレイヤーの初期手札のみを 0 としたベクトルを敵ベクトルと呼ぶ．提出ベクトルと敵ベクトルを並べて 106 次元のベクトルとしたものを注目したプレイヤーのプレイヤーベクトルと呼ぶ．

本研究では，すべてのプレイヤーについてプレイヤーベクトルを作成することによりゲームの分析を行う．

3. オートエンコーダによる分析

まず始めに，すべてのプレイヤーに関するプレイヤーベクトルをオートエンコーダを用いて学習することにより分析を行う．オートエンコーダは入力層と同じ数の出力層を持ったネットワーク [7] であり，学習は入出力層を同じデータとすることで行われる．結果的にこのネットワークの中間層に学習データの特徴が反映されることが期待される．オートエンコーダの学習に必要なデータは入力ベクトルだけであり，本研究におけるモデリングではこれは，プレイヤーベクトルのみということになる．複数のプログラムに関するプレイヤーベクトルを準備し，オートエンコーダで特徴を見つけることができれば，対戦ログのみからプログラムの特徴を抽出することができる．これは，過去の研究において分析のために利用した特徴量を自動で見つけることに相当する．

3.1 学習データとネットワーク構造

実験対象としたコンピュータ大貧民プログラムは，default, kou, beersong, snowl, fumiya の 5 つであり，これらの対戦ログからプレイヤーベクトルを作成した．対戦は 1000 ゲーム行い，順位のリセット，席順の決定などはコンピュータ大貧民大会と同条件とした．プレイヤーベクトルの構成要素である提出ベクトルと敵ベクトルはそれぞれコンピュータ大貧民大会通信プロトコルで用いられている 5 行 15 列の配列をそのまま使えるよう 75 の入力素子とした．したがって，プレイヤーベクトルの次元数は 150 である．

オートエンコーダは入力層，出力層の素子数がそれぞれ 150 であり，中間層は 1 層とした．中間層の素子数を変化させて実験を行った．実装は tensorflow を用いて行った．

3.2 実験結果

図 1, 2, 3 に中間素子数 2 とした場合の学習データに対する中間層出力値を示す．

学習終了時の誤差平均二乗和は 29.23 であり，学習回数はおよそ 20 万回であった．プレイヤーベクトルのサイズが 150 であり，初期状態での誤差は 80 前後であった．したがって，収束後も少ない誤差ではないが減少したといえる．この結果から，学習の収束は見られるものの，プログラムの特徴をつかむことはできなかった．

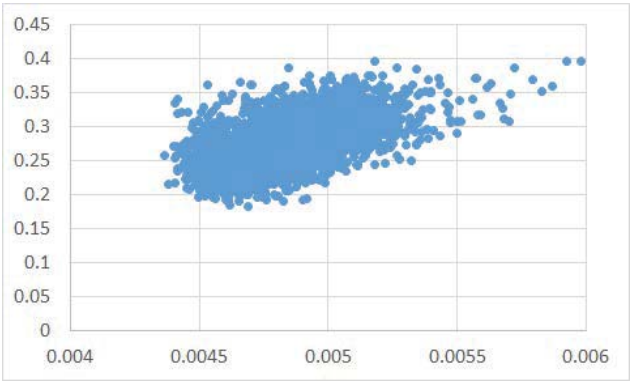

図 1 学習データに対する中間層の出力値

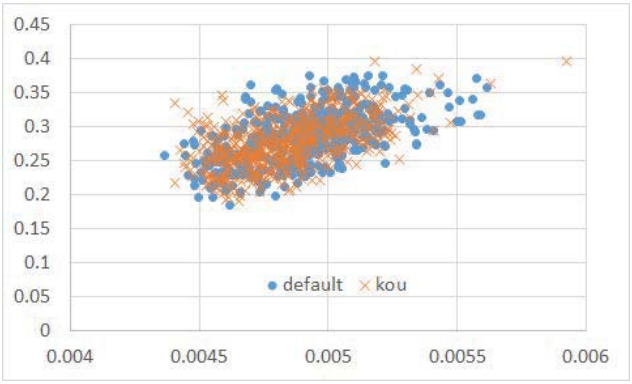

図 2 default, kou に対する中間層の出力値

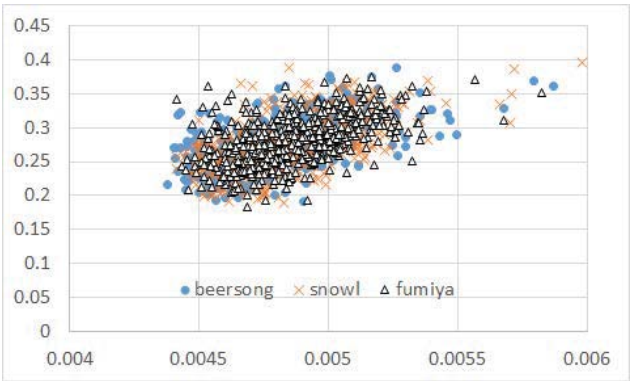

図 3 beersong, snowl, fumiya に対する中間層の出力値

図 4, 5, 6 にオープンデータに対する結果を示す．実験対象のデータは前述の対戦ログとは別に獲得した 500 ゲーム分である． この場合も学習データに対する結果と同じく，プログラムの特徴をとらえることはできなかった．誤差の平均二乗和は 5.879 であり，学習データよりも誤差は少なくできた．この結果からも，学習データに対する分解能は十分あるが，特徴をとらえることに失敗していることがわかる．

次に中間層の素子数を 3 とした場合に同様の実験を行った．図 7, 8 に中間層の出力値を示すが，素子数が 2 の場合と同様に学習は収束するが特徴をとらえることはできなかった． 中間素子数 3 の場合，学習収束後の誤差の平均二乗和は 29.26 であり，この場合も収束は十分であるが特

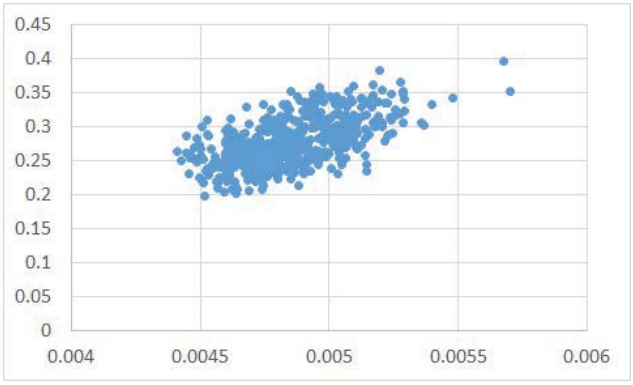

図 4 評価データに対する中間層の出力値

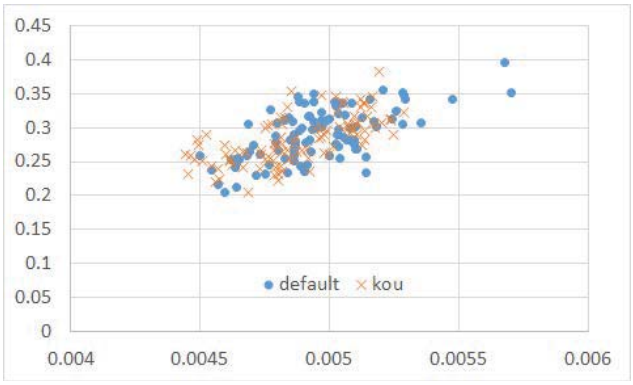

図 5 default, kou の評価データに対する出力値

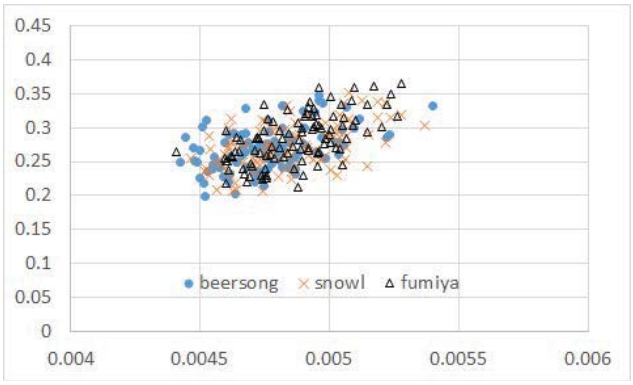

図 6 beersong, snowl, fumiya の評価データに対する出力値

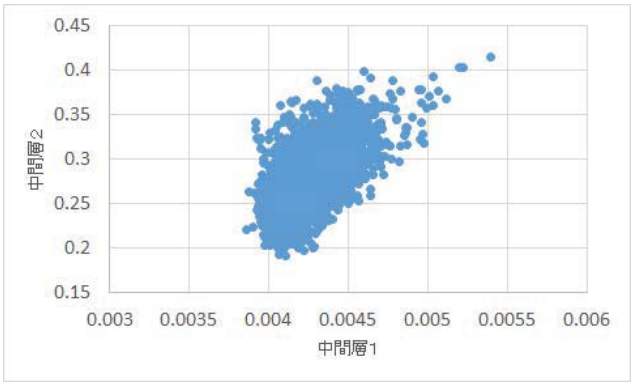

図 7 学習データに対する中間素子 1,2 の出力値

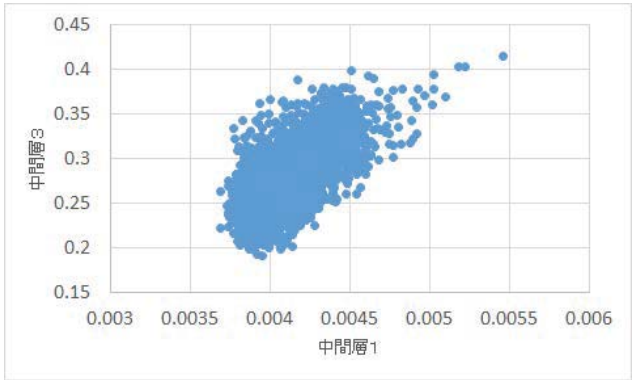

図 8 学習データに対する中間素子 1,3 の出力値

徴はとらえられていないことがわかる．

4. フィードフォワードネットによる分類

オートエンコーダーではプログラムの特徴をとらえることができなかったが，学習は収束していた．そこで，教師あり学習としてとらえ直しプレイヤーベクトルをフィードフォワードネットで学習させ，その性能をみる．

4.1 学習データとネットワーク構成

オートエンコーダにおける実験と同様に学習データとして 1000 ゲーム，評価データとして 500 ゲームの対戦ログを取り，各プログラムに対するプレイヤーベクトルを作成した．学習データ，評価データともに，ひとつのプログラムに関するプレイヤーベクトルは全体の 1/5 であり，残りは他のプログラムのプレイヤーベクトルである．フォードフォワードネットワークは，オートエンコーダの中間層が 1 層でも収束したことから，3 層とし中間層の素子数は対戦ログに出現するプログラム数と同数である 5 とした．

4.2 実験結果

図 9, 10 にそれぞれ学習データ中の正例に対する評価結果と負例に対する評価結果のグラフを示す．

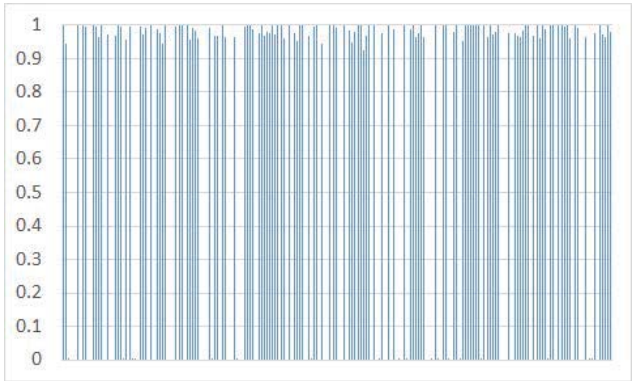

図 9 学習データ中の正例に対する評価結果 (横軸：サンプル番号)

正例に対する正解率は 0.6450 となり，負例に対する正解

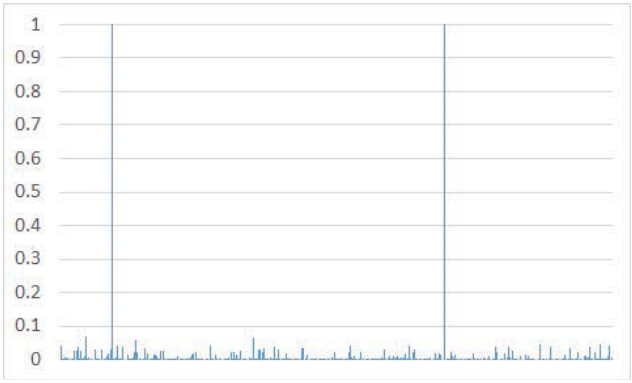

図 10 学習データ中の負例に対する評価結果 (横軸：サンプル番号)

率は 0.9975 となった．正例の割合は学習データ全体の 1/5 であるため，学習データ全体に対する正解率は，0.9270 となった．学習後の誤差の平均二乗和は 73.12 であり，オートエンコーダにおける誤差よりも大きな値となった．ここでも，学習データに対する収束は良くクローズドテストの結果も 9 割以上の正解率となった．

図 11, 12 にそれぞれ評価データに対する推定結果のグラフを示す．

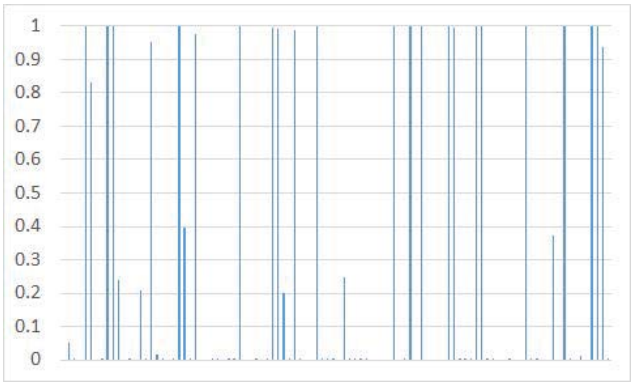

図 11 評価データ中の正例に対する結果 (横軸：サンプル番号)

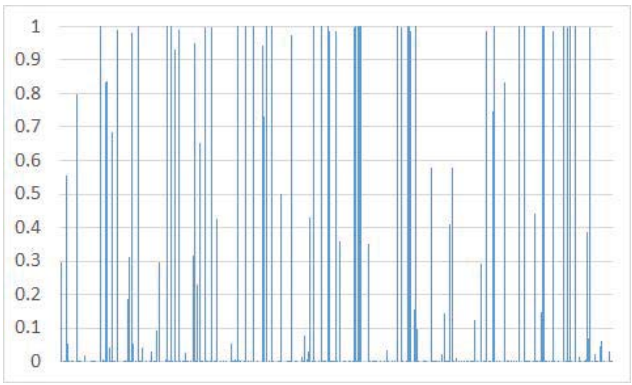

図 12 評価データ中の負例に対する結果 (横軸：サンプル番号)

正例に対する正解率は 0.2400 となり，負例に対する正解率は 0.8575 となった．評価データ全体に対する正解率は 0.7340 である．ここでも学習データに対する収束および正解率の高さに比べ，評価データでの性能が悪いことが示された．これは，ネットワークの分解能は十分であるがプログラムの判別という特徴が得られなかったことを示している．

5. おわりに

コンピュータ大貧民における 1 ゲームのカード提出時期に注目したベクトルを用いてオートエンコーダによる特徴抽出を試みた．その結果，ネットワークの学習は収束したが，特徴をとらえることはできなかった．さらに，フィードフォワードネットによるプログラムの分類を試みた．こちらも，学習データに対する性能に比べ評価データでの性能が低いことが示された．以上より特徴抽出には至らなかったが，プログラムの同定は 73% の正解率で判定することができた．今後の課題として，より複雑なネットワーク構成により特徴を抽出できる方式の考案が挙げられる．また，ベクトルの構成要素に過去の研究で効果の確認できている提出手の n-gram やゲームの状況を説明した変数を導入して性能向上があるかどうかの検証も重要な課題である．

参考文献

[1] 小沼啓, 西野哲朗：コンピュータ大貧民に対するモンテカルロ法の適用，情報処理学会研究報告 (GI)，vol.2011-GI-25, no.3, pp.1–4, 2011.
[2] 柳澤佑介，松崎公紀：大貧民における出現頻度と提出手役履歴を用いた相手手札推定，情報処理学会研究報告 (GI), vol.2015-GI-33, no.9, pp.1–6, 2015.
[3] 田頭幸三，但馬康宏：コンピュータ大貧民におけるヒューリスティック戦略の 実装と効果，情報処理学会論文誌, vol.57, no.11, pp.2403–2413, 2016.
[4] 大渡勝己，田中哲朗：大貧民の空場におけるパスの有効性の検証，情報処理学会研究報告 (GI), vol.2017-GI-37, no.11, pp.1–8, 2017.
[5] 綾部孝樹，大久保誠也，西野哲朗：大貧民プログラムの n-gram 統計による特徴抽出とクラスタ分析，情報処理学会研究報告 (MPS), vol.2013-MPS-93, no.2, pp.1–6, 2013.
[6] 小西正人，大久保誠也，若月光夫，西野哲朗：決定木を用いた大貧民プログラムの分析に関する研究，情報処理学会研究報告 (GI), vol.2016-GI-36, no.13, pp.1–8, 2016.
[7] Ian Goodfellow, Yoshua Bengio, Aaron Courville (岩澤有祐，鈴木雄大，中山浩太郎，松尾豊監訳)：深層学習，KADOKAWA, 2018.