clustering_catégorisation

April 11, 2021

[1]:
```
#Modification du dossier par défaut
import os
os.chdir('/Users/macbookair/Desktop/categorization/dataset')
#import file
import xlrd
```

Importation des librairies necessaires

[63]:
```
import numpy as np
import pandas as pd
import nltk
from nltk.stem import PorterStemmer
from nltk.stem import LancasterStemmer
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import word_tokenize
import re
import urllib
import io
from urllib.parse import urlencode, quote_plus
from urllib.request import Request, urlopen
import gzip
import json
from collections import Counter
import nltk
from nltk.corpus import webtext
from nltk.collocations import BigramCollocationFinder
from nltk.metrics import BigramAssocMeasures
```

0.0.1 CLUSTERING DES DOCUMENTS

Chargement des données

- Dans ce jeux de données on a effectué une transformation du dataframe de base de maniére à avoir les mots clés des articles sur une colonne dénommée keyword.
- Ce qui nous donne le dataset chargé ci-dessous.

[64]:
```
dataset = pd.read_csv('datset2.csv',encoding='latin_1',sep = ',')
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   article_title  300 non-null    object
 1   keyword        300 non-null    object
dtypes: object(2)
memory usage: 4.8+ KB
```

[65]:
```
dataset.head(5)
```

[65]:
```
                                        article_title  \
0  17.5 An intrinsically linear wideband digital …
1  3D Inception U-Net for Aorta Segmentation usin…
2  A 20.5TOPS and 217.3GOPS/mm<sup>2</sup> Multic…
3      A Birth-Death Process for Feature Allocation.
4  A Classroom Deployment of a Haptic System for …

                                             keyword
0  intrinsically linear wideband digital polar pa…
1  3d inception net segmentation using computed t…
2  mm sup multicore soc dnn accelerator image sig…
3              birth death process feature allocation
4  classroom deployment haptic system learning ce…
```

Transformation du dataframe

- Ici nous transformons notre dataframe en dicttionnaire pour faciliter les opération suivantes.

[66]:
```
dict_df = dataset.to_dict()
```

- On recupéres les clés(titres) et les valeurs(keywords) sous forme de dictionaires

[67]:
```
#cles, vals = zip(*dict_df.items())
cles = list(dict_df.keys())
vals = list(dict_df.values())
```

[ ]:
```
print(cles)
print(vals)
print(vals[1])
```

[70]:
```
cles0, vals0 = list(vals[0].keys()),list(vals[0].values())
cles1, vals1 = list(vals[1].keys()),list(vals[1].values())
```

- On recupéres les clés(titres) et les valeurs(keywords) sous forme de liste

[72]:
```
titres, keywords = list(vals[1].keys()),list(vals[1].values())
```

Création de la matrice document-termes

- utilisation de la librairie sklearn
- ici les lignes représente les articles les colonnes les mots clés

[73]:
```
import pandas as pd
from sklearn.feature_extraction.text import CountVectorizer

vec = CountVectorizer()
X = vec.fit_transform(keywords)
df = pd.DataFrame(X.toarray(), columns=vec.get_feature_names())
df['document'] = list(vals[0].keys())
df = df.set_index('document')
```

- Affichage de quelques lignes et quelques colonnes de la matrice

[80]:
```
df[['3d','absolute','abstract','abstraction','abstractive','abuse','word','words','workers']].
 ↪head()
```

[80]:

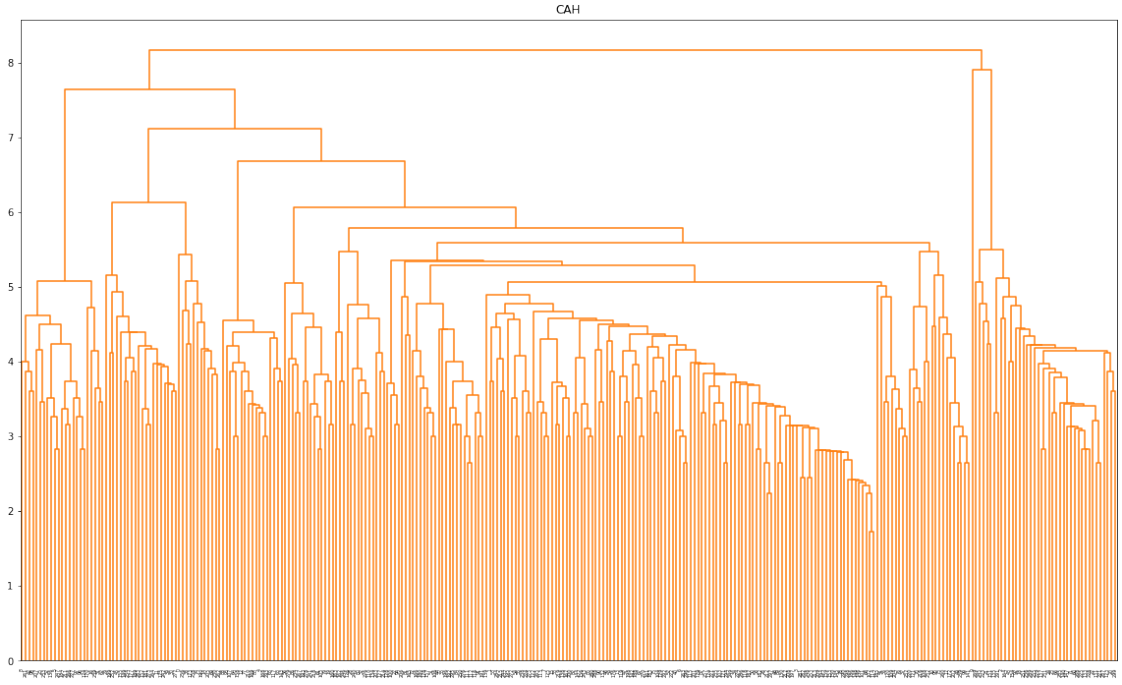
document	3d	absolute	abstract	abstraction	abstractive	abuse	word	words	workers
0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0

Réalisation du clustering

- Ici nou générons la matrice des liens Z avec linkage
- Génération et affichage du dendrogramme
- Affichages du nombre de clusters obtenu selon le niveau de coupure t=5

[32]:
```
#librairies pour la CAH
from matplotlib import pyplot as plt
from scipy.cluster.hierarchy import dendrogram, linkage
import scipy.cluster.hierarchy as sch

#générer la matrice des liens
#Z = linkage(df,method='single',metric='jaccard')
Z = linkage(df,method='ward',metric='euclidean')
```

```
# génération et affichage du dendrogramme
plt.figure(figsize=(20,12))
plt.title("CAH")

dendrogram(Z,labels=list(vals[0].keys()),color_threshold=100)

plt.show()

groupes_cah = sch.fcluster(Z,t=5,criterion='distance')
print(np.unique(groupes_cah).size, "groupes constitués")
```




```
26 groupes constitués
```

- Nous remarquons 26 clusters avec un tel niveau de coupure

Construction du dataframe avec les clusters

- Ici on crée un dataframe des individus et leurs classes d'appartenance

[34]:
```
#index triés des groupes
import numpy as np
idg = np.argsort(groupes_cah)
```

- affichage des observations et leurs groupes

[81]:
```
#affichage des observations et leurs groupes
dataf = pd.DataFrame(df.index[idg],groupes_cah[idg])
groupes = dataf
groupes['groupe'] = groupes.index
groupes.
```

[81]:

	document	groupe
1	261	1
1	227	1
1	211	1
1	207	1
1	25	1
..	…	…
25	255	25
25	268	25
25	22	25
25	299	25
26	9	26

[300 rows x 2 columns]

- Comptage du nombre d'individus par cluster

[ ]:
```
#nombre d'individus par clusters
groupes['groupe'].value_counts()
```

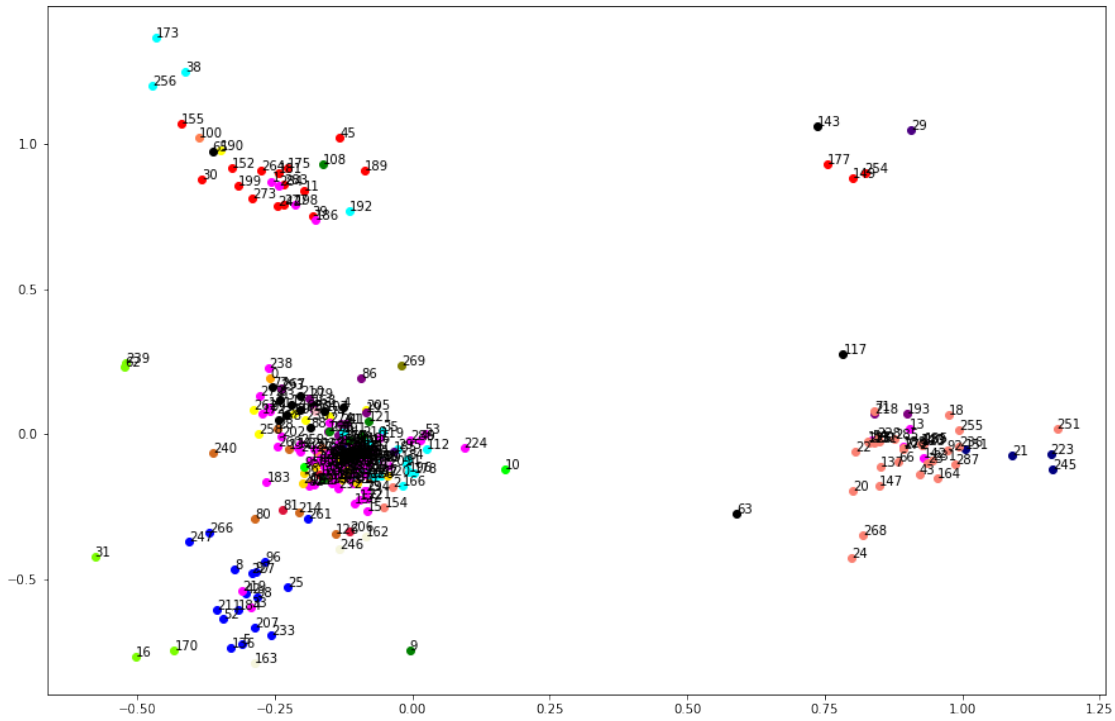
Affichage des groupes sur le plan factoriel

- Ici nous allons visualiser nos 26 clusters sur un plan factoriel
- avec un code couleur pour chaque groupe.

[82]:
```
colors = ['blue','lawngreen','red','indigo', 'aqua',␣
↪'yellow','orange','black','purple','pink',
         ␣
↪'beige','chocolate','coral','crimson','cyan','fuchsia','gold','indigo','green','lime',
          'magenta','navy','olive','plum','salmon','green']

numbers = [1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26]
```

[39]:
```
from sklearn.decomposition import PCA
#ACP
acp_subset = PCA(n_components=2).fit_transform(df)
#projeter dans le plan factoriel
#avec un code couleur selon le groupe
#remarquer le rôle de zip()
plt.figure(figsize=(15,10))
for couleur,k in zip(colors,numbers):
```

```
    plt.
↪scatter(acp_subset[groupes_cah==k,0],acp_subset[groupes_cah==k,1],c=couleur)

#mettre les labels des points
#remarquer le rôle de enumerate()
for i,label in enumerate(df.index):
    plt.annotate(label,(acp_subset[i,0],acp_subset[i,1]))

plt.show()
```



[83]:
```
# articles de chaque cluster ou groupe
groupes.groupby(['groupe','document']).count()
```

[83]:
```
Empty DataFrame
Columns: []
Index: [(1, 5), (1, 8), (1, 25), (1, 42), (1, 52), (1, 96), (1, 97), (1, 98),
(1, 124), (1, 136), (1, 184), (1, 207), (1, 211), (1, 227), (1, 233), (1, 247),
(1, 261), (1, 266), (2, 16), (2, 31), (2, 62), (2, 170), (2, 239), (3, 11), (3,
30), (3, 39), (3, 45), (3, 149), (3, 152), (3, 155), (3, 175), (3, 177), (3,
181), (3, 189), (3, 199), (3, 242), (3, 252), (3, 254), (3, 264), (3, 273), (3,
277), (3, 283), (4, 29), (5, 38), (5, 173), (5, 256), (6, 14), (6, 69), (6, 74),
(6, 190), (6, 205), (6, 230), (6, 258), (6, 262), (7, 0), (8, 4), (8, 12), (8,
63), (8, 64), (8, 65), (8, 77), (8, 83), (8, 88), (8, 93), (8, 107), (8, 117),
```

```
(8, 143), (8, 158), (8, 168), (8, 169), (8, 210), (8, 293), (9, 19), (9, 61),
(9, 79), (9, 86), (9, 105), (9, 160), (9, 193), (9, 218), (9, 234), (9, 241),
(9, 267), (9, 278), (9, 279), (10, 56), (11, 162), (11, 163), (11, 246), (12,
28), (12, 32), (12, 80), (12, 110), (12, 126), (12, 127), (12, 134), (12, 203),
(12, 214), (12, 226), (12, 240), …]

[300 rows x 0 columns]
```

Reconstitution du dataframe de base avec les clusters de chaque document

- On crée la colonne titres et keywords sous forme de liste
- en interrogeant le dataframe de base(dataset) et le dataframe groupes

[41]:
```
elements = list(groupes.document)
titres=[]
keywords = []
for i in elements:
    rowData = dataset.loc[i, : ]
    titres.append(rowData['article_title'])
    keywords.append(rowData['keyword'])
```

[84]:
```
# création de la colonne titre et keywords sur groupes
groupes['titre_article'], groupes['keywords'] = titres, keywords
```

- Affichage du dataframe reconstitué

[87]:
```
groupes.head()
```

[87]:

	document	groupe	titre_article	\
1	261	1	0	
1	227	1	1	
1	211	1	2	
1	207	1	3	
1	25	1	4	

	keywords
1	intrinsically linear wideband digital polar pa…
1	3d inception net segmentation using computed t…
1	mm sup multicore soc dnn accelerator image sig…
1	birth death process feature allocation
1	classroom deployment haptic system learning ce…

0.0.2 LABELISATION DES ARTICLES D'UN CLUSTER

Cas ou on se base sur les 10 premiers mots clés les plus recurrents des articles du cluster
Maitenant que nos clusters de documents sont construits, on va devoir attaquer la partie etiquettage. Le pricincipe utilisés est bien expliqué dans l'article. - Regroupper l'ensemble des keywords du clusters - Prendre les top 10 de keywords plus recurents - Construire des n-grammes(n=1…3) sur ces 10 mots clés - Interroger l'API Babelnet pour extraire les synsets, categories et domains.

Pour des contraintes techniques avec babelnet nous allons pas pouvoir executer le code sur l'ensemble des clusters en meme temps, car le nombre de requetes qu'on peut faire est limité(1000) pour la licence dont nous disposons. Du coup on va adopter une approche semi-automatique

- Cas du clusters 1;

[89]:
```
cluster = 1
cluster_kwlist = pd.DataFrame(groupes.groupby('groupe')['keywords'].apply(' '.
 ↪join))
cluster_kwlist_group = cluster_kwlist[cluster_kwlist.index==cluster]
```

[92]:
```
cluster_kwlist.head(10)
```

[92]:

groupe	keywords
1	intrinsically linear wideband digital polar pa…
2	simulation modeling framework autonomous vehic…
3	motion sensing based assistive system design i…
4	adaptive detection prevention architecture tra…
5	apnea detection method based correlation analy…
6	approximating maximum weighted decomposable gr…
7	automatic computation fundamental matrix based…
8	automatic adaptive approximation stencil compu…
9	communication avoiding neumann expansion metho…
10	cyber attacks impact security prevention modal…

[93]:
```
## Keywords du clusters 1
cluster_kwlist_group
```

[93]:

groupe	keywords
1	intrinsically linear wideband digital polar pa…

[94]:
```
# comptage du nombre d'occurences par mots clés
split_it = cluster_kwlist_group['keywords'][cluster].split()
Counter = Counter(split_it)
nbr_mots = 10
most_occur = Counter.most_common(nbr_mots)
mots_mo = ' '.join([x[0] for x in most_occur])
```

[95]:
```
# Fonction de création des n-grammes
def ngrames(sentence,n):
    ngramms = []
    ngrm = nltk.ngrams(sentence.split(), n)
    for grm in ngrm:
        ngramms.append(list(grm))
    return ngramms
```

Extraction des synonymes, catégories et dommaines

[96]:
```
# Initialisation
cat_all = {'cat':[],'mot':[]}
cat_all = pd.DataFrame(data=cat_all)
domain_all = {'domains':[],'ratio':[],'mot':[]}
domain_all = pd.DataFrame(data=domain_all)
ngrammes = [3,2,1]
for k in ngrammes:

    for i in ngrames(mots_mo, k):
        i = ' '.join(i)
        print(i)
        service_url = 'https://babelnet.io/v6/getSynsetIds'

        params = {
                'lemma' : i,
                'searchLang' : 'EN',
                'key'  : '53cc96fc-8b88-4ab4-8af8-b4e2870bac46'
        }

        url = service_url + '?' + urlencode(params)
        request = Request(url)
        request.add_header('Accept-encoding', 'gzip')
        response = urlopen(request)

        if response.info().get('Content-Encoding') == 'gzip':
            buf = io.BytesIO(response.read())
            f = gzip.GzipFile(fileobj=buf)
            data_ids = json.loads(f.read())

        ids = [d['id'] for d in data_ids]

        cat = []
        domains = []
        ratio = []

        for j in ids:
            service_url = 'https://babelnet.io/v6/getSynset'

            params = {
            'id' : j,
            'key'  : '53cc96fc-8b88-4ab4-8af8-b4e2870bac46'
            }

            url = service_url + '?' + urlencode(params)
            request = Request(url)
            request.add_header('Accept-encoding', 'gzip')
```

```
            response = urlopen(request)

            if response.info().get('Content-Encoding') == 'gzip':
                buf = io.BytesIO(response.read())
                f = gzip.GzipFile(fileobj=buf)
                data_cat = json.loads(f.read())
                cat = cat + data_cat['categories']
                domains=domains+list(data_cat['domains'].keys())
                ratio = ratio+list(data_cat['domains'].values())

        cat_art = [d['category'] for d in cat]
        domains_art = [d for d in domains]
        ratio_mot = [r for r in ratio]

        kw1 = [i]*len(cat_art)
        kw2 = [i]*len(domains_art)
        kw3 = [i]*len(ratio)

        df2 = {'cat':cat_art,'mot':kw1}
        df2 = pd.DataFrame(data=df2)
        cat_all = cat_all.append(df2)

        dfd = {'domains':domains_art,'ratio': ratio_mot, 'mot':kw2}
        dfd = pd.DataFrame(data=dfd)
        domain_all = domain_all.append(dfd)

occ1 = pd.crosstab(cat_all['cat'],cat_all['mot'])
occ2 = pd.crosstab(domain_all['domains'],domain_all['mot'])
```

```
approach image analysis
image analysis networks
analysis networks algorithm
networks algorithm control
algorithm control using
control using learning
using learning neural
learning neural deterministic
approach image
image analysis
analysis networks
networks algorithm
algorithm control
control using
using learning
learning neural
neural deterministic
approach
image
```

```
analysis
networks
algorithm
control
using
learning
neural
deterministic
```

0.0.3 Affichage des resultats

- Affichages des 10 premiers catégories obtenues

[98]:
```
cat_all.head(10)
```

[98]:
```
                              cat             mot
0  Articles_with_short_description  image analysis
1                  Formal_sciences  image analysis
2                  Computer_vision  image analysis
0                      2006_albums        approach
1      Von_Hertzen_Brothers_albums        approach
2              Cricket_terminology        approach
3                Bowling_(cricket)        approach
4    Cricket_captaincy_and_tactics        approach
0               Types_of_functions           image
1                Mathematics_stubs           image
```

- Affichages des 10 premiers domaines obtenues

[99]:
```
domain_all.head(10)
```

[99]:
```
                              domains     ratio             mot
0           MATHEMATICS_AND_STATISTICS  0.455841  image analysis
0     RELIGION_MYSTICISM_AND_MYTHOLOGY  0.424831        approach
1                PHYSICS_AND_ASTRONOMY  5.000000        approach
2                 TRANSPORT_AND_TRAVEL  5.000000        approach
3             LANGUAGE_AND_LINGUISTICS -2.000000        approach
..                                 …       …              …
2                EDUCATION_AND_SCIENCE  5.000000        learning
0                  HEALTH_AND_MEDICINE  5.000000          neural
1                  HEALTH_AND_MEDICINE  1.000000          neural
2               LITERATURE_AND_THEATRE -2.000000          neural
0   PHILOSOPHY_PSYCHOLOGY_AND_BEHAVIOR  5.000000   deterministic

[103 rows x 3 columns]
```

- Croisement des catégories et des mots clés

[101]:
```
occ1.head(10)
```

[101]:
mot cat	algorithm	analysis	approach	\
1941_births	0	0	0	
1960s_American_satirical_television_series	0	0	0	
1960s_American_sitcoms	0	0	0	
1960s_British_film_stubs	0	0	0	
1965_American_television_series_debuts	0	0	0	
1967_births	0	0	0	
1969_films	0	0	0	
1970_American_television_series_endings	0	0	0	
1970_radio_programme_debuts	0	1	0	
1970s_American_mystery_television_series	0	0	0	
1970s_American_satirical_television_series	0	0	0	
1970s_American_sitcoms	0	0	0	
1970s_erotic_drama_films	0	0	0	
1970s_pornographic_films	0	0	0	
1975_drama_films	0	0	0	
1975_films	0	0	0	
1980s_Italian_film_stubs	0	0	0	
1980s_thriller_novel_stubs	0	0	0	
1981_births	0	0	0	
1982_American_novels	0	0	0	

mot cat	control	deterministic	image	\
1941_births	0	0	1	
1960s_American_satirical_television_series	1	0	0	
1960s_American_sitcoms	1	0	0	
1960s_British_film_stubs	0	0	1	
1965_American_television_series_debuts	1	0	0	
1967_births	1	0	0	
1969_films	0	0	1	
1970_American_television_series_endings	1	0	0	
1970_radio_programme_debuts	0	0	0	
1970s_American_mystery_television_series	1	0	0	
1970s_American_satirical_television_series	1	0	0	
1970s_American_sitcoms	1	0	0	
1970s_erotic_drama_films	0	0	1	
1970s_pornographic_films	0	0	1	
1975_drama_films	0	0	1	
1975_films	0	0	1	
1980s_Italian_film_stubs	1	0	0	
1980s_thriller_novel_stubs	1	0	0	
1981_births	1	0	0	
1982_American_novels	1	0	0	

mot	image analysis	learning	neural	\

cat			
1941_births	0	0	0
1960s_American_satirical_television_series	0	0	0
1960s_American_sitcoms	0	0	0
1960s_British_film_stubs	0	0	0
1965_American_television_series_debuts	0	0	0
1967_births	0	0	0
1969_films	0	0	0
1970_American_television_series_endings	0	0	0
1970_radio_programme_debuts	0	0	0
1970s_American_mystery_television_series	0	0	0
1970s_American_satirical_television_series	0	0	0
1970s_American_sitcoms	0	0	0
1970s_erotic_drama_films	0	0	0
1970s_pornographic_films	0	0	0
1975_drama_films	0	0	0
1975_films	0	0	0
1980s_Italian_film_stubs	0	0	0
1980s_thriller_novel_stubs	0	0	0
1981_births	0	0	0
1982_American_novels	0	0	0

mot	using
cat	
1941_births	0
1960s_American_satirical_television_series	0
1960s_American_sitcoms	0
1960s_British_film_stubs	0
1965_American_television_series_debuts	0
1967_births	0
1969_films	0
1970_American_television_series_endings	0
1970_radio_programme_debuts	0
1970s_American_mystery_television_series	0
1970s_American_satirical_television_series	0
1970s_American_sitcoms	0
1970s_erotic_drama_films	0
1970s_pornographic_films	0
1975_drama_films	0
1975_films	0
1980s_Italian_film_stubs	0
1980s_thriller_novel_stubs	0
1981_births	0
1982_American_novels	0

- Croisement des domaines et des mots clés

[102]:
```
occ2
```

[102]:

mot domains	algorithm	analysis	approach	control
ART_ARCHITECTURE_AND_ARCHAEOLOGY	0	0	0	0
BIOLOGY	0	0	0	1
BUSINESS_INDUSTRY_AND_FINANCE	0	0	1	1
CHEMISTRY_AND_MINERALOGY	0	1	0	0
COMPUTING	1	0	0	2
CRAFT_ENGINEERING_AND_TECHNOLOGY	0	1	0	1
CULTURE_ANTHROPOLOGY_AND_SOCIETY	0	0	0	2
EDUCATION_AND_SCIENCE	0	0	0	2
EMOTIONS_AND_FEELINGS	0	0	1	1
HEALTH_AND_MEDICINE	1	0	0	0
LANGUAGE_AND_LINGUISTICS	0	1	1	1
LITERATURE_AND_THEATRE	0	1	0	1
MATHEMATICS_AND_STATISTICS	1	2	1	2
MEDIA_AND_PRESS	0	0	0	6
MUSIC_SOUND_AND_DANCING	0	0	1	16
PHILOSOPHY_PSYCHOLOGY_AND_BEHAVIOR	0	3	0	4
PHYSICS_AND_ASTRONOMY	0	0	1	0
RELIGION_MYSTICISM_AND_MYTHOLOGY	0	0	1	1
SEX	0	0	1	0
SPACE_AND_TOUCH	0	0	1	0
SPORT_GAMES_AND_RECREATION	0	0	2	0
TIME	0	0	1	0
TRANSPORT_AND_TRAVEL	0	0	2	0

mot domains	deterministic	image	image analysis
ART_ARCHITECTURE_AND_ARCHAEOLOGY	0	5	0
BIOLOGY	0	0	0
BUSINESS_INDUSTRY_AND_FINANCE	0	0	0
CHEMISTRY_AND_MINERALOGY	0	0	0
COMPUTING	0	2	0
CRAFT_ENGINEERING_AND_TECHNOLOGY	0	0	0
CULTURE_ANTHROPOLOGY_AND_SOCIETY	0	1	0
EDUCATION_AND_SCIENCE	0	0	0
EMOTIONS_AND_FEELINGS	0	0	0
HEALTH_AND_MEDICINE	0	1	0
LANGUAGE_AND_LINGUISTICS	0	1	0
LITERATURE_AND_THEATRE	0	2	0
MATHEMATICS_AND_STATISTICS	0	5	1
MEDIA_AND_PRESS	0	3	0
MUSIC_SOUND_AND_DANCING	0	1	0
PHILOSOPHY_PSYCHOLOGY_AND_BEHAVIOR	1	4	0
PHYSICS_AND_ASTRONOMY	0	1	0

```
RELIGION_MYSTICISM_AND_MYTHOLOGY                0      0                0
SEX                                             0      0                0
SPACE_AND_TOUCH                                 0      0                0
SPORT_GAMES_AND_RECREATION                      0      1                0
TIME                                            0      0                0
TRANSPORT_AND_TRAVEL                            0      0                0
```

mot	learning	neural	using
domains			
ART_ARCHITECTURE_AND_ARCHAEOLOGY	0	0	0
BIOLOGY	0	0	0
BUSINESS_INDUSTRY_AND_FINANCE	0	0	0
CHEMISTRY_AND_MINERALOGY	0	0	0
COMPUTING	0	0	0
CRAFT_ENGINEERING_AND_TECHNOLOGY	0	0	0
CULTURE_ANTHROPOLOGY_AND_SOCIETY	0	0	0
EDUCATION_AND_SCIENCE	2	0	0
EMOTIONS_AND_FEELINGS	0	0	0
HEALTH_AND_MEDICINE	0	2	0
LANGUAGE_AND_LINGUISTICS	0	0	0
LITERATURE_AND_THEATRE	0	1	0
MATHEMATICS_AND_STATISTICS	0	0	0
MEDIA_AND_PRESS	0	0	0
MUSIC_SOUND_AND_DANCING	1	0	0
PHILOSOPHY_PSYCHOLOGY_AND_BEHAVIOR	0	0	1
PHYSICS_AND_ASTRONOMY	0	0	0
RELIGION_MYSTICISM_AND_MYTHOLOGY	0	0	0
SEX	0	0	0
SPACE_AND_TOUCH	0	0	0
SPORT_GAMES_AND_RECREATION	0	0	0
TIME	0	0	0
TRANSPORT_AND_TRAVEL	0	0	0

Cas ou on se base sur les mots clés d'un articles du clusters

- Ici on chosit le top 1 des articles de chaque clusters
- On recupére les mots clés de l'article
- On constitue les n-grammes à partir du mots clés
- On intérroge BabelNet Api avec les trigrammes qui a tendance à données des domaines exactes et uniques
- S'il ne retourne pas de resultats, on l'interroge avec les bigrammes, sinon avec les mots clés de l'articles pris un par un.

[111]:
```
# Recuperation des top1 article de chaque clusters(26 clusters ---> 26 articles)
top1_groupe = groupes.groupby(['groupe'])['groupe','titre_article','keywords'].
 ↪apply(lambda x: x.nlargest(1, columns=['groupe']))
```

```
top1_groupe = top1_groupe[['groupe','titre_article','keywords']].
 ↪set_index(top1_groupe['groupe'])
top1_groupe[['titre_article','keywords']]
```

```
<ipython-input-111-3f77f0acc273>:2: FutureWarning: Indexing with multiple keys
(implicitly converted to a tuple of keys) will be deprecated, use a list
instead.
  top1_groupe =
groupes.groupby(['groupe'])['groupe','titre_article','keywords'].apply(lambda x:
x.nlargest(1, columns=['groupe']))
```

[111]:

groupe	titre_article	keywords
1	0	intrinsically linear wideband digital polar pa…
2	18	simulation modeling framework autonomous vehic…
3	23	motion sensing based assistive system design i…
4	42	adaptive detection prevention architecture tra…
5	43	apnea detection method based correlation analy…
6	46	approximating maximum weighted decomposable gr…
7	54	automatic computation fundamental matrix based…
8	55	automatic adaptive approximation stencil compu…
9	72	communication avoiding neumann expansion metho…
10	85	cyber attacks impact security prevention modal…
11	86	data driven modeling simulation daily activity…
12	89	deep hashing triplet quantization loss
13	100	digital map using augmented reality smart devi…
14	104	discovering typical entities multi timeline su…
15	107	learning channel really matter insights commer…
16	127	exploring performance children writing chinese…
17	234	revenue oriented air quality prediction micros…
18	242	secure trustable electronic medical records sh…
19	243	security requirements engineering framework cy…
20	249	simultaneous geometric radiometric calibration…
21	251	space time frequency mimo relaying system rece…
22	260	synchronization tracking class uncertain nonid…
23	264	impact digitization product using direct digit…
24	265	influence online academic information search s…
25	267	social construction personal data protection s…
26	299	scalable distributed kernel support vector mac…

[ ]:
```
# Recuperation des mots clés du clusters
keywords = top1_groupe.loc[1,:]['keywords']
```

[ ]:
```
trigrams = ngrames(keywords, 3)

cat_all = {'cat':[],'mot':[]}
cat_all = pd.DataFrame(data=cat_all)
```

```
domain_all = {'domains':[],'ratio':[],'mot':[]}
domain_all = pd.DataFrame(data=domain_all)
corbeill = ['MUSIC_SOUND_AND_DANCING',' MEDIA_AND_PRESS']

for i in trigrams:
    i = ' '.join(i)
    print(i)
    service_url = 'https://babelnet.io/v6/getSynsetIds'

    params = {
            'lemma' : i,
            'searchLang' : 'EN',
            'key'  : '1ed7054d-95e6-46f3-a86a-de79d654fb23'
    }

    url = service_url + '?' + urlencode(params)
    request = Request(url)
    request.add_header('Accept-encoding', 'gzip')
    response = urlopen(request)

    if response.info().get('Content-Encoding') == 'gzip':
        buf = io.BytesIO(response.read())
        f = gzip.GzipFile(fileobj=buf)
        data_ids = json.loads(f.read())


    ids = [d['id'] for d in data_ids]
    cat = []
    domains = []
    ratio = []
    for j in ids:
        service_url = 'https://babelnet.io/v6/getSynset'

        params = {
        'id' : j,
        'key'  : '1ed7054d-95e6-46f3-a86a-de79d654fb23'
        }

        url = service_url + '?' + urlencode(params)
        request = Request(url)
        request.add_header('Accept-encoding', 'gzip')
        response = urlopen(request)

        if response.info().get('Content-Encoding') == 'gzip':
            buf = io.BytesIO(response.read())
            f = gzip.GzipFile(fileobj=buf)
            data_cat = json.loads(f.read())
```

```
            cat = cat + data_cat['categories']
            domains=domains+list(data_cat['domains'].keys())
            ratio = ratio+list(data_cat['domains'].values())

    cat_art = [d['category'] for d in cat]
    domains_art = [d for d in domains]
    ratio_mot = [r for r in ratio]

    kw1 = [i]*len(cat_art)
    kw2 = [i]*len(domains_art)
    kw3 = [i]*len(ratio)

    df2 = {'cat':cat_art,'mot':kw1}
    df2 = pd.DataFrame(data=df2)
    cat_all = cat_all.append(df2)

    dfd = {'domains':domains_art,'ratio': ratio_mot, 'mot':kw2}
    dfd = pd.DataFrame(data=dfd)
    domain_all = domain_all.append(dfd)

occ1 = pd.crosstab(cat_all['cat'],cat_all['mot'])
occ2 = pd.crosstab(domain_all['domains'],domain_all['mot'])
```

[56]:
```
occ2
```

[56]:
```
mot                          critical path analysis
domains
MATHEMATICS_AND_STATISTICS                        1
```

[57]:
```
trigrams
```

[57]:
```
[['tackling', 'mobile', 'traffic'],
 ['mobile', 'traffic', 'critical'],
 ['traffic', 'critical', 'path'],
 ['critical', 'path', 'analysis'],
 ['path', 'analysis', 'passive'],
 ['analysis', 'passive', 'active'],
 ['passive', 'active', 'measurements']]
```

[58]:
```
domain_all
```

[58]:
```
                     domains  ratio                     mot
0  MATHEMATICS_AND_STATISTICS   -2.0  critical path analysis
```

[ ]:
```

```