DP レポート

オープンソースによる Twitter 検索およびデータ可視化の方法

江谷典子（Peach・Aviation（株））

概要　本稿では，ツイートデータの可視化を行うにあたり，開発環境を準備する手順や API の使い方を明確にして，基本となる Twitter の検索，単語の出現頻度リスト作成，出現頻度の Word Cloud による可視化，地名の地図マーカー表示，を Python のプログラムを用いて解説を行う．また，日本語および英語のツイートデータを利用した実験結果を示す．本プログラムは GitHub にて公開している．

1. はじめに

潜在的顧客の動向や特徴を調査するため，リアルタイム性と拡散力があり，登録しなくても情報を閲覧できる点に優れている Twitter に着目した．目的に応じたツイートデータを収集し，機械学習，自然言語処理とテキスト分析，データマイニングを用いたデータ分析を行った結果，何らかの知見を発見できるようなプロセスを支援するシステム開発を目指している．本稿では，ツイートデータの収集やデータ分析に関するオンラインドキュメントやオープンソースは散在しているので，開発環境を準備する手順や API の使い方を明確にして，基本となる Twitter の検索，単語の出現頻度リスト作成，出現頻度の Word Cloud による可視化，地名の地図マーカー表示，を Python のプログラムを用いて解説を行う．このプログラムは，下記の URL から取得できる．

https://github.com/NorikoEtani/DP

プログラムを参照しながら本稿を読んで利用して頂きたい．プログラムの構成は図 1 に示す.

2. 準備

筆者の開発環境は次の通りである.

- システムの種類：64 ビットオペレーティングシステム
- Windows のエディション：Windows 10 Home
- Java version 9
- Python 3.5.2/ Anaconda 4.1.1（64 ビット）

Python の環境を構築する際に，Python 本体だけではなく，機械学習や科学計算でよく使うライブラリがたくさんまとめられているディストリビューション「Anaconda」を Windows にインストールしておく[1], [2]. 次の手順で準備する.

2.1 Twitter API を使用するための「Consumer API keys」「アクセストークン」の取得

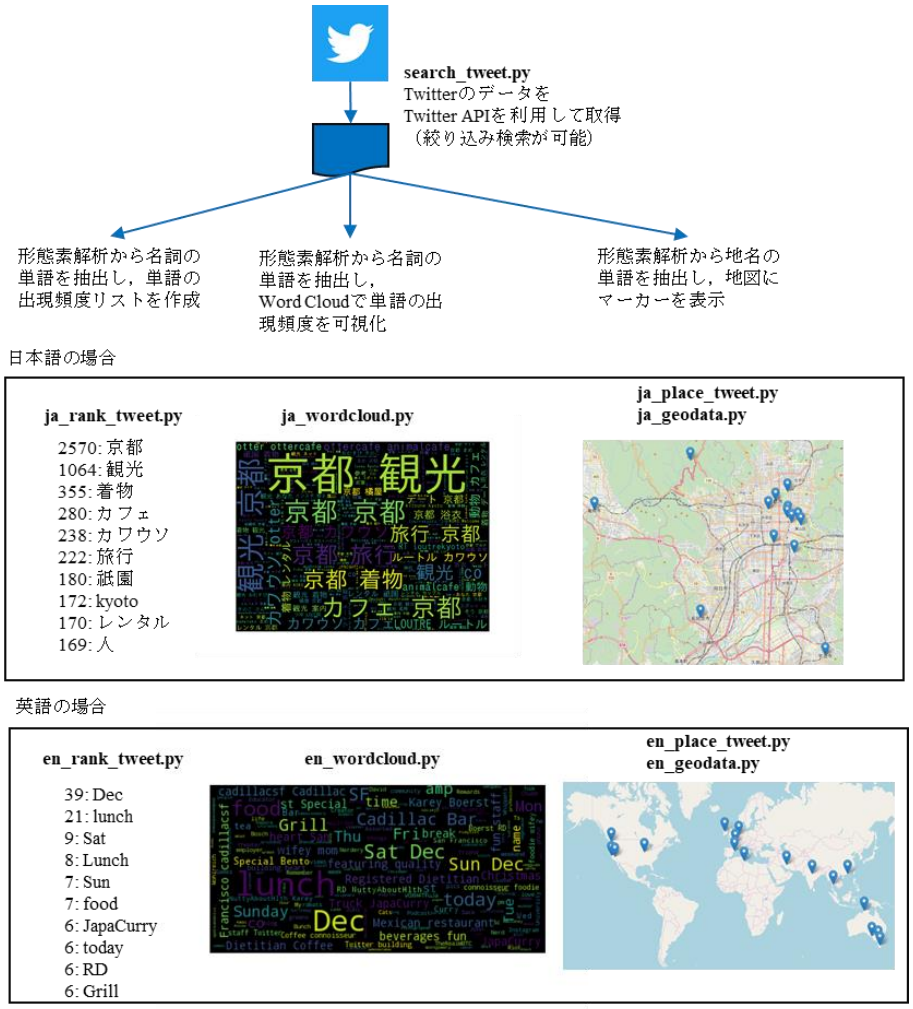

図 1　プログラムの構成

文献[3]を参考にしてアカウント申請を行い，下記に示した「Consumer API keys」および「Access token & access token secret」を取得する．

Consumer_key = '<Twitter API 登録申請して取得した API key>'

Consumer_secret = '<Twitter API 登録申請して取得した Consumer_secret>'

Access_token = '<Twitter API 登録申請して取得した Access_token>'

Access_secret = '<Twitter API 登録申請して取得した Access_secret>'

2.2 認証ライブラリのインストール

Twitter API への認証を行うためのライブラリである．文献[4]を参考にして Anaconda Prompt で「 pip install requests-oauthlib 」 にてインストールする．

2.3 日本語形態素解 MeCab のインストール

MeCab は日本語形態素解析システムである．文献[5],[6]を参考にして MeCab をインストールする．Python3 のソースコードのデフォルトエンコーディングは UTF-8 で，文字列は Unicode を保持している．インストールする際に辞書の文字コードの指定がある場合は UTF-8 を選択するようにする[6]. 筆者は，MeCab 0.996 を利用している．

2.4 英語形態素解析 NLTK のインストール

NLTK（Natural Language Toolkit）は Python 用自然言語処理用ライブラリである．文献[7]を参考にして NLTK をインストールする．Anaconda Prompt で「 pip install nltk 」 にてインストールした後，wordnet のコーパスを Python インタプリタからダウンロードする．GUI の画面が起動するので，「ALL」を選択して，ダウンロードする．

2.5 World Cloud のインストール

Word Cloud は出現頻度が高い単語を複数選び出し，その頻度に応じた大きさで図示する手法のモジュールである．文献[8],[9]を参考にして Anaconda Prompt で「 pip install wordcloud 」にてインストールする．次に，Word Cloud で表示する文字のフォントをダウンロードする．例えば，TTF ファイル　4 書体パック (Ver.003.03) IPAfont00303.zip をダウンロードし， IPAfont00303¥ipag.ttf をインストールインする[10].

2.6 Geocoder のインストール

地図にマーカーを表示させるために オープンライセンスの地図のベースである Open Street Map における地点の緯度経度を取得するライブラリである．文献[11],[12]を参考にして Anaconda Prompt で「 pip install geocoder 」にてインストールする．

2.7 Folium のインストール

Folium は，javascript ライブラリの leaflet を，Python で使えるようにしたものである．文献[13],[14],[15]を参考にして Anaconda Prompt で「 pip install folium 」にてインストールする．

ひとまず必要なインストールを行ったが，開発環境によっては不足しているライブラリやモジュールによりプログラム実行エラーが表示されると思う．その都度、該当のライブラリあるいはモジュールをインストールして頂きたい．

3. Twitter 検索

3.1 Twitter レスポンスデータ構造

Twitter を検索するには，Twitter Search API を使用する．文献[16]では，次の URL https://api.twitter.com/1.1/search/tweets.json からのレスポンスのデータ構造を解説している．search_tweet.py の該当箇所を図 2 に示す．

```
for tweet in tweets["statuses"]:
    tweet_id = tweet[u'id_str']
    text = tweet[u'text']
    created_at = tweet[u'created_at']
    user_id = tweet[u'user'][u'id_str']
    user_description = tweet[u'user'][u'description']
    screen_name = tweet[u'user'][u'screen_name']
    user_name = tweet[u'user'][u'name']
```

図 2　レスポンス部（search_tweet.py）

3.2 Twitter Search API

文献[17]では前述の URL を利用した検索でのパラメータを示す. search_tweet.py の該当箇所を図 3 に示す.

```
params = {
    "q": search_word,
    "lang": "ja",
    "result_type": "recent",
    "count": "15"
    }
```

図 3　検索パラメータ（search_tweet.py）

3.3 Twitter 検索

文献[17]で示しているパラメータ「q」にキーワードを設定して検索ができる. さらに，クエリ演算子を用いると絞り込み検索ができる[18]．ただし，運営側が設定したシステムの条件により検索結果が出てこないことがある[19]．筆者のユーザーID 名では検索が出来なかった．文献[18],[20]ではクエリ演算子を用いた検索一覧を示す.

パラメータ「lang」を利用した lang 検索で指定した言語に絞って検索ができる[17]．また，パラメータ「geocode」を利用した geocode 検索でツイートされた場所を検索できる[17]．図 4 では，サンフランシスコ地点から半径 1 マイル（1.6 キロメートル）以内で投稿されたツイートが対象となる例を示す.

```
params = {
    "q": search_word,
    "geocode": "37.7749,-122.4194,1mi",
    "lang": "en",
    #"lang": "ja",
    "result_type": "recent",
    "count": "15"
    }
```

図 4　検索パラメータ lang **および** geocode **の設定**（search_tweet.py）

3.4 制限事項

Twitter の過去ツィートを見ることができる限界は最新 3,200 ツィートまでと公式発表されている[21]．また，Twitter API にアクセスできる回数を制限する新しい基準を導入している．1 アプリ当たりツイートとリツイート（合算値）は 3 時間で 300 件，「いいね」は 24 時間で 1000 件，フォローは 24 時間で 1000 件，ダイレクトメッセージは 24 時間で 1 万 5000 件，という基準を設けている[22]．さらに，Twitter API には一定時間内にアクセスできる回数に上限がある．例えば，https://api.twitter.com/1.1/search/tweets.json の検索では，15 分間に 180 回までと決まっており，この規定回数を超えてアクセスするとエラーになる．15 分毎にアクセス制限はリセットされる[23].

4. データ可視化

4.1 単語の出現頻度

4.1.1 リスト作成

日本語および英語の形態素解析を行い，名詞，動詞，形容詞の単語を抽出し，単語の出現頻度を降順に並べたリストを作成する．本稿では名詞のみを抽出する．

日本語の場合（ja_rank_tweet.py を参照），MeCab を利用して形態素解析を行い，名詞の単語を抽出し，単語の出現頻度を計算し，結果をソートしてリストを作成する．「UnicodeEncodeError: 'cp932' codec can't encode character」という CP932 へ変換できないコードが含まれているために発生するエラーを回避するため，そのコードを無視するように記述する（図 5）．削除したい単語がある場合はリストに記述し，単語抽出時に無視するように処理を行うことができる（図 6）．

```
with open(fname,'r',encoding="utf-8") as f:
    reader＝f.readline()
    #UnicodeEncodeErrorを避けるため
    before_reader＝reader.encode('cp932',"ignore")
    reader_after＝before_reader.decode('cp932')
```

図 5　CP932 コードを無視する記述（ja_rank_tweet.py）

```
#削除したい文字
del_words＝[ 'https','*','-','#','@','://','(',')','/',':','.',　#','_','t','co','RT']
```

図 6　削除したい単語リスト（ja_rank_tweet.py）

英語の場合（en_rank_tweet.py を参照），NLTK を利用して形態素解析を行い[24]，名詞相当である品詞タグ「NN(名詞)」「NNS(名詞の複数形)」「NNP(固有名詞)」「NNPS(固有名詞の複数形)」の単語を抽出し，単語の出現頻度を計算し，結果をソートしてリストを作成する．NLTK の品詞については，文献[25],[26]を参照して頂きたい．日本語の場合と同様に，CP932 コードを無視する記述や削除したい単語リスト記述は可能である．

4.1.2 Word Cloud 作成

日本語および英語の形態素解析を行い，名詞，動詞，形容詞の単語を抽出し，Word Cloud を用いて単語の出現頻度に応じた大きさで単語を表示する．本稿では名詞のみを抽出する．

日本語の場合（ja_wordcloud.py を参照），MeCab を利用して形態素解析を行い，名詞の単語を抽出し，Word Cloud による可視化を行う．英語の場合（en_wordcloud.py を参照），NLTK を利用して形態素解析を行い，名詞相当である品詞タグ「NN(名詞)」「NNS(名詞の複数形)」「NNP(固有名詞)」「NNPS(固有名詞の複数形)」の単語を抽出し，Word Cloud による可視化を行う．

文献[27]では，画像でマスク処理をする Word Cloud の作成方法を解説している．図 7 は日本語の場合を利用して，右端のアリスの画像をマスク画像として作成した Word Cloud を左端と中央に示している（ja_wordcloud_image.py を参照）。中央部は、右端のマスク画像の色から文字の色付けを行っている．

図 7　Word Cloud の出力（ja_wordcloud_image.py）

4.2 地図マーカー表示

4.2.1 地名の抽出

日本語の場合（ja_place_tweet.py を参照），地名は，MeCab で形態素解析を行うと下記のように「名詞,固有名詞,地域」というタグが付くので抽出する．

京都　名詞,固有名詞,地域,一般,*,*,京都,キョウト,キョート

英語の場合（en_place_tweet.py を参照），地名は，NLTK chunk.ne_chunk を用いた固有表現抽出により，下記のように GPE に存在するエンティティの場所を抽出する[12]．ただし，場所以外の単語も抽出できるので注意が必要である．

(GPE Tokyo/NNP)

4.2.2 geocoder から緯度経度取得と地図マーカー表示

抽出した地名から緯度経度を Open Street Map から取得し，Folium で地図にマーカーを表示する HTML ファイルを出力する．日本語および英語ともプログラムは共有できるが，本稿では地図のズームの違いやある地域内のマーカーを限定するため，共有していない．

日本語の場合（ja_geodata.py を参照），地図の基準として兵庫県明石市を設定し、日本近郊を表示できる倍率に設定する（図 8）．本プログラムでは，京都府内のマーカーを表示させるために Open Street Map から緯度経度を取得した時，address プロパティ（図 9）に「京都府」がある地名のみを抽出している(図 10)．Geocoder から取得できるプロパティについては文献[14]を参考にして頂きたい．

```
# 地図の基準 兵庫県明石市に設定
japan_location = [35, 135]

# 基準地点と初期の倍率(日本近郊）を指定し、地図を作成する
map = folium.Map(location=japan_location, zoom_start=5)
```

図 8　地図基準値と倍率設定（ja_geodata.py）

```
>>> import geocoder
>>> ret = geocoder.osm('京都嵐山', timeout=5.0)
>>> print(ret.latlng)
[35.0096056, 135.6666769]
>>> print(ret.address)
嵐山, 西京区, 京都市, 京都府, 近畿地方, 616-0004, 日本
```

図 9　geocoder から取得した address プロパティ

```
ret = geocoder.osm(reader_after, timeout=5.0)
if ret.ok != False:
    #UnicodeEncodeErrorを避けるため
    s_add = ret.address.encode('cp932', "ignore")
    add_after = s_add.decode('cp932')

    #京都府内のマーカーのみ表示
    if add_after.find('京都府') > -1:
        latitude = ret.latlng[0]
        longtude = ret.latlng[1]
        name = reader_after
        folium.Marker(location=[latitude, longtude], popup=name).add_to(map)
```

図 10　address プロパティから京都府を抽出（ja_geodata.py）

英語の場合（en_geodata.py を参照），地図の基準として兵庫県明石市を設定し、世界地図を表示できる倍率に設定する（図 11）．特定の地域を抽出せずに Open Street Map から緯度経度を取得している．

```
# 地図の基準 兵庫県明石市に設定
japan_location = [35, 135]

# 基準地点と初期の倍率(日本近郊）を指定し、地図を作成する
map = folium.Map(location=japan_location, zoom_start=1)
```

図 11　地図基準値と倍率設定（en_geodata.py）

5. 実験

解説してきた Twitter 検索とデータ可視化の実験結果を図 1 に示す．日本語の場合は，検索クエリ「#京都観光」，lang「ja」(日本語)，にて検索を行い，名詞の出現頻度リストおよび Word Cloud を作成した．地図のマーカーは，京都府内の地名に限定して作成を行った．英語の場合は，検索クエリ「lunch」，geocode「サンフランシスコ地点から半径 1 マイル（1.6 キロメートル）以内」，lang「en」(英語)，にて検索を行った．ともに，出現頻度に応じた Word Cloud を出力している．地名から緯度経度を取得するプログラムでは，国内の地名と同じ地名が海外にもある場合を考慮していない．例えば，京都の嵐山と中国の嵐山の区別を行わない場合，geocoder は中国の嵐山の緯度経度を出力している．この点は分析者の目的に応じて調整を行って頂きたいと考える．

6. まとめ

基本的な Twitter 検索やデータ可視化を行うために本稿とプログラムを提供した．ビッグデータであるツイートデータを可視化することで，利用者の動向調査を行うことができる可能性を示すことができた．

参考文献

1) Anaconda を Windows にインストールする手順 (2019), https://weblabo.oscasierra.net/python-anaconda-install-windows/（2020 年 1 月 1 日現在）
2) Anaconda で Python 環境をインストールする (2018), https://qiita.com/t2y/items/2a3eb58103e85d8064b6（2020 年 1 月 1 日現在）
3) Twitter API 登録 (アカウント申請方法) から承認されるまでの手順まとめ (2019), https://qiita.com/kngsym2018/items/2524d21455aac111cdee（2020 年 1 月 1 日現在）

4) requests-oauthlib 1.3.0 (2019),
https://pypi.org/project/requests-oauthlib/（2020 年 1 月 1 日現在）
5) 64Bit 版 Windows10 で Mecab をインストールする方法 (2017),
https://toolmania.info/post-9815/（2020 年 1 月 1 日現在）
6) Python と MeCab で形態素解析(on Windows) (2019),
https://qiita.com/menon/items/f041b7c46543f38f78f7（2020 年 1 月 1 日現在）
7) Python,NLTK で自然言語処理 (2019),
http://haya14busa.com/python-nltk-natural-language-processing/（2020 年 1 月 1 日現在）
8) Word Cloud でツイートを可視化してみた(python) (2018),
https://qiita.com/turmericN/items/04cd0b40f91076f0ef42（2020 年 1 月 1 日現在）
9) Python Twitter からツイートを取得してテキスト分析(wordcloud で見える化) (2018),
https://qiita.com/kngsym2018/items/3719f8da1f129793257c（2020 年 1 月 1 日現在）
10) IPA フォント ダウンロードページ (2012),
https://ipafont.ipa.go.jp/old/ipafont/download.html（2020 年 1 月 1 日現在）
11) Python 製ジオコーディングライブラリ Geocoder を試す (2018),
https://astropengu.in/blog/18/（2020 年 1 月 1 日現在）
12) Geocoder: Simple, Consistent (2019),
https://geocoder.readthedocs.io/（2020 年 1 月 1 日現在）
13) Folium: Python で地図可視化 (2019),
https://takaishikawa42.hatenablog.com/entry/2019/01/11/234716（2020 年 1 月 1 日現在）
14) Folium: Python でデータを地図上に可視化 (2016),
https://qiita.com/nanakenashi/items/824c0cb16860ca59a424（2020 年 1 月 1 日現在）
15) Folium (2013),
https://python-visualization.github.io/folium/（2020 年 1 月 1 日現在）
16) Twitter REST API データ構造 (2016),
https://qiita.com/kenmatsu4/items/23768cbe32fe381d54a2（2020 年 1 月 1 日現在）
17) Twitter Documentation Search Tweets Standard search API (2019),
https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets
（2020 年 1 月 1 日現在）
18) Twitter Documentation Search Tweets Standard search operators (2019),
https://developer.twitter.com/en/docs/tweets/search/guides/standard-operators
（2020 年 1 月 1 日現在）
19) Twitter 検索で調べても出てこないユーザー名やアカウントのなぜ (2018),
https://startupsns.com/noreflected-in-thesearch-results-1385（2020 年 1 月 1 日現在）
20) Twitter（ツイッター）の検索コマンド全 19 選 日付やアカウントを指定して探す (2019),
https://mag.app-liv.jp/archive/81735（2020 年 1 月 1 日現在）
21) Twitter 過去ツイートはどこまで遡れる?限界以降の調べ方 (2019),
https://startupsns.com/how2reserch-pasttweets-2593（2020 年 1 月 1 日現在）
22) Twitter、API へのアクセス回数を制限 ツイートとリツイートは 3 時間で 300 件まで (2018),
https://www.itmedia.co.jp/news/articles/1810/23/news109.html（2020 年 1 月 1 日現在）
23) TwitterAPI でツイートを大量に取得。サーバー側エラーも考慮（python で） (2019),
http://ailaby.com/twitter_api/（2020 年 1 月 1 日現在）
24) 【python】nltk で英語の形態素解析 (2017),
https://www.haya-programming.com/entry/2018/03/21/234126（2020 年 1 月 1 日現在）
25) NLTK の使い方をいろいろ調べてみた (2019),
https://qiita.com/m__k/items/ffd3b7774f2fde1083fa（2020 年 1 月 1 日現在）
26) NLTK で IOB タグ付けと頻出単語描画とストップワード除去とシノニムを探す (2018)情,
http://hatunina.hatenablog.com/entry/2018/04/14/224812（2020 年 1 月 1 日現在）
27) Word Cloud for Python documentation 1.6.0 (2019),
http://amueller.github.io/word_cloud/index.html（2020 年 1 月 1 日現在）

江谷 典子（正会員）kerotan@kcn.ne.jp
2001 年 3 月奈良先端科学技術大学院大学情報科学研究科博士後期課程修了 博士（工学）. 現在，Peach・Aviation 株式会社勤務．人工知能，データ活用による業務改善や新規ソリューションの情報システム企画立案開発運用に従事.

投稿受付：2020 年 1 月 2 日
採録決定：2020 年 1 月 20 日
編集担当：立床雅司（三菱電機）