wekeypedia Documentation

Release 0.1.4

tam kien duong

April 26, 2015

Contents

The wekeypedia python toolkit is a set of class and helpers that have been written during the overall wekeypedia project. It main purpose is to give back some shortcuts to the science community. We hope this work will help future data scientist and web scrappers make them win some time about the tedious part of the work, be able to spend more time on the more fun parts and conduct studies with wikipedia materials.

Its main features are :

- data retrieval from the wikipedia API and 3rd party (statistics, semantic web, etc)
- information extraction of API contents
- network modeling of graph structures included in pages architecture
- computation of various metrics (readibility, convergence, lsm, etc)
- generation of reading maps based on a recommandation system

CHAPTER 1

contents:

1.1 Information retrieval

1.1.1 Retrieve and extract information of a wikipedia page

Overview

WikipediaPage (*title=None*, *lang='en'*)

- http://www.mediawiki.org/wiki/API:Query
- http://www.mediawiki.org/wiki/API:Revisions

Methods

Create a page handler

WikipediaPage.__init__([title, lang])	
WikipediaPage.fetch_info(title[, ...])	

__init__
WikipediaPage.**__init__** (*title=None*, *lang='en'*)

fetch_info
WikipediaPage.**fetch_info** (*title*, *opt_params={'inprop': 'url', 'prop': 'info'}*, *lang='en'*)

Retrieving revisions

WikipediaPage.get_revision([revid, force, ...])	Retrieve the content of a revision by its revision id
WikipediaPage.get_revisions_list([extra_params])	Retrieve all the revisions and their info
WikipediaPage.get_current()	Retrieve the content of the current revision

get_revision
WikipediaPage.**get_revision** (*revid=''*, *force=False*, *extra_params={}*)
Retrieve the content of a revision by its revision id

For more paramaters, you can check the wikipedia API documentation.

Examples

```
>>> p = WikipediaPage("Pi")
>>> p.get_content()
```

Parameters

- **revid** (*string, optional*) – Revision id of the article. If none is given, it just check the last revision id give by the wikipedia API
- **force** (*boolean, optional*) – If set to *True*, it fetch the content whatever is in the cache object. Useful to retrieve different version without touching the cache
- **extra_params** (*dict*) – todo: document extra_params@get_content

Returns **content** – todo: document content@get_content

Return type string

get_revisions_list

`WikipediaPage.get_revisions_list(extra_params={})`

Retrieve all the revisions and their info

Returns **revisions**

Return type list

get_current

`WikipediaPage.get_current()`

Retrieve the content of the current revision

Returns **content**

Return type string

Retrieving parts

`WikipediaPage.get_links`([extra_params])	Retrieve links contained by a wikipedia page according to the API
`WikipediaPage.get_categories`([extra_params])	Retrieve a list of all categories used on the provided pages
`WikipediaPage.get_langlinks`()	Retrieve the list of hyperlinks to translation of the current page
`WikipediaPage.get_pageviews`([fr, to])	Retrieve daily page view statistics from http://stats.grok.se/

get_links

`WikipediaPage.get_links(extra_params={})`

Retrieve links contained by a wikipedia page according to the API

Parameters **extra_params** (*dict, optional*) – By default, the method will only retrieve links from the namespace 0 (usual pages) and skipped everything like templates, etc.

You can still get the other namespaces by updating the query with an extra parameters.

```
>>> p.get_links({ plnamspace: 12 })
```

Returns **links**

Return type list

get_categories

`WikipediaPage.`**`get_categories`**(*extra_params={}*)

Retrieve a list of all categories used on the provided pages

Parameters **extra_params** (*dict, optional*) –

- http://www.mediawiki.org/wiki/API:Property/Categories

Returns **links**

Return type list

get_langlinks

`WikipediaPage.`**`get_langlinks`**()

Retrieve the list of hyperlinks to translation of the current page

Returns **langlinks** – List of language codes (e.g "en", "fr", "es", "ru", etc) todo: put a link to a page with the list of languages

Return type list

get_pageviews

`WikipediaPage.`**`get_pageviews`**(*fr='200712'*, *to=''*)

Retrieve daily page view statistics from http://stats.grok.se/

Parameters

- **fr** (*string, optional*) – Start of the range (minimum is december 2007) represented as *yearmonth* (*%Y%m*).
- **to** (*string, optional*) – End of the range represented as *yearmonth* (*%Y%m*).

  If no end date is given, the current date is used as an end date.

Returns **pageviews** – List of page views per day represented as tuples *[(day, views),...]*

Return type list

Extracting editors

`WikipediaPage.get_editors`([revisions_list])	Retrieve revisions and extract editors

get_editors

`WikipediaPage.`**`get_editors`**(*revisions_list=[]*)

Retrieve revisions and extract editors

Parameters **revisions_list** (*list, optional*) – If a list of revisions id is passed as an argument, the method will filter out the revisions that are in that list while still retrieving the list of all revisions firsts.

A more optimal rewriting will be to fetch only the selected revisions.

Returns **editors**

Return type list

Retrieving and parsing diff


```
{
  "comment": "/* Overview */",
  "timestamp": "2007-01-11T19:06:02Z",
  "revid": 100042918,
  "anon": "",
  "user": "129.24.51.153",
  "parentid": 100036516,
  "diff": {
    "to": 100042918,
    "*": "<tr>\n  <td colspan=\"2\" class=\"diff-lineno\">Line 21:</td>\n  <td colspan=\"2\" class=\"
```


WikipediaPage.get_diff([rev_id])	Retrieve diff content between a revision and its predecessor.
WikipediaPage.get_diff_full([rev_id])	Retrieve the full json response from a request for diff.
WikipediaPage.extract_plusminus(diff_html)	Transform HTML Wikipedia API response into a plus/minus dict.
WikipediaPage.count_stems(sentences[, ...])	Count the number of stems in a list of sentences.

get_diff

WikipediaPage.**get_diff**(*rev_id=''*)

Retrieve diff content between a revision and its predecessor. The content is extracted from the API json response. To get the full response, you can still use *get_diff_full*

Examples

Parameters **rev_id** (*string, optional*) – If no revision id is supplied, the method retrieve the diff from the current version of the page and compare it to its predecessor.

Returns **content**

Return type string

See also:

get_diff_full()

get_diff_full

WikipediaPage.**get_diff_full**(*rev_id=''*)

Retrieve the full json response from a request for diff.

Parameters **rev_id** (*string, optional*) – If no revision id is supplied, the method retrieve the diff from the current version of the page and compare it to its predecessor.

extract_plusminus

WikipediaPage.**extract_plusminus**(*diff_html*)

Transform HTML Wikipedia API response into a plus/minus dict. Information extraction is made with BeautifulSoup

Parameters **diff_html** (*string*) –

Returns **diff** – *diff* contains two keys: diff["added"] and diff["deleted"]. Each of those entries correspond to blocks and inline extraction of addition, deletion and substition.

Return type dict

See also:

`count_stems()`

count_stems

`WikipediaPage.`**`count_stems`**(*sentences*, *inflections=None*)

Count the number of stems in a list of sentences.

An optional parameter allows to provide an inflections dictionary in order to register them. It can be usefull when looking for the most used form of a stem to produce more readable outputs.

This function use a dummy normalizer (*self.normalize*) that take a tokenized sentences using the Punkt NTLK parser and apply a simple word normalization process (lowercase, stemmization, lemmatization).

Parameters

- **sentences** (*list*) –
- **inflections** (*dict, optional*) –

Returns **stems**

Return type dict

See also:

`extract_plusminus()`

Page views

`WikipediaPage.get_pageviews`([fr, to])	Retrieve daily page view statistics from http://stats.grok.se/

get_pageviews

`WikipediaPage.`**`get_pageviews`**(*fr='200712'*, *to=''*)

Retrieve daily page view statistics from http://stats.grok.se/

Parameters

- **fr** (*string, optional*) – Start of the range (minimum is december 2007) represented as *yearmonth* (*%Y%m*).
- **to** (*string, optional*) – End of the range represented as *yearmonth* (*%Y%m*).

  If no end date is given, the current date is used as an end date.

Returns **pageviews** – List of page views per day represented as tuples *[(day, views),...]*

Return type list

Function helpers

`url2title`(*url*)

Transform an url into a title

Parameters **url** (*string*) –

Returns **title**

Return type string

url2lang (*url*)

Transform an language code into a title

Parameters **url** (*string*) –

Returns **lang**

Return type string

1.1.2 Make custom queries to the wikipedia api

Le toolkit wekeypedia inclut une classe qui permet de passer des requêtes plus fines et adaptées à des recherches d'information spécifiques et peu généralisables. Par exemple, la plupart des classes implémentées gèrent des objets à une échelle individuelle alors que pour des raisons d'optimisation, il est parfois nécessaire d'affiner les requêtes afin d'en réduire leur nombre.

class api (*lang='en'*)

Parameters **lang** (*string, optional*) –

get (*query*, *method='get'*)

Parameters **query** (*dict*) –

Returns **result**

Return type dict

Examples

Here is piece of code that retrieve all links included in the *Wisdom* page and check if all these links (n=184) have an equivalent in the french wikipedia. It does so by asking for langlinks of 50 pages at once instead of building one query per links. In this case, the network load reduction goes from 184 queries to 4. #win

```
from __future__ import division
from math import ceil
from collections import defaultdict

import wekeypedia
from wekeypedia.wikipedia.api import api as api

def api_bunch(page_titles, lang, req):
  results = defaultdict(list)
  param  = req

  w = api(lang)

  for i in range(0,int(ceil(len(page_titles)/50))):
    param["titles"] = "|".join(page_titles[i*50:i*50+50-1])

    while True:
      r = w.get(param)
      results.update({ p["title"]: p['langlinks'] for pageid, p in r["query"]["pages"].items() if 'la
```

```
      if "continue" in r:
        param.update(r["continue"])
      else:
        break

  return results

def get_lang_projection(pages, source, target):
  """
  Retrieve all correspondance from a set of pages into another language

  Parameters
  ----------
  pages : list
    List of page titles

  Returns
  -------
  correspondances : list
    List of '(redirect(initial page), corresponding page)'
  """

  params = {
    "redirects": "",
    "format": "json",
    "action": "query",
    "prop": "info|langlinks",
    "lllimit": 500,
    "lllang": target,
    "continue":""
  }

  r = api_bunch(pages, source, params)

  return [ (page, t["*"]) for page,tt in r.items() for t in tt if t["lang"] == target ]

u = wekeypedia.WikipediaPage("Wisdom")
pages = list(set([ x["title"] for x in u.get_links() ]))

get_lang_projection(pages, "en", "fr")
```

1.1.3 wikipedia user

class **WikipediaUser** (*lang='en'*, *name=None*)

create a new wikipedia user object

Keyword Arguments

- **name** (*string*)
- **lang** (*string*)

Example

```
>>> from wekeypedia.wikipedia_user import WikipediaUser as User
>>>
>>> u = User(name="taniki")
```

fetch_contribs()
get all contributions from a user

1.2 Computing metrics

1.2.1 Linguistic Style Matching

lsm.compare(text1, text2)	Compare two texts using the Linguistic Style Matching (LSM) [1]_ method
lsm.extract_categories(text)	Extract percentages of LSM word categories over total words counting
lsm.extract_categories_raw(text)	Extract raw counting of LSM word categories

compare

compare(*text1*, *text2*)
Compare two texts using the Linguistic Style Matching (LSM) [1] method

$$LSM_{preps} = 1 - \frac{|preps_1 - preps_2|}{preps_1 + preps_2 + 0.0001}$$

$$LSM = \frac{LSM_{personalpronouns} + LSM_{impersonalpronouns} + LSM_{articles} + LSM_{auxiliaryverbs} + LSM_{high-frequencyadverbs}}{9}$$

Parameters

- **text1** (*string*) –
- **text2** (*string*) –

Returns lsm

Return type dict

References

extract_categories

extract_categories(*text*)
Extract percentages of LSM word categories over total words counting

Parameters **text** (*string*) –

Returns categories

Return type dict

[1] Amy L. Gonzales, Jeffrey T. Hancock, and James W. Pennebaker : Language Style Matching as a Predictor of Social Dynamics in Small Groups. Communication Research 37(1):3-19, 2010.

Examples

```
>>> txt = "One morning, when Gregor Samsa woke from troubled dreams, he found himself transforme
>>> wekeypedia.lsm.extract_categories(txt)
{'personal pronouns': 0.11578947368421053, 'articles': 0.06315789473684211, 'hf adverbs': 0.0, '
```

extract_categories_raw

extract_categories_raw(*text*)

Extract raw counting of LSM word categories

Parameters **text** (*string*) –

Returns **categories**

Return type dict

Examples

```
>>> txt = "One morning, when Gregor Samsa woke from troubled dreams, he found himself transforme
>>> wekeypedia.lsm.extract_categories_raw(txt)
{'personal pronouns': ['his', 'his', 'his', 'his', 'he', 'himself', 'He', 'he', 'he', 'him', 'he
```

CHAPTER 2

installation (with virtual env)

```
$ virtualenv e/py --no-site-packages
$ source e/py/bin/activate
(py)$ pip install wekeypedia
```

CHAPTER 3

todo list:

W

Symbols

A

C

E

F

G

U

W