Specifications of RSS crawlers

Components for related files

1. rsscrawler.py is a main class for crawling all RSS sources from one RSS news/blog source.
2. rsscrawler-doc.txt is the output of pydoc for rsscrawler.py to specify the class.
3. rsscrawler.cfg gives some system configuration and other some options.
4. SpeicalSites.py defines special processing for some special websites.
5. specialsites-doc.txt is the output of pydoc for SpeicalSites.py to specify the class.
6. newsextractor.py includes extracting functions that extract title, date and content from a webpage fetched.
7. newsextractor-doc.py is the output of pydoc for newsextractor.py to specify the class.
8. testcase.py is used to test functions by unittest class.
9. testcase-doc.txt is the output of pydoc for testcase.py to specify the class.
10. test.results.txt is output of running the test case.
11. SpecificationofRSScrawler.pdf is readme file .

Command format as follows:

rsscrawler.py -n <RSS name> -i <sources file> [-s <stopwords file>]

notes: RSS name is required in a special format, e.g., “12345678.cnn”, where, ‘12345678’ means timestamp that is used to generate sub-directory, ‘cnn’ means RSS source name is uesed to generate parent-directory .

“sources file” format as follows:

http://rss.cnn.com/rss/edition.rss
http://rss.cnn.com/rss/edition_world.rss
http://rss.cnn.com/rss/edition_africa.rss
http://rss.cnn.com/rss/edition_americas.rss
...

“stopwords file” format as follows:

a
the
who
…

rsscrawler.cfg format as follows:

```
[System]
timezonedifference = 8
wordsFrequency = false
storagefile = MERGE.TXT

[Specialprocessing]
specialsites = ['newyorktimes', 'straitstimes']
multisource = ['blog-en', 'blog-cn']
```

Role of generated files in running time

The following explanation gives the purpose of some files that are generated in running time:

1. RSSName_wordsfreq.db stores words frequency file generated from fetched web pages in order to visualize words trends of News API.
2. fileName.db (fileName means one RSS source's link address) stores all links of web pages fetched in terms of the RSS source.
3. MERGE.TXT stores updated records that include link, title, source, and date of web pages fetched from last commit in terms of one source.
4. filename.html stores a whole webpage fetched, where filename is gotten by the News' title.

Main algorithm of fetching a RSS source as follows:

```
1.      while a new sub-source is still not fetched:
2.          load all fetched Links from a filename .db file;
3.          load all words frequency statistic from a RSSName_wordsfreq.db file
4.          get a XML file from a RSS source;
5           while there is a new item to not fetched in the XML:
6.              if there is new link to not fetched:
7.                  generate information that include the link, title, date, and name
                      in a record file;
8.                  store a whole webpage in a HTML file in terms of the new link;
9.                  calculate words frequency in terms of the new webpage's title;
10.         if there are new links to fetched in above loop:
11.             update all new links into a record file;
12.             update words frequency information into a record file ,
                    RSSName_wordsfreq.dbt;
13.             generate words frequency record output as format
                 "date \t source \t words \t frequency" to visualize news words trends;
```

The code is shared in Github repository , that is, https://github.com/wing-nus/RSScrawler-1 .