

Web上のデータから表を抽出・整形する WebスクレイピングAPIの提案

本多 佑希^{1,2} 岸本 有生³ 漆原 宏丞² 兼宗 進²

概要：小中高とプログラミングが必修化され、大学でも「数理・データサイエンス・AI」のモデルカリキュラムが示されるなど、プログラミングやデータサイエンスの重要性が増している。データ分析を授業で扱うには Web で公開されているオープンデータを使うことが考えられるが、そういったデータは分析ツールやプログラムから扱いやすい形式になっていない場合も多い。そのため、授業で扱うためには印刷用のデータから表を抜き出したり、不要なデータを削除したり、データの形式を揃えたりといった、データごとに対して個別に前処理を行う必要があり、教員の負担は大きい。そこで、Web上のURIを引数として与えることで、html や pdf、xlsx などの形式から表を自動で検出・抽出・整形する Web スクレイピング API を提案する。この Web スクレイピング API を使うことで、生徒・学生が自分で見つけたデータをそのまま分析に使うことができる。オプションにより、複数検出された表から1つを選択することや、出力形式の指定もできるため、多くの分析ツール・プログラミング言語で扱うことができる。本稿では、この Web スクレイピング API の設計と実装を報告する。

1. はじめに

新学習指導要領で小学校から高等学校までプログラミングが必修化され、プログラミング教育の注目度が高まっている。生徒が全てのプログラムを自分で記述するだけでは、実用的なプログラムを作るのは困難であるため、高校の学習指導要領 [1] では「言語のライブラリ」「OSなどのAPI」などで提供される便利な関数の利用を推奨している。その結果、OpenCV や Python の numpy を利用した画像処理や機械学習が扱えるようになった。

また、文部科学省の教員研修資料「情報 I」[2] では、WebAPI の利用が扱われており、高校「情報 I」の学習に WebAPI を利用することが可能である。現在公開されている WebAPI の例としては、郵便番号から住所を検索できるもの [6] や、住所や日付から気象情報を取得できるもの [7] などがあり、データを検索するものが多い。加えて教員研修資料の中では、オープンデータの活用も挙げられている。しかし、オープンデータの中には、再利用を考慮せず、印刷用に余分な情報を付与しているケースが多い(いわゆる、「ネ申 Excel [9]」など)。加えて、HTML のテーブルと

して提供されているデータはそもそも分析ツールに与えることが難しい。そのため、教員はデータごとに表の抽出、加工などを事前に行う必要がある。

「情報 II」の教員研修資料ではデータサイエンスが大きく扱われ [3]、大学では「数理・データサイエンス・AI」のモデルカリキュラムが示される [4] など、データサイエンス教育の重要性も大きく高まっている [8]。情報処理学会が公開する MOOC 教材でもオープンデータの活用や統計は大きく扱われている [5]。そこで、Web上のデータから表を抽出し、データ分析ツールに入力しやすい形式に変換する Web スクレイピング API を検討した。WebAPI として提供することにより、様々な言語・ツールで扱うことができる。

本稿では、試作した Web スクレイピング API を報告するとともに、それを適用した授業案について報告する。

2. 設計

2.1 概要

授業利用を考えると、国や自治体が公開しているオープンデータを活用することが考えられる。特に小学校から高校では、地域のデータは重要である。教科書では全国のデータを扱うため、地域ごとのデータは学校ごとに対応する必要がある。しかし、国や自治体のオープンデータは、画面での表示や紙に印刷して使うために用意されているも

¹ 四天王寺大学
Shitennoji University

² 大阪電気通信大学
Osaka Electro-Communication University

³ 大阪電気通信大学高等学校
Osaka Electro-Communication University High School

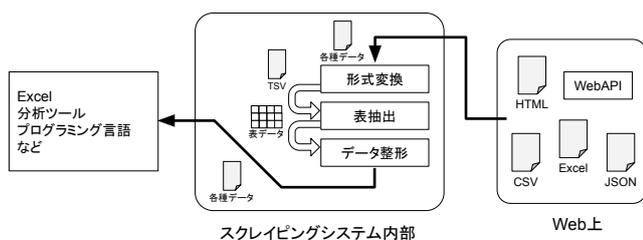


図 1 スクレイピングの流れ

のが多く、プログラムから扱うことを目的としていないため、プログラムから活用することは難しい。

以前報告した内容 [10] の際にスクレイピング対象とした新型コロナウイルス感染症の感染者データも、各自治体によって以下のように多様な形式で提供されていた。

- HTML(本文テキスト、テーブル)
- PDF
- Excel
- CSV
- JSON

そのため、これらのデータを分析する授業を行う際には、以下の準備が必要になる。

- (1) これらのデータを CSV などの扱いやすい形式に変換する
- (2) 余分な部分を削除して必要なデータだけを切り出す
- (3) データ自体を整形する

これらを教員が授業の前準備として用意することは難しい。そのため、Web 上の URI を指定することで、これらの処理を自動で行う WebAPI を提供することにした。

図 1 に、Web スクレイピング API を用いてデータを取得するモデルを示す。Web 上に存在する HTML や XLSX などのファイル、または WebAPI の実行結果などを本システムが取得し、次の流れでデータを整える。

- (1) 形式変換
- (2) 表抽出
- (3) データ整形

2.2 形式変換

前述のように、オープンデータは作成する機関やそのデータによって形式が異なる。汎用的に様々なデータをスクレイピングしたい今回は、これらのデータを統一的形式に変換する必要があった。流れとしては、ファイル形式を検出してから、プログラムで処理しやすい形式に変換を行う。今回は、様々なファイル形式を TSV に統一的に変換してから処理を行うことにした。

2.3 表抽出

オープンデータを調査する中で、1つのファイルの中に複数の表が記述されているデータがあることに気がつい

た。これらは表が明確に分けられていることからわかるように、基本的には同じ表として扱ってはいけない。したがって、検出された複数の表を縦に繋げてスクレイピング結果とするのは好ましくない。

そのため、検出された表をユーザが目視で確認し、その中でどの表をスクレイピング結果として使うかを選択できる機能を検討することにした。

3. データ整形

オープンデータを調査する中で、データを公開している機関によってデータの表現方法が異なる場合が多いことに気がついた。よくある例としては、日付データである。「1994年7月15日」と書かれているデータもあれば、「1994/07/15」と書かれているデータもある。

全角半角の変換や日付データのフォーマット指定などは、分析学習の重要な部分からは外れている。そのため、こういった日付データは、自動で統一的形式に変換することにした。

また、「50%」など、「0.5」という数値を異なる形で表現したにすぎないものもあった。表として一覧で見るとは分かりやすいが、プログラムから処理するには、0.5 という数値である方が好ましい。こういったデータも、統一的に変換することにした。

4. 実装

4.1 概要

Web スクレイピング API の内部では、次のような流れで処理が行われる。

- (1) ユーザからリクエストを受け付ける
 - (2) スクレイピング対象を curl で取得
 - (3) ファイル形式を検出する
 - (4) ファイルを TSV 形式に変換する
 - (5) TSV を読み込み、各データに対して型情報を付け加える
 - (6) 型情報をもとに、表の範囲を検出し、抽出する
 - (7) 型情報をもとに、データの形式を整える
 - (8) ユーザのオプションに応じて表を選択し、文字コードやファイル形式を整える
 - (9) ユーザに対して結果のファイルを返却する
- また、この流れを図示したものを図 2 に示す。

4.2 データ取得

今回は、Web 上に存在するデータを、URI を入力する形で指定する。したがって、システムからは単純に curl で取得することにした。また、授業での利用を考えた際には、同じドメインから大量のアクセスが行われる。この際には、アクセス先の負荷が問題となる。過去には図書館データを定期的に収集したために逮捕される事件も発生してい

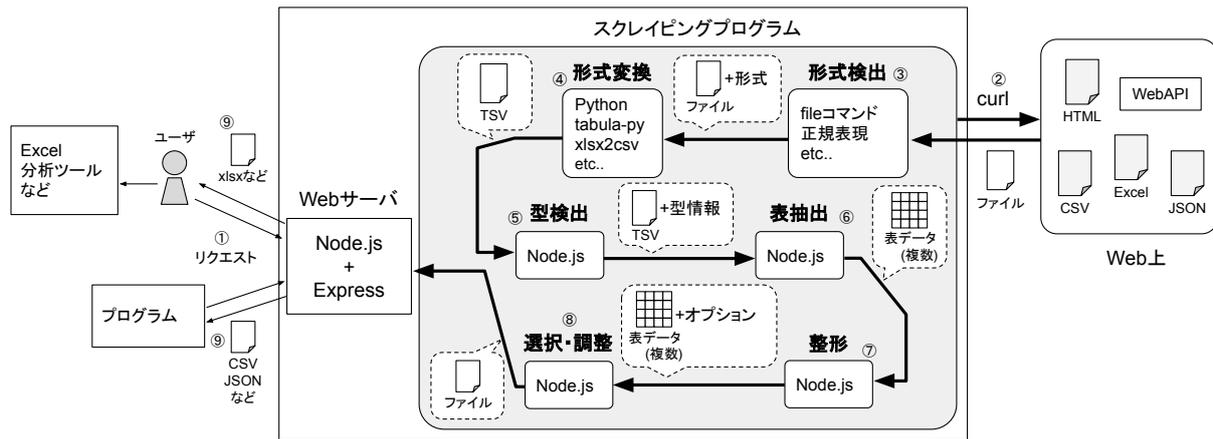


図 2 スクレイピングシステムの内部動作

る [12]。この対処として、URI にアクセスした結果をシステム内部にキャッシュし、同じ URI にアクセスが行われるときには内部のキャッシュを参照するようにした。二次配布の問題は残るが、負荷の問題は解決する。

4.3 形式変換

ファイルを TSV に変換するにあたって、多くの有用なライブラリやツールがあったのでそれを活用することにした。PDF から表を抜き出すために tabula-py を、xlsx は、xlsx2csv を用いて CSV を生成したあと、TSV に変換した。これらのツールを使用するにあたっては、ファイルの形式を正しく認識する必要がある。今回は、file コマンドによる検出と正規表現を組み合わせることにした。これにより、適切に変換に用いるツールを切り替えられるようになった。

4.4 表抽出

表を抽出するにあたって、どこからどこまでが一つの表なのかを判断する必要がある。今回は表の各データの型に着目した。同じような型情報を持つ行が複数続いたら、それは表である可能性が高い。

図 3 の型情報を抜き出したものを表 1 に示す。それぞれ、文字列は S、数値は N、空白は. で示している。また、「-」や「*」、「**」など、ハイフンやアスタリスクのみで構成されたデータは空白と同等に扱う。この型情報の表を見ると、2 行目以降はほぼ同じ型の並びが続いている。この各行をそれぞれ正規表現として扱えば、前後の行と比較することで「同じような形式が続いている」ことを認識できる。また、このような同じ形式が続いている表の手前の行には、おおそヘッダーとなる行が存在する。このように、表の型情報を使えば、ある程度は表の区切りを自動で検出できる。

図 4 の型情報を抜き出したものを表 2 に示す。この例では、「2-1」や「2-2」、「2-3」が文字列として判定されるため上手く表を認識することができない。「2-1」は数値で

No	古墳名称	よみ	墳丘の形	墳丘の長さ	備考
-	天王古墳	てんのうこふん	方墳	11m	
-	鈴山古墳	すずやまこふん	方墳	22m	
-	鏡塚古墳	かがみづかこふん	円墳	26m	国史跡
-	坊主山古墳	ぼうずやまこふん	円墳	10m	
-	狐山古墳	きつねやまこふん	円墳	30m	
-	穂の谷古墳	ほのたにこふん	円墳	47m	
-	グワシヨウ坊古墳	ぐわしよぼうこふん	円墳	61m	国史跡

図 3 オープンデータの例 [13]

表 1 図 3 の型情報を抜き出した例

S	S	S	S	S	S
S	S	S	S	N	.
S	S	S	S	N	.
S	S	S	S	N	S
S	S	S	S	N	.
S	S	S	S	N	.
S	S	S	S	N	.
S	S	S	S	N	S

はないため、この判定は正しいのだが、この場合はその後「1」や「3」なども質的データとして使われているため、数値ではなく文字列として扱うのが正しい。しかし、このデータだけを見て 1 列目の数値が質的データだと判定することはできない。こういったデータへの対処は検討が必要である。

また、頻繁に見られる「複数列にまたがった見出し」への対応も検討中である。どのように検出するかを検討するとともに、どのように扱い、どのようにユーザに提供すべきかは議論が必要である。

4.5 データ整形

「文字」「数値」「欠損値」というざっくりとした型情報で表を抽出した後、もう一度、今度は詳細に型情報を推定する。この際には、上記の 3 パターンに加えて「日付」「単位付き数値」「割合」の型を追加した 6 パターンで判定する。

No	古墳名称	よみ	墳丘の形	墳丘の長さ	備考
1	反正天皇陵古墳	ほんぜいてんのうりょうこふん	前方後円墳	148m	
2-1	仁徳天皇陵古墳	にんとくてんのうりょうこふん	前方後円墳	486m	日本最大の規模
2-2	茶山古墳	ちやまこふん	円墳	56m	
2-3	大安寺山古墳	だいいんじやまこふん	円墳	62m	
3	永山古墳	ながやまこふん	前方後円墳	100m	
4	源右衛門山古墳	げんえもんやまこふん	円墳	34m	
5	塚廻古墳	つかまわりこふん	円墳	35m	国史跡
6	取塚古墳	おさめづかこふん	帆立貝形墳	59m	国史跡

図 4 オープンデータの例 2 [13]

表 2 図 4 の型情報を抜き出した例

S	S	S	S	S	S
N	S	S	S	N	.
S	S	S	S	N	S
S	S	S	S	N	.
S	S	S	S	N	.
N	S	S	S	N	.
N	S	S	S	N	.
N	S	S	S	N	S
N	S	S	S	N	S

- 1 古墳 = テーブル! "data.csv" ファイルから作る。
- 2 円墳 = 古墳! 「墳丘の形 = "円墳"」選択。
- 3 円墳! 「墳丘の長さ」大きい順 表示。

図 6 円墳を大きい順に並べ替えて表示するプログラム例

4.7 プログラミング言語や分析ツールからの利用

プログラムから実際にこのスクレイピング結果を利用する場合には、前述した Web サイトの「URL 生成」を使用する。このボタンを押下すると、スクレイピング結果にアクセスするための URL が発行される。

この際には、検出されたとの表を使用するかや、文字コードやファイルの形式をオプションとして指定することができる。教員はこの URL を生徒に伝え、授業内で利用することになる。プログラムに URL を記述したり、ブラウザからアクセスしてファイルをダウンロードして Excel などのツールで開いたりすることで、実際に分析に取り掛かることができる。

5. 授業で利用する構想

この Web スクレイピング API を使うことで、プログラミングによるオープンデータの活用が行いやすくなると考えられる。実際には、生徒にオープンデータの例を示し、自分で興味を持つデータを調査させ、それをプログラミングによって処理・可視化させる演習が考えられる。

従来は教員が予め分析データを用意する必要があった。そのため、生徒が自分で見つけたデータを学習に使用することは難しく、それをプログラミングで行うことは特に現実的ではなかった。しかし、この Web スクレイピング API を使うとプログラムから扱いやすい形式に自動で整えられる。

図 6 に、図 4 をファイルダウンロードして、ドリトルで扱う例を示す。墳丘の形が円墳であるものを、大きい順に並べ替えて表示する。もちろんこのファイルは、Excel やその他ツールでも扱うことができる。

このデータを ConnectDB [11] に読み込んだ例を図 7 に示す。このように、多くの分析ツールで扱うことができるのは、大きな利点である。教員の手間なく、Web 上のデータを授業で扱えることは魅力である。こういったグラフィカルに分析が行えるツールを使うと、手軽にデータを可視化・分析して議論することができる (図 8)。

6. まとめと今後の展望

本稿では、授業で利用するための WebAPI の提供について提案した。汎用的にオープンデータをスクレイピングする手法を検討し、実際にいくつかのサイトでは表を抽出、プログラムから利用できることを確認している。しかし、まだ実際に授業実践に用いることができていない。型の推

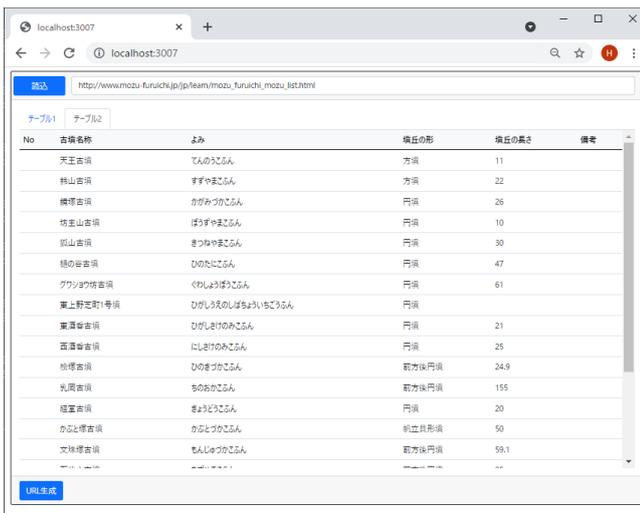


図 5 スクレイピング結果を確認する画面

その上で、数値はカンマを外し、日付の表現方法は正規化する。パーセント付きの割合は百分率に正規化し、単位が付いている数値は単位を外す。このような処理によって、プログラムから扱いやすい形式に整える。

4.6 結果の確認

スクレイピングが正しく行えるかどうか、Web 上で確認できるようにした。この Web サイトの画面を図 5 に示す。URL を入力することで、どのようなテーブルを得られて、どのような結果になったかを確認できる。検出されたテーブルが複数存在した場合、それらはタブとして分かれて表示される。ユーザはこの結果を見ながら、プログラム上でどのデータを使うかを確認する。

