Social Media Mining For Wheater Data Documentation

Release 0.1

Dominic Looser

August 26, 2015

CONTENTS

Contents:

CHAPTER

ONE

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

- 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, latitutude, 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 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/

CHAPTER

TWO

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

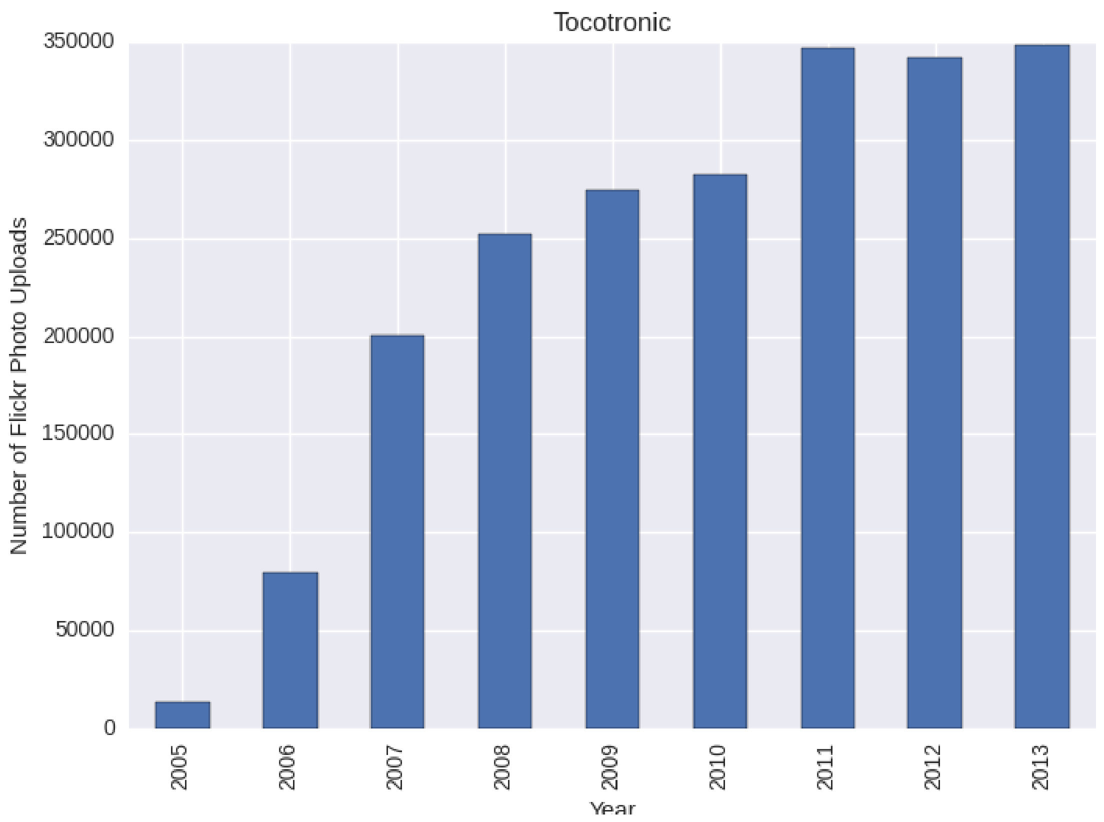
_json follow_request_sent profile_use_background_image default_profile_image id verified profile_image_url_https profile_sidebar_fill_color

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

CHAPTER
THREE

RESULTS



CHAPTER

FOUR

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)

CHAPTER

FIVE

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 object 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*)

PYTHON MODULE INDEX

INDEX