
Social Media Mining For Wheater Data Documentation

Release 0.1

Dominic Looser

August 26, 2015

CONTENTS

1	Flickr	3
1.1	API	3
1.2	Python Library	4
2	Twitter	5
2.1	Misc	5
2.2	Api	5
2.3	Tweepy	6
2.4	Geolocation	6
3	Results	7
4	Codebase	9
4.1	Important Libraries	9
5	Modules	11
5.1	apis package	11
5.2	config module	12
5.3	flickr_analysis module	12
5.4	geo module	12
5.5	secrets module	13
5.6	store module	13
5.7	twitter_analysis module	13
5.8	utils module	13
	Python Module Index	15
	Index	17

Contents:

FLICKR

- Created by Ludicorp in 2004
- Acquired by Yahoo in 2005
- 6 billion images in 2011 (we)
- 87 million registered users in 2013 (we)
- 3.5 million new images daily in 2013 (we)
- Written in PHP

1.1 API

- REST endpoint: <https://api.flickr.com/services/rest/>
- Return formats: XML, JSON, ...
- Parameters: method, api_key, format

1.1.1 flickr.photos.search

Parameters: * woe_id: A 32-bit identifier that uniquely represents spatial entities * place_id: A Flickr place id

Response structure: - photos

- page
- pages
- perpage
- **photo**
 - id
 - latitude
 - longitude
 - place_id
 - title
 - woeid

1.1.2 flickr.places.getInfo

Get informations about a place. Parameters: * woe_id * place_id

response structure:

rsp > place > country

> shapedata > polylines > polyline > urls > shapefile

Attributes: rsp: stat place: place_id, woeid, latitude, longitude, place_url, place_type, place_type_id, timezone, name, woe_name, has_shapedata country: place_id, woeid, latitude, longitude, place_url shapedata: created, alpha, count_points, count_edges, has_donuthole, is_donuthole

1.1.3 woe id vs place id

WOE = where on earth

1.2 Python Library

We use the library called flickrapi. Documentation: <http://stuvel.eu/media/flickrapi-docs/documentation/>

TWITTER

2.1 Misc

- 140 Characters per tweet
- 1.9 million tweets January 2009 (twitter api: up and running, p.4)
- 340 milion tweets each day (2012)
- launched July 2006
- Twitter Inc in San Francisco

2.2 Api

- rest-api vs. streaming api

schema:

- text
- created_at
- coordinates
- place
- **entities**
 - **hashtags**
 - * text

2.2.1 REST-api

<https://api.twitter.com/{version}>

Search

The Search API is not complete index of all Tweets, but instead an index of recent Tweets. At the moment that index includes between 6-9 days of Tweets. (<https://dev.twitter.com/rest/public/search>)

2.3 Tweepy

Python library used to connect to Twitter API through python.

Schema Place full_name

Schema Status streaming-api: contributors truncated text in_reply_to_status_id id favorite_count author

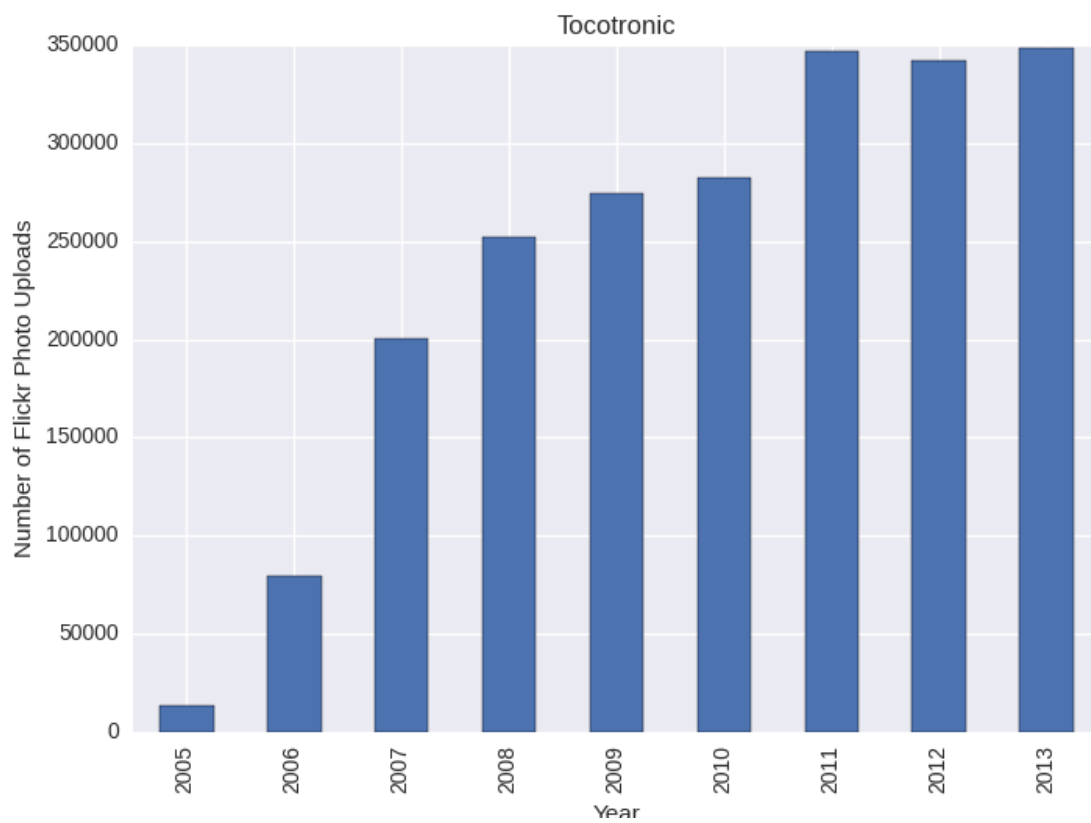
User follow_request_sent profile_use_background_image

_json follow_request_sent profile_use_background_image default_profile_image id verified profile_image_url_https profile_sidebar_fill_color

2.4 Geolocation

- tweet is geotagged by user
- in germany 1% of tweets are geotagged
- Approximately 3-5% of all tweets are geo-enabled (<https://github.com/Ccantey/GeoSearch-Tweepy>)
- induce location from user profile
- induce location from tweet text

RESULTS



CODEBASE

All code is based on Python 2.7

4.1 Important Libraries

- Pandas (data analysis)
- Matplotlib/Seaborn (plotting)
- flickrapi
- tweepy
- nltk (natural language processing)

MODULES

5.1 apis package

5.1.1 Submodules

5.1.2 apis.facebook_api module

5.1.3 apis.flickr_api module

Classes and functions which abstract over the flickr api.

```
class apis.flickr_api.FlickrQuery(tags=None, woe_id=None, year=None,
                                   only_geotagged=False)
```

Bases: *apis.Query*

```
class apis.flickr_api.PhotoCollection(iterator)
```

```
    count_photos()
```

```
    get_random_link()
```

```
    to_points()
```

```
apis.flickr_api.get_photo_collection(query, per_page=200)
```

```
apis.flickr_api.get_points(query, per_page=200)
```

```
apis.flickr_api.retrieve_place_name(woe_id=None, place_id=None)
```

5.1.4 apis.instagram_api module

5.1.5 apis.twitter_api module

Defines important classes Tweet, TwitterSearchQuery, and TwitterStreamingQuery. Enable downloading tweets for search queries and to start streaming with filtering according to a given TwitterStreamingQuery.

```
class apis.twitter_api.PrintingListener
```

Bases: *apis.twitter_api.TwitterStreamListener*

```
    on_status(status)
```

```
class apis.twitter_api.StoringListener(status_handler)
```

Bases: *apis.twitter_api.TwitterStreamListener*

```
    on_connect ()
    on_status (status)
class apis.twitter_api.Tweet (status=None)
    Bases: object
class apis.twitter_api.TwitterSearchQuery (place_id=None, date=None)
    Bases: apis.Query
class apis.twitter_api.TwitterStreamListener
    Bases: tweepy.streaming.StreamListener
    on_error (status_code)
class apis.twitter_api.TwitterStreamingQuery (bounding_box)
    Bases: apis.Query
apis.twitter_api.date_string_to_datetime (date)
apis.twitter_api.download_search_tweets (query)
apis.twitter_api.print_place_info (place_id)
apis.twitter_api.print_places (query_string)
apis.twitter_api.start_streaming (stream_listener, bounding_box=None)
```

5.1.6 Module contents

```
class apis.Query
    Bases: object
```

5.2 config module

5.3 flickr_analysis module

```
flickr_analysis.compute_geotag_usage ()
flickr_analysis.plot_photos_per_year (woe_id=None, use_cache=False)
flickr_analysis.plot_statistics ()
flickr_analysis.save_map (queries, use_cache=False, n_bins=60,
    color_maps=[<matplotlib.colors.LinearSegmentedColormap object
    at 0x7f7abf648ed0>, <matplotlib.colors.LinearSegmentedColormap ob-
    ject at 0x7f7abf648f50>, <matplotlib.colors.LinearSegmentedColormap
    object at 0x7f7abf648f90>], mix_points=False, formats=['png'])
```

5.4 geo module

```
class geo.BoundingBox
    Bases: object
class geo.Map (bounding_box, map_resolution=<MapResolution.INTERMEDIATE: 1>)
    Bases: object
```



```
draw_densities (points, n_bins, color_map='Blues')  
draw_points (points)  
save (path, format='png')  
show ()  
class geo.MapResolution  
    Bases: enum.Enum  
class geo.Point  
    Bases: object
```

5.5 secrets module

5.6 store module

```
class store.StoreType (directory)  
    Bases: object  
  
store.get_search_tweets (place_id, begin, end=None, use_cache=False)  
store.read (query, store_type)  
store.save (query, store_type)
```

5.7 twitter_analysis module

```
class twitter_analysis.Topic (terms)  
    Bases: object  
  
twitter_analysis.contains_topic (tweet, topic)  
twitter_analysis.plot_rain_data ()  
twitter_analysis.print_search_tweet_counts (place_id=None, begin_date=None, end_date=None, use_cache=False)  
twitter_analysis.topic_distribution (topic=None, place_id=None, begin=None, end=None, use_cache=False)
```

5.8 utils module

```
class utils.Stopwatch  
    Bases: object  
  
    start ()  
  
utils.measure_download_time (query, per_page)  
utils.print_totals (queries)
```


a

apis, [12](#)
apis.facebook_api, [11](#)
apis.flickr_api, [11](#)
apis.instagram_api, [11](#)
apis.twitter_api, [11](#)

c

config, [12](#)

f

flickr_analysis, [12](#)

g

geo, [12](#)

s

secrets, [13](#)
store, [13](#)

t

twitter_analysis, [13](#)

u

utils, [13](#)

A

apis (module), 12
 apis.facebook_api (module), 11
 apis.flickr_api (module), 11
 apis.instagram_api (module), 11
 apis.twitter_api (module), 11

B

BoundingBox (class in geo), 12

C

compute_geotag_usage() (in module flickr_analysis), 12
 config (module), 12
 contains_topic() (in module twitter_analysis), 13
 count_photos() (apis.flickr_api.PhotoCollection method), 11

D

date_string_to_datetime() (in module apis.twitter_api), 12
 download_search_tweets() (in module apis.twitter_api), 12
 draw_densities() (geo.Map method), 12
 draw_points() (geo.Map method), 13

F

flickr_analysis (module), 12
 FlickrQuery (class in apis.flickr_api), 11

G

geo (module), 12
 get_photo_collection() (in module apis.flickr_api), 11
 get_points() (in module apis.flickr_api), 11
 get_random_link() (apis.flickr_api.PhotoCollection method), 11
 get_search_tweets() (in module store), 13

M

Map (class in geo), 12
 MapResolution (class in geo), 13
 measure_download_time() (in module utils), 13

O

on_connect() (apis.twitter_api.StoringListener method), 11
 on_error() (apis.twitter_api.TwitterStreamListener method), 12
 on_status() (apis.twitter_api.PrintingListener method), 11
 on_status() (apis.twitter_api.StoringListener method), 12

P

PhotoCollection (class in apis.flickr_api), 11
 plot_photos_per_year() (in module flickr_analysis), 12
 plot_rain_data() (in module twitter_analysis), 13
 plot_statistics() (in module flickr_analysis), 12
 Point (class in geo), 13
 print_place_info() (in module apis.twitter_api), 12
 print_places() (in module apis.twitter_api), 12
 print_search_tweet_counts() (in module twitter_analysis), 13
 print_totals() (in module utils), 13
 PrintingListener (class in apis.twitter_api), 11

Q

Query (class in apis), 12

R

read() (in module store), 13
 retrieve_place_name() (in module apis.flickr_api), 11

S

save() (geo.Map method), 13
 save() (in module store), 13
 save_map() (in module flickr_analysis), 12
 secrets (module), 13
 show() (geo.Map method), 13
 start() (utils.Stopwatch method), 13
 start_streaming() (in module apis.twitter_api), 12
 Stopwatch (class in utils), 13
 store (module), 13
 StoreType (class in store), 13
 StoringListener (class in apis.twitter_api), 11

T

`to_points()` (`apis.flickr_api.PhotoCollection` method), [11](#)

`Topic` (class in `twitter_analysis`), [13](#)

`topic_distribution()` (in module `twitter_analysis`), [13](#)

`Tweet` (class in `apis.twitter_api`), [12](#)

`twitter_analysis` (module), [13](#)

`TwitterSearchQuery` (class in `apis.twitter_api`), [12](#)

`TwitterStreamingQuery` (class in `apis.twitter_api`), [12](#)

`TwitterStreamListener` (class in `apis.twitter_api`), [12](#)

U

`utils` (module), [13](#)