Events as an example

Summary

|  |  |  |
| --- | --- | --- |
|  | Source | Code/Function |
| 1.Query | 1. Events2.0 on Google BigQuery (Run the code on the right in the following website)  <https://bigquery.cloud.google.com/table/gdelt-bq:full.events?pli=1>  You will firstly need to sign up a google cloud account and active it first, then search ‘BigQuery Data Transfer API’ in the console, then click ‘ENABLE’:    After that then click the link provided above.  2.On the left of the webpage, click on the downward arrow sign, click on “Create new dataset”(If you didn’t have one before), then click on “Save as Table” right above the data frame shown. The window shows like this:     1. Click ‘View Files’ and follow the instruction to create a new bucket(if didn’t have one before) 2. Write the url like the example in the picture:   ‘bucketname/foldername(if has)/datafilename’  eg:Mybucket/201501.csv | SELECT \* FROM [gdelt-bq:gdeltv2.events\_partitioned] #WHERE \_PARTITIONTIME >= "2015-03-01 00:00:00" AND \_PARTITIONTIME < "2016-04-01 00:00:00" where (MonthYear>=201503 and MonthYear<201604) and (Actor1CountryCode like "ARG" or Actor1Geo\_CountryCode like "AR" or Actor1Geo\_FullName like "%Argentina%" or Actor2Geo\_FullName like "%Argentina%" or Actor2CountryCode like "ARG" or Actor2Geo\_CountryCode like "AR" or Actor1Code like "ARG" or Actor1Name like "ARGENTINA" or Actor2Code like "ARG" or Actor2Name like "ARGENTINA" or ActionGeo\_FullName like "%Argentina%" or ActionGeo\_CountryCode like"AR") |
| 2.Data downloading and cleaning | 1.If there are lots of lines in the datafile we should seperate them by month(also we can say by column ‘MonthYear’) in order to use multiprocessing step later | sort\_by\_month.ipynb |
| 2.Load the csv datafile and save them into pickle file, use multiprocessing code code\_for\_mp.ipynb: Remember to change the date in list ‘Date\_file’, and the target and args if needed  Remember to save these files in the same folder | PickleCleaning.py  code\_for\_mp.ipynb |
| 3.pick 3 days you like, get its csv file and save it to pickle as in Step 2, and use code text\_downloading.ipynb to get the language list, then count percentage and number of each language(this part didn’t have code till now, you can write your own one).  Based on the percentage, choose 2 or 3 language(maybe more in the future) to translate | text\_downloading.ipynb  l |
| 4.downloading all the text of the news and set the language of the text | pickletext\_downloading.py |
| 5.mapping with English triggerlist (this part would be done on the big computer, don’t need to do it on your local computer) | Triggerlist:  triggerlist-english-all.txt |
| 3.Translate the triggerlist | 1.Translate the English trigger list into the other languages: translate the part after ‘AND+’ or ‘TEXT+’, word by word, phrase by phrase, because this is a dictionary not article  For each word or phrase find the most frequency translation using google translate:https://translate.google.com/?hl=en&tab=TT  keep all of the translation of which has a frequency level 3 or 2, the frequency shows as marked below:  permutation and combination, create the new dictionary(the format is dictionary)  copy the translation into the ‘triggerlist\_construction\_translated.ipynb’, and use ‘#’ to comment lines don’t delete it, because they will be used in the next part. | Triggerlist:  triggerlist-english-all.txt  Examples:  triggerlist-Spanish-extend-co-f.txt  triggerlist\_construction\_translated.ipynb |
| 2.Choose several date with the most mapping frequency of English(the day with the biggest result of the column ‘NumNewsFullLan’ in Step2-4), mapping these days’ news with the language triggerlist we translated, get the taxonomy of the news of the language we translate |  |
| 3.Coordinate translated triggerlist with the values frequency we get: Keep the translations have mapped, or keep all if no one mapped |  |
| 4.Mapping the triggerlist with the news text and get the result | Mapping with the coordinated triggerlist we got in Step3-3, using the code in Step2-5, return the csv file we want |  |
| 5.Visualization | 1.Use the code to convert the daily result into weekly(7 days moving average) | result\_combination.ipynb |
| 2.then convert them into Tableau format | convert\_Tableau.ipynb |
| 3.Visualize by Tableau |  |
| 4.get the wordcloud for each peak day | wc, bigrams, trigrams\_Final.ipynb |

1.Query

get the original data from database, need to notice that there maybe some country use abbr and other extension like ‘Argentina Government’, so we need to use ‘like’ in sql

2.Data downloading and cleaning

Sort and split by 'MonthYear' first in order to use multiprocessing later, and then remove the duplicate urls. Then do the text downloading. In the meanwhile, detect the data size and language percentages of the events.

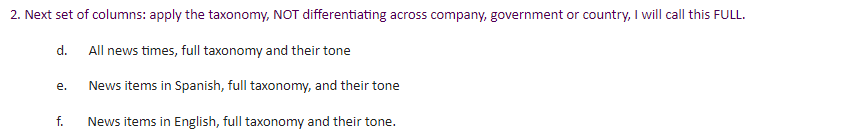
And mapping the text with English trigger list. First, use triggerlist to get the taxonomy of each news(use function’read\_trigger\_config’ and ‘filter\_stories\_bypickle’ in events\_pickle8.py). Then save the news list as a new pickle file. Second, count the daily avgtone, numnews, std using countNumAndMean\_new1.ipynb and countStdAndCv\_new1.ipynb, output as a csv file, the head format of the result is like this:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Date | TotalNum | NumToneAllAll | MeanToneAllAll | StdToneAllAll | NumToneAllSpan | MeanToneAllSpan | StdToneAllSpan | NumToneAllEng | MeanToneAllEng | StdToneAllEng | NumToneFullLan | MeanToneFullLan | StdToneFullLan | NumToneFullSpan | MeanToneFullSpan | StdToneFullSpan | NumToneFullEng | MeanToneFullEng | StdToneFullEng | NumToneCmpLan | MeanToneCmpLan | StdToneCmpLan | NumToneCmpSpan | MeanToneCmpSpan | StdToneCmpSpan | NumToneCmpEng | MeanToneCmpEng | StdToneCmpEng | NumToneCtyLan | MeanToneCtyLan | StdToneCtyLan | NumToneCtySpan | MeanToneCtySpan | StdToneCtySpan | NumToneCtyEng | MeanToneCtyEng | StdToneCtyEng | NumToneGovLan | MeanToneGovLan | StdToneGovLan | NumToneGovSpan | MeanToneGovSpan | StdToneGovSpan | NumToneGovEng | MeanToneGovEng | StdToneGovEng |

The algorithm of each line is in the ipynb file.

The name rule is as follows:

1. Date, TotalNum
2. All, Full taxonomy and their tone, News in different languages, full taxonomy and their tone. Like:



3.Translate the triggerlist

Translate the triggerlist into difference language so we can use it for next step mapping. The key of the translated triggerlist is like ‘triggerlist-Spanish-all.txt’. Follow the steps then get the cordinate triggerlist we use in steps later.

4.Mapping the triggerlist with the news text and get the result

Using all the triggerlist we want to use and the code in Step2-5, then get the result file similar like the head in Step2.

5.Visualization

Convert the daily result in to 7 days moving average, then convert them into the Tableau format so we can get the visualization by Tableau. Besides that, we need to find the peak days(count by the column ‘AvgToneFullLan’), algorithm is in the code. Find 6-10 peak days, then do the wordcloud with code.