Vol.2020-CLE-30 No.16
2020/3/9

学習分析の効率化に向けたオープンソースライブラリ「OpenLA」の開発

村田 隆介[1,a)]　島田 敬士[2,b)]　峰松 翼[2,c)]　谷口 倫一郎[2,d)]

概要：近年注目されているラーニング・アナリティクス (LA) 分野の研究では，学習管理システムや電子教材などの e-learning システムから収集された教育ビッグデータが用いられる．収集されたデータ（ログ）は通常，そのままの形で分析に用いられることはなく，適切な形に集計・加工する必要がある．現在，そのようなデータ整形は研究者それぞれが行っており，そのために多くの時間が費やされている．そこで本研究では LA の効率化に向けて，データ整形のためのオープンソースライブラリ「OpenLA」を開発する．また，OpenLA を適用した事例として，電子教材から収集された学習行動に基づく学生の成績予測を紹介する．

1. はじめに

情報化社会の進展に伴い，教育分野においても教育の情報化が加速している．なかでも，デジタル学習環境を積極的に活用して，学習の成果のみでなくその過程に関するデータをきめ細かく収集し，データ解析に基づく学習や教育の改善を目的とした取り組みは，ラーニングアナリティクス（Learning Analytics，LA，学習分析）と呼ばれ，注目が集まっている．学習の過程を記録するシステムには，学習管理システムやポートフォリオシステムなどが一般的であるが，近年ではタブレット端末やノート PC などの普及によりデジタル教科書システムを活用して学習活動を記録する事例も増えている．

デジタル教科書システムには，出版関連会社が開発したシステムや大学生協が開発したシステムなど様々なシステムが開発されてきているが，基本的には，学習者が教材にアクセスした時刻や，教材内の各ページに遷移した時刻，各ページに記録されたマーカー（ハイライト）やメモ，ブックマークなどの情報が記録されている．実際に京都大学と九州大学でオープンソースとして開発が進められているデジタル教科書システム BookRoll [1], [2] においても同様の学習活動が記録されている．このような学習活動の記録を利用した研究成果として，学習行動の分析 [7] や学習行動パターンの発見 [5]，成績予測 [3], [4]，注意が必要な学生の発見 [6] などが報告されている．これらの研究に取り組むためには，デジタル教科書システムに記録されたログデータから各学習者がどの教材をどの程度の時間をかけて閲覧したのか，どのページにどのくらいの量の注目が集まっていたのかなどの情報に変換・加工・整形する必要があり，これらの事前処理部については処理が共通する点が多いにも関わらず，各研究者が独自にプログラムコードを開発しているのが実情である．

そこで本研究では，事前処理でしばしば利用される計算処理の冗長な開発を軽減し，より先進的なラーニングアナリティクス技術開発を支援するためのオープンソースライブラリ「OpenLA: An Open-Source Library for e-Book Log Analytics」を開発する．開発したライブラリは，LAK Data Challenge Workshop[*1]で提供されているデジタル教科書の閲覧ログの解析に利用可能である．本稿では，ライブラリが提供する API 関数群の紹介ならびに実際の利用例について述べる．

2. OpenLA ライブラリ

OpenLA ライブラリは，LAK Data Challenge Workshop で公開されているデジタル教科書のイベントログを分析することを目的に設計・開発を進めている．公開データは，BookRoll システムに記録されたデータに基づいているが，他のデジタル教科書システムであっても同様の形式のログが記録されていれば，OpenLA ライブラリを利用することが可能である．公開データセットには 4 種類のファイルが

1 九州大学 工学部 電気情報工学科
2 九州大学 大学院システム情報科学研究院
a) murata@limu.ait.kyushu-u.ac.jp
b) atsushi@ait.kyushu-u.ac.jp
c) minematsu@limu.ait.kyushu-u.ac.jp
d) rin@ait.kyushu-u.ac.jp

*1 https://sites.google.com/view/lak20datachallenge

ⓒ 2020 Information Processing Society of Japan

含まれている．

Course_#_EventStream.csv
　デジタル教科書の操作履歴が記録されたファイル．

Course_#_LectureMaterial.csv
　各デジタル教科書のページ数が記録されたファイル．

Course_#_LectureTime.csv
　各回の授業が実施された時間が記録されたファイル．

Course_#_QuizScore.csv
　各学生の最終成績が記録されたファイル．

各ファイルの詳細やイベントログの詳細については，LAK Data Challenge Workshop の Web サイトを確認されたい．

OpenLA で提供する API 関数群は Python 言語で開発されている．Python 3.7.X 系以上をサポートしており，pandas 0.25.X 系，numpy 1.16.X 系，matplotlib 3.1.X 系のライブラリが追加で必要である．AIP 関数群は，「Basic Module」，「Data Extraction Module」，「Data Aggregation Module」，「Data Conversion Module」，「Data Visualization Module」の 5 つのサブモジュールで構成されている．以下では，各モジュールで提供された API 関数を紹介するが，関数の引数に共通して使われる変数名の定義を先に説明しておく．

course_id　授業コースを指定する変数．
contents_id　デジタル教科書を指定する変数．
lecture_week　授業回（第何週目）を指定する変数．
user_id　学生を指定する変数．

2.1 Basic Module

Basic Module は，データを読み込み，授業コース，授業時間，学習者リストなど，分析に必要な基本情報を取得するための関数群を提供している．デジタル教科書のイベントログは，pandas.DataFrame フォーマットで読み込まれる．本モジュールで提供している関数リストを表 1 に示す．紙面の都合上，一部の関数は関数名のみを掲載しているが，詳細については Web サイト[*2]を参照されたい．

2.2 Data Extraction Module

Data Extraction Module は，Basic Module で読み込まれた DataFrame 形式のイベントログを受け取り，指定した情報抽出処理を行うための関数群を提供している．特定の学習者や特定の教科書に関連するイベントログのみを抽出することが可能である．本モジュールで提供している関数リストを表 2 に示す．

2.3 Data Aggregation Module

Data Aggregation Module は，Basic Module で読み込まれた DataFrame 形式のイベントログを受け取り，集計処理を行う関数群を提供している．ユニーク学習者数，学習者 ID のリスト，デジタル教科書の ID リスト，イベントログに含まれるデジタル教科書の操作種別（イベント）などの情報を集計することができる．本モジュールで提供している関数リストを表 3 に示す．

2.4 Data Conversion Module

Data Conversion Module は，Module で読み込まれた DataFrame 形式のイベントログを受け取り，デジタル教科書の各ページに記録されたイベント数やページ間の遷移，ページごとの閲覧情報などに情報変換を行うための関数群を提供している．各学習者が，デジタル教科書のどのページでどのような操作を行ったかなどの情報を容易に取得することができる．本モジュールで提供している関数リストを表 4 に示す．

2.5 Data Visualization Module

Data Visualization Module は，DataFrame 形式のデータを受け取り，可視化グラフを生成するための関数群を提供している．散布図，系列グラフ，棒グラフをサポートしている．本モジュールで提供している関数リストを表 5 に示す．

3. 利用例

OpenLA を利用した学習分析に関するプログラミングコードの例を 2 つ紹介する．ひとつは，コース内で最も成績が良かった学生を特定し，その学生が授業中に教材をどのように読み進めていたのかを可視化する例である (図 1)．コードを実行すると，図 2 に示すグラフが表示される．もうひとつは，各学生の学習特徴を集計し，成績予測のためのモデルを生成する例である (図 3)．こちらの例では，OpenLA に加えて，Python で提供されている他の機械学習ライブラリ（scikit-learn）も併用している．なお，サンプルコードで利用しているデータセットは，LAK Data Challenge Workshop で提供されているものを利用した．

4. おわりに

本稿では，デジタル教科書のイベントログを用いた学習分析のプログラム開発を支援するためのオープンソースライブラリ OpenLA について紹介した．多くの学習分析研究で共通して計算される教科書へのアクセス状況やページごとの閲覧状況などを容易に抽出することが可能である．また，Python で提供されているその他のライブラリとの親和性も高いため，プログラムコード開発の効率化が期待できる．最新の情報については，Web サイト[*2]を参照されたい．

謝辞　本研究は，JST AIP 加速課題 JPMJCR19U1 と科研費基盤研究（A）JP18H04125 の支援を受けた．

[*2] https://www.leds.ait.kyushu-u.ac.jp

表 1　Basic Module の関数群

```
load_eventstream(self):
 :return:  DataFrame containing EventStream (pandas.DataFrame).

contents_ids_in_course(self):
 :return: list of contents ids (list of str).

contents_id_to_lecture_week(self, contents_id):
 :return: lecture week(s) of the contents_id(s) (int or list of int).

lecture_week_to_contents_id(self, lecture_week):
 :return: contents id(s) of the lecture week(s) (str or list of str).
```

その他，同等の仕様の関数で基本情報を取得する関数群．関数名のみ掲載．

```
contents_id_to_num_pages(self, contents_id):
lecture_week_to_num_pages(self, lecture_id):
contents_id_to_start_time(self, contents_id):
lecture_week_to_start_time(self, lecture_week):
contents_id_to_end_time(self, contents_id):
lecture_week_to_end_time(self, lecture_week):
user_num_in_course(self):
users_in_course(self):
user_score(self, user_id):
users_in_selected_score(self, users_list=None, bottom=None, top=None):
```

表 2　Data Extraction Module の関数群

```
select_user(self, dataframe, user_id):
 :return: Extracted result (pandas.DataFrame).
          If the argument “user_id” is given as a list of user ids,
          the function extracts all the users in the list.
```

その他，同等の仕様の関数でデータ抽出を行う関数群．関数名のみ掲載．

```
select_contents(self, dataframe, contents_id):
select_lecture(self, dataframe, lecture_week)
select_operation(self, dataframe, operation_name):
select_marker_type(self, dataframe, marker_type):
select_device(self, dataframe, device_name):
select_page(self, dataframe, bottom=None, top=None):
select_memo_length(self, dataframe, bottom=None, top=None):
select_time(self, dataframe, start_time, end_time):
select_by_lecture_time(self, dataframe, contents_id=None, timing='during'):
```

参考文献

[1] Ogata, H., Oi, M., Mouri, K., Okubo, F., Shimada, A., Yamada, M., Wang, J. and Hirokawa, S.: Learning Analytics for E-Book-Based Educational Big Data in Higher Education, *Smart Sensors at the IoT Frontier*, pp. 327–350 (2017).

[2] Ogata, H., Yin, C., Terai, M., Okubo, F., Shimada, A., Kentaro, K. and Yamada, M.: E-book-based learning analytics in University education, *Proceedings of the 23rd International Conference on Computers in Education, ICCE 2015*, pp. 401–406 (2015).

[3] Okubo, F., Yamashita, T., Shimada, A. and Ogata, H.: A Neural Network Approach for Students' Performance Prediction, *The 7th International Conference on Learning Analytics & Knowledge Understanding* (2017).

[4] Okubo, F., Yamashita, T., Shimada, A., Taniguchi, Y. and Konomi, S.: On the Prediction of Students' Quiz Score by Recurrent Neural Network, *Multimodal Learning Analytics Across Spaces Workshop (CrossMMLA)* (2018).

[5] Shimada, A., Mouri, K., Taniguchi, Y., Ogata, H., Taniguchi, R. and Konomi, S.: Optimizing Assignment of Students to Courses based on Learning Activity Analytics, *Proceedings of the 12th International Conference on Educational Data Mining, EDM 2019* (2019).

[6] Shimada, A., Taniguchi, Y., Okubo, F., Konomi, S. and Ogata, H.: Online Change Detection for Monitoring Individual Student Behavior via Clickstream Data on e-Book System, *8th International Conference on Learning Analytics & Knowledge*, pp. 446–450 (2018).

[7] Yin, C., Yamada, M., Oi, M., Shimada, A., Okubo, F., Kojima, K. and Ogata, H.: Exploring the Relationships between Reading Behavior Patterns and Learning Outcomes based on Log Data from e-books: A Human Factor Approach, *International Journal of Human-Computer Interaction* (2018).

ⓒ 2020 Information Processing Society of Japan

表 3 Data Aggregation Module の関数群

```
user_num_in_dataframe(self, dataframe):
 :return:  number of user ids in dataframe (int)

users_in_dataframe(self, dataframe)
 :return:  list of user ids in dataframe (list of str)
```

その他，同等の仕様の関数でデータ統合集計を行う関数群．関数名のみ掲載．

```
contents_ids_in_dataframe(self, dataframe):
operations_in_dataframe(self, dataframe):
marker_types_in_dataframe(self, dataframe):
device_codes_in_dataframe(self, dataframe):
```

表 4 Data Conversion Module の関数群

```
users_operation_count(self, dataframe, user_list):
 :return:  “user id” vs. “operation”s, i.e., how many times each learner used each operation
          (pandas.DataFrame, columns['user id', ‘each event’]).

users_page_transition_and_event_count(self, dataframe):
 :return:  “user id” vs. “contents id”, “page no”, “operation”s with consideration of page transition,
          i.e., how many times each learner used each operation in each page and in each content.
          The counting is performed every page transition (pandas.DataFrame, columns['user id', 'contents id',
          'page no', ‘each event’]).

users_pageno_and_event_count(self, dataframe):
 :return:  “user id” vs. “contents id”, “page no”, “staying time” , i.e., how many times each learner
          used each operation in each page and in each content. The result is equivalent to the page-wise
          aggregation of “user_page_transition_and_event_count” function (pandas.DataFrame, columns['user id',
          'contents id', 'page no', ‘each event’]).

users_page_transition_and_staying_seconds(self, dataframe):
 :return:  “user id” vs. “contents id”, “page no”, “operation”s with consideration of page transition,
          i.e., how long each learner stayed in each page and in each content. The calculation is performed
          every page transition (pandas.DataFrame, columns['user id', 'contents id', 'page no', 'time_to_enter',
          'time_to_leave', 'staying_seconds']).

users_pageno_and_staying_seconds(self, dataframe):
 :return:  “user id” vs. “contents id”, “page no”, “operation”s, i.e., how long each learner stayed
          in each page and in each content. The result is equivalent to the page-wise aggregation of
           “users_page_transition_and_staying_seconds “ function (pandas.DataFrame, columns['user id',
          'contents id', 'page no', 'staying_seconds']).
```

表 5 Data Visualization Module の関数群

```
scatter_graph(dataframe, column_x, column_y, save_file=None):
 :return:  Draw scatter plots between two columns in DataFrame.
          If the “save_file” is indicated, the graph is saved.

time_series_graph(dataframe, column, graph_type='line', start_time=None, end_time=None, save_file=None):
 :return:  Draw a line series graph of indicated “column”.
          If the “save_file” is indicated, the graph is saved.

operation_count_bar_graph_in_pages(self, user_id, contents_id, operation_name, save_file=None):
 :return:  Draw a bar graph which represents page-wise counting result of each operation used by a specific learner.
          If the “save_file” is indicated, the graph is saved.
```

ⓒ 2020 Information Processing Society of Japan

```
from OpenLA import OpenLA

# Load an EventStream data of a specific course
c1 = OpenLA(files_dir="dclak20/Train", course_id="6b1900c56c")
event_stream_df = c1.load_eventstream()
user_id_list = c1.users_in_course()
contents_id_list = c1.contents_ids_in_course()
score_list = c1.user_score()

# Get the user id who got the best score
top_score = max(score_list)
top_score_users = c1.users_in_selected_score(users_list=user_id_list, bottom=top_score, top=top_score)
a_top_score_user = top_score_users[0]

# Extract the event logs of the learner
top_score_user_df = c1.select_user(dataframe=event_stream_df, user_id=a_top_score_user)

# Extract the operations of a specific lecture material (contents_id) during the lecture time
top_score_in_lecture_df = c1.select_by_lecture_time(dataframe=top_score_user_df,
                                                    contents_id=contents_id_list[0],
                                                    timing='during')

# Draw a line graph
c1.time_series_graph(dataframe=top_score_in_lecture_df, column='pageno',
                     graph_type='line', start_time=None, end_time=None, save_file='example.png')
```

図 1　学習状況の可視化コード例

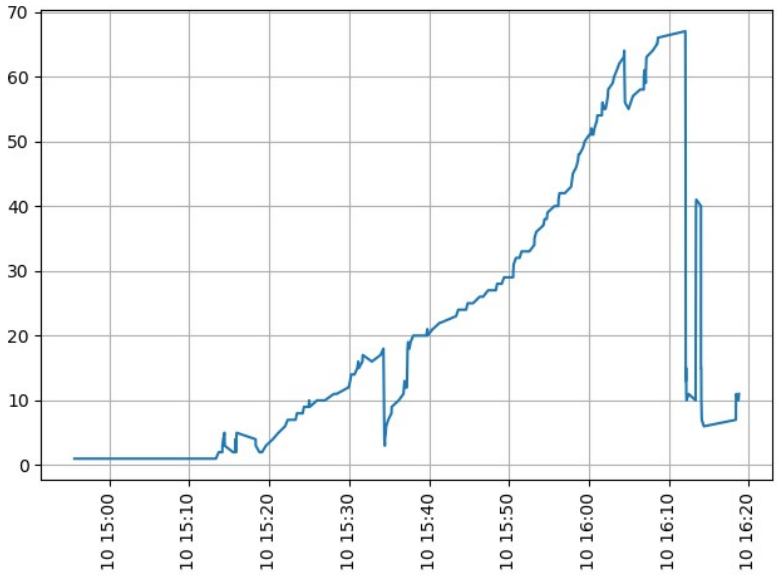

図 2　学習状況の可視化コードによる分析結果の可視化例

```
import pandas as pd
from OpenLA import OpenLA
from sklearn.ensemble import RandomForestRegressor

def input_output_processing(files_dir, course_id, is_train):
    """
    Processing input and output by OpenLA.
    :param files_dir: The directory containing dataset files (str).
    :param course_id: The course to create input (users feature vector) and output (quiz score)  (str).
    :param is_train: Whether to process train or predict dataset (bool). If False, None is returned for output.
    :return:
      input_df: "user id" vs "feature vector" (pandas.DataFrame)
        5 operations ("NEXT", "PREV", "ADD BOOKMARK", "ADD MARKER", "ADD MEMO") in each of the 8 lectures
        create feature vector, so that the feature vector is 40 dimensional (5 operations x 8 lectures)

      users_score: The quiz score of users in "input_df" (list of float)
    """
    # Load an EventStream data of a specific course and get information about the course
    c = OpenLA(files_dir=files_dir, course_id=course_id)
    event_stream = c.load_eventstream()
    user_list = c.users_in_course()
    contents_ids = c.contents_ids_in_course()
    lecture_weeks = c.contents_id_to_lecture_week(contents_id=contents_ids)
    # Process input
    input_df = pd.DataFrame(index=user_list)
    input_df.index.name = "userid"
    for lecture_week in lecture_weeks:
        # Aggregate operation in each lecture: the number of each operation represents the users feature
        lecture_df = c.select_lecture(dataframe=event_stream, lecture_week=lecture_week)
        user_vs_operation_df = c.users_operation_count(dataframe=lecture_df, user_list=user_list,
                                                       operation_name=["NEXT", "PREV", "ADD BOOKMARK",
                                                                       "ADD MARKER", "ADD MEMO"])
        # Concatenate all lecture features
        input_df = pd.merge(input_df, user_vs_operation_df, on="userid", how="left")
    # Process output
    users_score_list = c.user_score(user_id=user_vs_operation_df.index) if is_train else None
    return input_df, users_score_list

if __name__ == "__main__":
    # Train dataset
    train_files_dir = "dclak20/Train"
    train_course_ids = ['60ab104927', '6b1900c56c', '9a683161f5', '86066cba6d', '792efa2c1b',
                        '34451e8c77', '24a65f29b6', 'dbed6c966a', '39a67f80f4', '65bb6224af']
    X = pd.DataFrame()
    Y = []
    for id in train_course_ids:
        input_df, users_score_list = input_output_processing(files_dir=train_files_dir, course_id=id, is_train=True)
        X = pd.concat([X, input_df], axis=0)
        Y.extend(users_score_list)
    # Predict Dataset
    predict_files_dir = "dclak20/Predict"
    predict_course_id = '68cbab26ff'
    predict_X, _ = input_output_processing(files_dir=predict_files_dir, course_id=predict_course_id, is_train=False)
    # Model training and score prediction
    model = RandomForestRegressor()
    model.fit(X, Y)
    pred_score = model.predict(predict_X)
```

図 3　機械学習ライブラリを併用した成績予測モデル生成コード例