JASLA Research Report
Vol.2017, No.1 (2017-5)

プログラミング行動の履歴に対する Deep Learning 分析

加藤利康*, 久保田純**, 卯木輝彦**, 梅澤健二**, 児玉靖司***

＊ 日本工業大学
** 株式会社フォトロン
*** 法政大学

Deep Learning Analysis on the History of Programming Behaviors

Toshiyasu Kato*, Jun Kubota**, Teruhiko Unoki**, Kenji Umezawa**, Yasushi Kodama***

＊ Nippon Institute of Technology
** Photron Limited
*** Hosei University

＊ katoto@nit.ac.jp

概要：本論文は、学習分析学会主催のハッカソンで行われた Deep Learning を用いた分析結果を報告する。このハッカソンの分析対象はプログラミング演習の授業における学生のプログラミング行動の履歴である。本研究は、プログラミング行動の特徴を推定することを目的として、コンパイルのエラー回数や成功回数などにデータマイニングを行ってきた。前年・翌年度分のデータを用意し、従来手法の結果を教師データとして Deep Learning でモデルを作成した。このモデルに翌年度分のデータを適用したところ、推定精度が 95%であった。従来手法に相当する推定が Deep Learning を用いても可能であることを示す。

Abstract: This paper reports analysis results using Deep Learning conducted at Hackathon hosted by the Japanese Society for Learning Analytics. This Hackathon's analysis subject is a history of student programming behaviors in the programming exercises. The aim of the authors is to infer the characteristics of programming behaviors. In this research, data mining has been performed on the number of compile errors and compilation successes. We prepared data for the previous year and the following year, and created the model with Deep Learning as the teacher data of the result of the conventional method. When the data for the following fiscal year was applied to this model, the estimation accuracy was 95%. We show that the estimation equivalent to the conventional method is also possible using Deep Learning.

キーワード： ハッカソン、ディープラーニング、プログラミング演習、プログラミング行動、対面授業
Keywords: Hackathon, Deep Learning, Programming Exercises, Programming Behaviors, Face-to-Face classes

1. はじめに

本論文は、2017 年 2 月 25 日から 26 日に行われた学習分析学会主催のハッカソンにおいて、Deep Learning を用いた分析結果の一例を報告する。今回のハッカソンの参加者が大学生をはじめとした Deep Learning の初学者であったため、ハッカソンの目的を Deep Learning の学習とした。

ハッカソンのテーマは、プログラミング行動の履歴に対する Deep Learning 分析である。具体的には、高等教育機関において実施されているプログラミング教育の演習時におけるコンパイル回数やコンパイルエラーの内容などから分析を試みる。筆者らは従来研究（加藤, 2016）として、これらのプログラミング行動のデータをクラスタリングして特徴別に分類した。ハッカソンにおいては、この分類結果を教師データとして提供し、別の年度の特徴を Deep Learning によって推定する。

本発表は、Deep Learning による解析例を報告する。はじめに、従来手法の結果を教師データとして Deep Learning でモデルを作成する。つぎに、別年度のデータを適用して推定精度を確認し、精度が高まるようにパラメータを調整する。調整の結果、筆者らは推定精度を95%まで向上させた。従来手法に相当する推定が Deep Learning を用いても可能であることを示す。

2. ハッカソンのテーマ

ここでは、Deep Learning 分析の教師データとして用いる従来研究とハッカソンの課題について述べる。

2.1 分析の教師データとして用いる従来研究

ハッカソンの分析で用いるデータは、日本工業大学の Java プログラミング科目において実施された演習時のプログラミング行動の履歴である。ここでのプログラム行動の履歴とは、著者らが開発したプログラミング演習支援システム(加藤, 2014)で取得するログのことである。このログのうち、演習課題を解答するにあたり、プログラミング行動が記録されるコンパイルログを使用した。コンパイルログの内容は以下のとおりである。

1. 平均課題解決時間
2. コンパイル回数
3. コンパイルエラー回数
4. 同一エラー回数
5. プログラム実行回数
6. 平均コンパイル時間間隔
7. 平均実行時間間隔

これらのデータを用いて、著者らは学習支援経験の少ないTA(Teaching Assistant)のためにプログラミング行動の特徴を提示することを目的とした研究を行ってきた。従来研究では行動の特徴を分類するためにクラスタ分析を行った。

クラスタ分析の結果を図1に示す。図1は階層的クラスタリングの ward 法(M. R. Anderberg, 1973)を用いて作成したデンドログラムである。この図1から大きく4つに分類できることがわかる。4つの分類別の集計を表1に示す。表1の特徴はつぎのとおりである。

クラスタ1:課題解決時間が短い。学生は、エラー内容を理解しプログラミングしている。

クラスタ 2:課題解決時間が短い。同一エラー回数が多いため、学生は、エラー内容をあまり理解せずにプログラミングしている。

クラスタ 3:課題解決時間が長い。平均実行時間間隔が長いため、学生は、考えているがエラー内容を理解出来ていない可能性がある。

クラスタ 4:課題解決時間が長い。平均実行時間間隔が短いため、学生は、ひたすらコンパイルと実行を繰り返している。この理由は、筆者らの演習支援システムには、教員の用意した模範解答と同一の解答でなければ提出できない仕組みがあるためである。

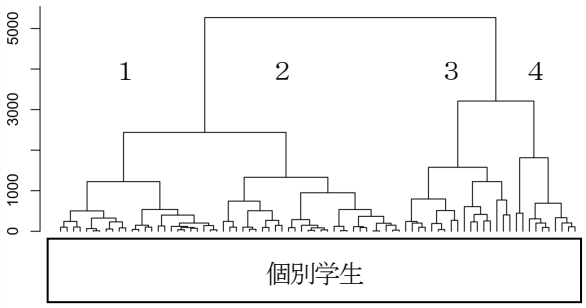

図1 ward 法によるクラスタ分析

表1 クラスタ別の集計

	1	2	3	4	5	6	7	人数
1	528	7	1	1	5	209	267	25
2	812	6	3	3	3	379	606	28
3	1405	7	4	3	3	590	1160	17
4	1506	18	5	4	13	228	337	10

行タイトルの数字は、コンパイルログの内容を表す
列タイトルの数字は、クラスタ番号を表す

Deep Learning 分析において教師あり学習を用いる場合、教師データが必要となるため、今回はこのクラスタを教師データとして、学生の特徴が推定できるかを試みる。

2.2 ハッカソンの課題

ハッカソンの目的がDeep Learningの学習であるため、ハッカソンの課題は、推定精度を競うものとした。具体的

な課題の内容はつぎのとおりである。

- 2014 年度と 2015 年度のプログラミング行動の履歴データ、および従来研究による 2014 年度の教師データ(ラベル)から Deep Learning を用いてモデルを作成し、2015 年度のラベルを推定する。
- 推定したラベルを主催者に提出してスコアを算出してもらい、そのスコアを競うものとする。

ハッカソンの主な作業の流れはつぎのとおりである。

1. データファイルの読み込み
2. 簡単な統計解析(データの素性を知る)
   Python の統計解析パッケージや表計算ソフトの利用など、やり方は自由とする
3. 特徴量の作成
4. Tensor Flow(Google)を用いて Deep Neural Networks (DNN)を構成
5. 2014 年度のラベルで交差検定しながら学習
6. DNN 構造やパラメータの調整
7. 2015 年度のデータに対して DNN を適用して 2015 年度のラベルを推定
8. 推定ラベルを保存して提出し、スコアを確認

3. 解析例

ここでは、実際に解析例を3つ紹介し、その考察を述べる。解析例の手順はつぎのとおりである。

1. コンパイルログの概要をつかむ
   2014 年度と 2015 年度は同じ学生ではないため、傾向が異なると考えられる。また、コンパイルログには、ラベルに存在しない学生がいる。この学生は、数回の授業しか受講しておらず最後まで履修しなかった者であると考えられる。そのため、分析時に課題着手率が明らかに少ない学生を除外する必要がある。
2. データのヒストグラムを作成し、課題解決時間の分布から、傾向をつかむ
   課題の所要時間は全体的に 2015 年度のほうが多いが、傾向としては似ているため、正規化することで使えるようになる。
3. 特徴量を決める
   データファイルの特徴量(2.1 節のコンパイルログ参照)を正規化し、分析しながら取捨選択していく。また、2.1 節のコンパイルログの内容に加えて、平均コンパイル成功回数と課題着手率をコンパイルログより抽出した。
4. DNN の構成とパラメータを決める
   入力層、出力層、隠れ層、活性化関数、損失関数、オプティマイザ、学習率を決める。
5. DNN のモデルを作成し、推定精度を求める
6. 推定精度を確認し、DNN の構成とパラメータを調整する
7. 推定精度が向上するように手順 5、6 を繰り返す

3.1 解析例の紹介

隠れ層が2層と3層の場合、および精度を維持しつつ入力層のユニットを減らした例を紹介する。なお、DNN の構成として、出力層はラベルが4なので4ユニット固定である。また、活性化関数は、事前分析において良い結果を得た sigmoid、オプティマイザは Adam、損失関数は softmax cross-entropy をそれぞれ固定した。これらが良い結果となったのは、画像の分析とは異なりデータ数が少ないためである。

3.1.1 隠れ層が２層の場合

ニューラルネットワークの構成として、入力層は7ユニットとした。隠れ層は2層に固定し、そのユニット数を1層目 64 ユニット、2 層目 32 ユニットにした。その結果、ラベル推定精度は約88%であった。損失を図2、精度を図3に示す。

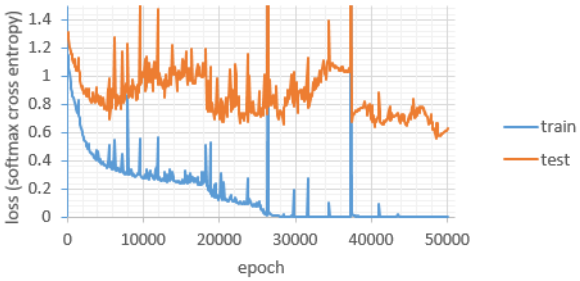

図２ 隠れ層が２層の損失

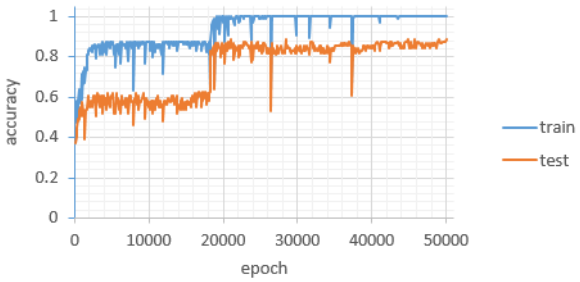

図 3　隠れ層が 2 層の精度

3.1.2　隠れ層が 3 層の場合

入力層は7ユニットとした。隠れ層は 3 層に固定し、そのユニット数を1層目 128 ユニット、2 層目 64 ユニット、3 層目 32 ユニットにした。その結果、ラベル推定精度は約 95%であった。損失と精度を図 4 に示す。

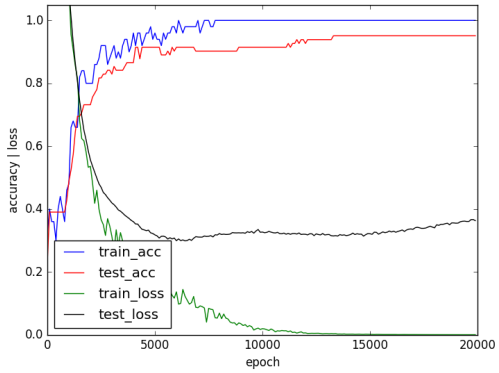

図 4　隠れ層が 3 層の損失と精度

3.1.3　隠れ層が 3 層で入力層が 4 ユニット

入力層は 4 ユニットとした。4 ユニットの内訳は、平均課題解決時間、コンパイルエラー回数、平均コンパイル成功回数、課題着手率である。隠れ層の構成は 3.1.2 節と同一とした。その結果、ラベル推定精度は約 95%であった。損失と精度を図 5 に示す。

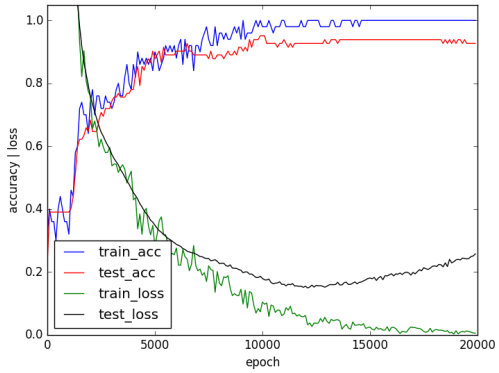

図 5　隠れ層が 3 層で入力層が 4 ユニット

3.2　解析例の考察

隠れ層を増やすことで推定精度が向上したのは、複雑なニューラルネットが構成され、学習が進んだと考えられる。しかしながら、どこまで隠れ層を増やせばいいのか、ユニット数はいくつにすれば良いのかはオーバーフィッティングの問題もあるため、試行錯誤が必要である。

入力層のユニット数を削減しても推定精度が維持できたのは、相関の高いユニットを削除したためである。相関の高い特徴量がある場合は、特徴量数を減らしてもある程度の層数、ユニット数があれば推定精度を上げることができる。これは、ニューラルネットワーク内で相当する特徴量が生成されていると考えられる。

隠れ層のユニット数を多くしすぎるとオーバーフィッティングしてしまう傾向があった。ユニット数が多くなるということは、ニューラルネットワークの表現力が高まることになり、入力データに整合する学習ができる。これでは未知のデータに対応できない。学習時にすぐに Loss が減少して収束することがあるが、それはオーバーフィッティングしているので交差検定が重要である。

4. おわりに

Deep Learning によりクラスタリングを用いた従来手法に相当する推定が可能であることを示した。分析前のデータ解析や特徴量抽出が重要であり、Deep Learning を使う場合においても同様である。予測に対する精度向上のためにはドロップアウトやノイズ付加も考える必要がある。

謝辞　本研究は独立行政法人日本学術振興会 科学研究費補助金（課題番号：26240008，15K01094）の助成を受けたものである。

参考文献

加藤 利康 & 石川 孝 (2014). プログラミング演習のための授業支援システムにおける学習状況把握機能の実現, 情報処理学会論文誌, 55(8), 1918-1930.

加藤 利康 (2016). プログラミング演習における TA のためのプログラミングの仕方分析, 日本 e-Learning 学会誌, 16, 101-106.

Google. Tensor Flow. https://www.tensorflow.org

M. R. Anderberg (1973). Cluster Analysis for Applications, Academic Press.