

---

# **Social Media Mining For Wheater Data Documentation**

***Release 0.1***

**Dominic Looser**

September 02, 2015



## CONTENTS

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



Contents:



## FLICKR

- Created by Ludicorp in 2004
- Acquired by Yahoo in 2005
- 6 billion images in 2011 (we)
- 87 million registred 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 > photo

photos: page, pages, perpage, total

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
place > country
country > shapedata
shapedata > polylines, urls
polylines > polyline
urls > shapefile
```

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

### 1.1.3 flickr.places.find

Returns a list of place objects for a given query string.

Parameter: query

Response: | rsp > places | places > place\*

```
rsp: stat
places: query, total
place: place_id, woeid, latitude, longitude, place_url, place_type
```

### 1.1.4 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 Basics

- 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}>

### 2.2.2 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

### 2.3.1 API

- `API.rate_limit_status` ([http://docs.tweepy.org/en/v3.2.0/api.html#API.rate\\_limit\\_status](http://docs.tweepy.org/en/v3.2.0/api.html#API.rate_limit_status))

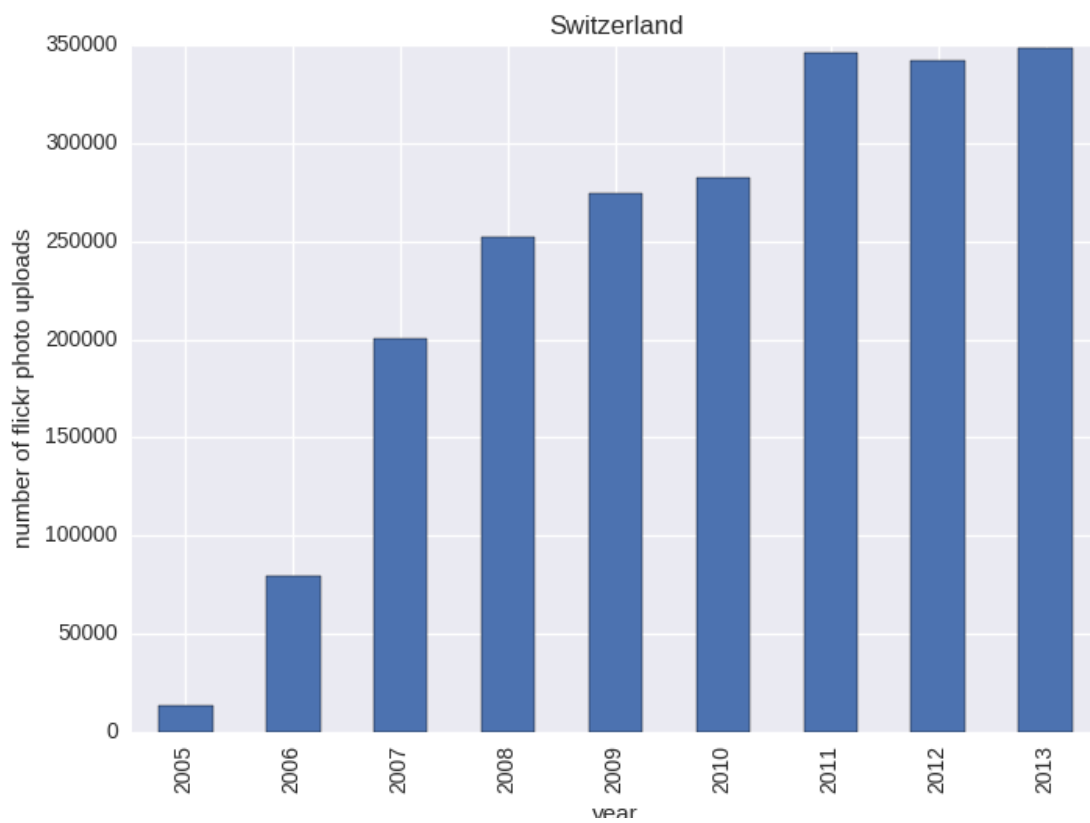
Response Schema:

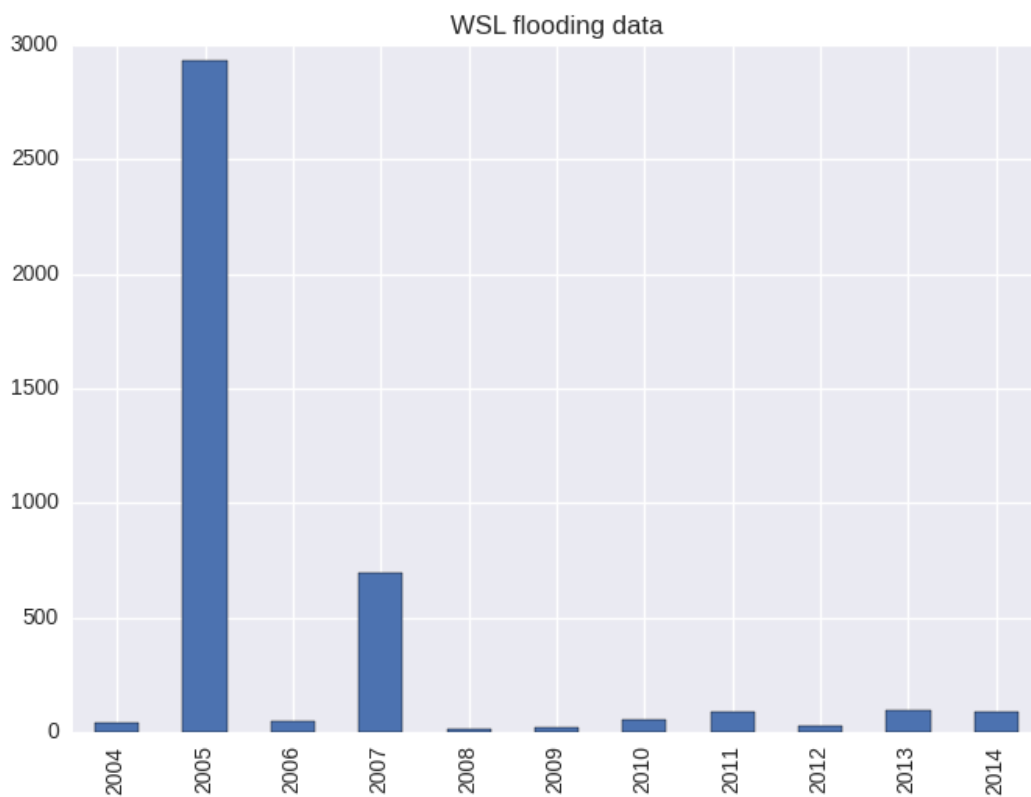
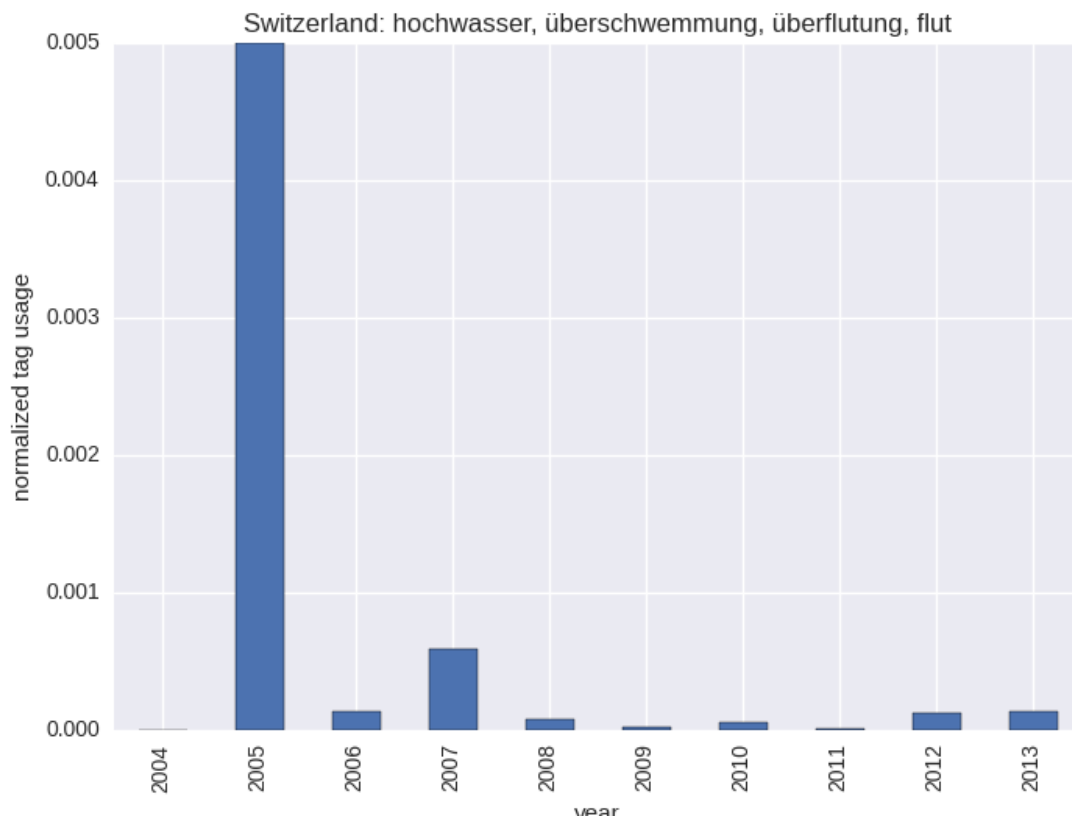
```
{
  rate_limit_context
  access_token
  resources
    *resource_type*
      *resource_name*
        limit
        remaining
        reset
}
```

### 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





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.count_photos (query)
```

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

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

```
apis.flickr_api.print_places (query)
```

```
apis.flickr_api.retrieve_place_name (woe_id)
```

#### 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.Place (place_id)
```

Bases: object

```
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_limit_status ()

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_normalized_tag_usage (tags=None, woe_id=None, save2docs=False)

flickr_analysis.plot_photos_per_year (woe_id=None, use_cache=False, save2docs=False)

flickr_analysis.plot_wsl_flooding_data ()

flickr_analysis.save_map (queries, use_cache=False, n_bins=60,
    color_maps=[<matplotlib.colors.LinearSegmentedColormap object
    at 0x7fc04ab55610>, <matplotlib.colors.LinearSegmentedColormap ob-
    ject at 0x7fc04ab55650>, <matplotlib.colors.LinearSegmentedColormap
    object at 0x7fc04ab55690>], 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)

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.plot_topic_distribution (topic=None, place_id=None, begin=None,
                                          end=None)

twitter_analysis.print_search_tweet_counts (place_id=None, begin_date=None,
                                          end_date=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, [13](#)

## **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), 13

**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  
 count\_photos() (in module apis.flickr\_api), 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), 13  
 draw\_points() (geo.Map method), 13

**F**

flickr\_analysis (module), 12  
 FlickrQuery (class in apis.flickr\_api), 11

**G**

geo (module), 13  
 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), 13  
 MapResolution (class in geo), 13

measure\_download\_time() (in module utils), 14

**O**

on\_connect() (apis.twitter\_api.StoringListener method), 12  
 on\_error() (apis.twitter\_api.TwitterStreamListener method), 12  
 on\_status() (apis.twitter\_api.PrintingListener method), 12  
 on\_status() (apis.twitter\_api.StoringListener method), 12

**P**

PhotoCollection (class in apis.flickr\_api), 11  
 Place (class in apis.twitter\_api), 11  
 plot\_normalized\_tag\_usage() (in module flickr\_analysis), 12  
 plot\_photos\_per\_year() (in module flickr\_analysis), 12  
 plot\_rain\_data() (in module twitter\_analysis), 13  
 plot\_topic\_distribution() (in module twitter\_analysis), 13  
 plot\_wsl\_flooding\_data() (in module flickr\_analysis), 12  
 Point (class in geo), 13  
 print\_limit\_status() (in module apis.twitter\_api), 12  
 print\_place\_info() (in module apis.twitter\_api), 12  
 print\_places() (in module apis.flickr\_api), 11  
 print\_places() (in module apis.twitter\_api), 12  
 print\_search\_tweet\_counts() (in module twitter\_analysis), 13  
 print\_totals() (in module utils), 14  
 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`), [12](#)

## T

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