Compte Rendu Web Mining et TAL

March 23, 2020

1 Mini Projet : Web Mining et Traitement Automatique de Langues

1.1 Auteur : Boubacar TRAORE & Zakaria Jarraya

[1]:
```
# Changement de la police utilisée et de sa taille
from IPython.core.display import HTML, display
from tp_tools import change_font, plot_history
display(HTML(change_font()))
```

```
<IPython.core.display.HTML object>
```

[2]:
```
from jyquickhelper import add_notebook_menu
add_notebook_menu()
```

[2]:
```
<IPython.core.display.HTML object>
```

1.2 Importation des librairies nécessaires

[3]:
```
import json
import time
import spacy
import numpy as np
import pandas as pd
import newspaper as npp
import feedparser as fp
import itertools as it

from pprint import pprint
from collections import defaultdict

from plotly.offline import iplot, init_notebook_mode
import plotly.graph_objs as go

init_notebook_mode(True)
```

1.3 1. Acquisition de données par écoute d'un flux RSS

1.3.1 1.1. Récupération des fichiers RSS

Q1.

[4]:
```
#Lecture de la base de données déjà existente
with open('data/france-info-20200301.json', 'r') as f:
    articles = json.load(f)
```

[5]:
```
#type et taille de la base lue
type(articles), len(articles)
```

[5]: (dict, 660)

[6]:
```
len(list(articles.items()))
```

[6]: 660

[7]:
```
# Affichage du premier élément du dictionnaire (le 1er article)
dict(list(articles.items())[0:1])
```

[7]:
```
{'https://www.francetvinfo.fr/sante/prevention/eure-quand-les-elus-apprennent-a-
se-defendre_3791635.html#xtor=RSS-3-[france]': {'title': "Violences contre les
maires : dans l'Eure, des élus apprennent à se défendre",
  'date': '2020-01-19T16:31:16',
  'author': [],
  'category': 'france',
  'content': 'Beaucoup de maires et de secrétaires de mairie ont rencontré au
cours de leurs mandats des problèmes de violence, essentiellement verbale.
```

```
C\'est le cas de Claire. Un jour, un homme est entré dans son bureau et a frappé
très fort sur son bureau pour l\'intimider. "J\'ai réagi à l\'instinct, j\'ai
essayé de discuter, mais ça s\'est envenimé. Heureusement qu\'une dame est
arrivée à la permanence et a menacé d\'appeler la gendarmerie", raconte Claire
Carrère-Godebout, maire (SE) de Graveron-Sémerville (Eure).\n\nDes professeurs
de karaté et un psychologue\n\nLa plupart de ces élus ne savent pas comment se
comporter face à une situation conflictuelle. Alors, l\'Association des maires
ruraux de l\'Eure les initie à l\'autodéfense, avec l\'aide de professeurs de
karaté. Et un psychologue leur a expliqué comment désamorcer un différent.
"C\'est nécessaire qu\'ils aient confiance en eux donc, à partir de là, je pense
que cet atelier ne peut avoir que des bénéfices", explique Christophe Micaux.',
  'image_link':
'https://www.francetvinfo.fr/image/75rt1bxrp-7049/1500/843/20799631.png'}}
```

Il s'agit d'un dictionnaire (venant d'un fichier json) contenant 660 clés associées chacune à des valeurs. Chaque clé correspond à l'URL d'un article extrait via le flux RSS. A chaque clé est associée une valeur qui est également un dictionnaire contenant comme information le titre, la date, l'auteur, la catégorie, le contenu et le lien image de l'article en question.

Passons à la collecte de nouveaux articles via les flux RSS.

[8]:
```
# Stockage des urls
urls = ['https://www.francetvinfo.fr/france.rss',
        'https://www.francetvinfo.fr/europe.rss',
        'https://www.francetvinfo.fr/entreprises.rss']
```

[9]:
```
urls[0].split('.rss')[0].split("/")[-1]
```

[9]:
```
'france'
```

Nous allons sélectionner le premier flux et jeter un coup d'oeil à son contenu.

[10]:
```
# on essaie le 1er feed
feed0 = fp.parse(urls[0])
```

[11]:
```
#overview of feed0
feed0.keys()
```

[11]:
```
dict_keys(['feed', 'entries', 'bozo', 'headers', 'etag', 'updated',
'updated_parsed', 'href', 'status', 'encoding', 'version', 'namespaces'])
```

Nous remarquons que ce flux retourne énormément d'informations. La clé 'entries' du dictionnaire correspond à la liste des articles trouvées. Nous pouvons également utiliser les informations annexes pour construire notre base de données.

Nous allons maintenant extraire tous les flux et identifier les nouveaux articles qui ne sont pas déjà présent dans la base json dont on dispose déjà.

[12]:
```
new_articles_id = []
for url in urls:
    feed = fp.parse(url)
    for article in feed.entries:
        if article.id not in list(articles.keys()):
            new_articles_id.append([article.id, article.title]) #Ajout du lien␣
↪du nouvel article
```

[13]:
```
new_articles = pd.DataFrame(new_articles_id, columns=["id", "title"])
new_articles.sample(3, random_state=42)
```

[13]:

	id	title
0	https://www.francetvinfo.fr/sante/maladie/coro…	Coronavirus : un retour à l'école le 4 mai est…
5	https://www.francetvinfo.fr/sante/maladie/coro…	Coronavirus : les hôpitaux en Île-de-France s'…
30	https://www.francetvinfo.fr/economie/industrie…	VIDEO. La filière de la chaussure portugaise m…

[14]:
```
# Voir les liens complets
new_articles.id.sample(3, random_state=42).tolist()
```

[14]:
```
['https://www.francetvinfo.fr/sante/maladie/coronavirus/coronavirus-un-retour-a-
l-ecole-le-4-mai-est-le-scenario-privilegie-par-le-
gouvernement_3880565.html#xtor=RSS-3-[france]',
 'https://www.francetvinfo.fr/sante/maladie/coronavirus/coronavirus-les-
hopitaux-en-ile-de-france-sorganisent-pour-faire-face-au-pic-de-
lepidemie_3879865.html#xtor=RSS-3-[france]',
 'https://www.francetvinfo.fr/economie/industrie/video-la-filiere-de-la-
chaussure-portugaise-mise-sur-la-qualite-le-design-le-service-et-le-mandarin-
pour-gagner-des-marches_3857591.html#xtor=RSS-3-[monde/europe]']
```

[15]:
```
len(new_articles_id)
```

[15]:
```
57
```

Nous pouvons constater qu'il y a 60 nouveaux articles tirés de ces 3 flux.

1.3.2 1.2. Récupéreation des articles

Q2. Regardons le code source du premier article affiché après l'extraction des articles des flux RSS : https://www.francetvinfo.fr/sante/maladie/coronavirus/coronavirus-des-aperos-au-balcon-pour-vaincre-l-ennui-du-confinement_3878493.html#xtor=RSS-3-[france]

Le code source de la page écite en HTML est très difficile à comprendre, les informations sont dispersées partout et on a du mal à s'y retrouver.

Le package newspaper3k permet de faire de l'extraction automatique des contenus des articles. Il permet facilement de dissocier le titre, le contenu et plein d'autres information sur les sites d'articles. Sa prise en main est très facile. Prenons l'exemple d'un nouvel arcticle qu'on vient d'extraire.

[16]:
```
# Exemple du premier article extrait
article = npp.Article(new_articles.id[0])
article.download()
article.parse()
article.text
```

[16]:
```
'Le ministre de l\'Education nationale, Jean-Michel Blanquer, a précisé que ce
calendrier restait dépendant de l\'évolution de l\'épidémie.\n\nUn retour des
élèves en classe le 4 mai est le "scénario privilégié" mais reste "tributaire de
l\'évolution de l\'épidémie" en France, a indiqué dimanche 22 mars le ministre
de l\'Education Jean-Michel Blanquer. "Ce n\'est pas une annonce car tout sera
conditionné à l\'évolution du coronavirus et nous appliquons ce que nous dit le
ministère de la Santé", a précisé l\'entourage du ministre auprès de
l\'AFP.\n\n> Suivez l\'évolution de la pandémie de Covid-19 dans notre
direct.\n\n"On se prépare à tout. Le scénario privilégié est celui d\'un retour
en classe après les dernières vacances de printemps, le 4 mai, mais nous sommes
évidemment tributaires de l\'évolution de l\'épidémie", a indiqué Jean-Michel
Blanquer dans une interview au Parisien. Depuis lundi, tous les établissements
scolaires et universitaires sont fermés en France pour lutter contre la
propagation de l\'épidémie de coronavirus.\n\n"Le but n\'est pas de toucher aux
vacances"\n\nInterrogé pour savoir si l\'année scolaire pourrait être prolongée
et les congés d\'été rabotés, le ministre a indiqué que "le but n\'est pas de
toucher aux vacances". Si le scénario d\'un maintien du bac et du brevet en fin
d\'année reste "la principale option" à ce stade, le ministère travaille "sur
différents scénarios en fonction de la durée du confinement". Parmi ces
hypothèses, "il y a la prise en compte, importante ou partielle, du contrôle
continu", a expliqué également Jean-Michel Blanquer.\n\nConcernant les
inégalités entre enfants liées à l\'école à la maison, le ministre a assuré que
serait mis en place après la crise "des modules de soutien gratuits pour les
enfants les plus en difficulté", "au moins à la fin du mois d\'août".'
```

Comme nous pouvons le voir, le texte a été très bien extrait.

Q3. et Q4. Nous allons d'abord définir des fonctions utiles à notre mise à jour.

[17]:
```
def extract_content(article_url):
    """Permet d'extraire l'auteur et contenu d'un article à partir de son url

    Arguments:
        article_url {str} -- Lien URL d'un article
```

```
    Returns:
        tuple of str -- Le tuple contient deux éléments. La 1ère est l'auteur␣
↪et la 2e le contenu.
    """
    article = npp.Article(article_url)
    article.download()
    article.parse()

    return article.authors, article.text
```

[18]:
```
def get_category(url):
    """ Retourne la catégorie d'un flux RSS de franceinfo à partir de son url

    Arguments:
        url {str} -- URL d'un flux RSS de franceinfo de type 'https://.../
↪category.rss'

    Returns:
        str -- La catégorie du flux.
    """
    try :
        return urls[0].split('.rss')[0].split("/")[-1]
    except:
        return "" # Il ne s'agit pas d'URL de franceinfo
```

[19]:
```
def get_image_link(article):
    """ Permet d'obtenir le lien vers l'image d'un article de franceinfo

    Arguments:
        article {dict} -- Un article de franceinfo correspondant à un␣
↪dictionnaire python.

    Returns:
        str -- Le lien URL vers l'image de l'article
    """
    try:
        if article.links[1].type == "image/jpeg": #le lien vers l'image existe
            return article.links[1].href
    except:
        pass

    return "" #there is no image file link
```

Maintenant, passons à la définition de notre grande fonction de mise à jour de notre base de données. Elle fonctionne comme ci : * Charge la base de données sous format json grâce au chemin d'accès donné en entrée * Ajoute à cette base de nouveaux articles trouvés via les flux RSS * Resauvegarde la base en format json dans le même répertoire donné

[20]:
```
def maj_database(database_path, urls):
    """Permet de mettre à jour la base de données des articles.

    Arguments:
        database_path {str} -- Chemin d'accès à la base de données au format␣
↪json
        urls {list} -- List d'urls vers les flux RSS
    """

    # Lecture du fichier json
    try:
        with open('data/france-info-20200301.json', 'r') as f:
            articles = json.load(f)
    except: #stop
        return "Erreur, veuillez bien spécifier le chemin d'accès du fichier"


    #Initialisation des conteneurs d'informations
    date_times, authors, categories, contents, image_links = [], [], [], [], []

    # Collecte des articles venant des flux
    for url in urls:
        feed = fp.parse(url)
        for article in feed.entries:
            if article.id not in list(articles.keys()): #Mise à jour de la base

                # get author and text of the article
                author, content = extract_content(article.id)

                new_article = dict()
                new_article['title']      = article.title
                new_article['date']       = time.strftime('%Y-%m-%dT%H:%M:%S',␣
↪article.published_parsed)
                new_article['author']     = author
                new_article['category']   = get_category(url)
                new_article['content']    = content
                new_article['image_link'] = get_image_link(article)

                #Ajouter du nouvel élément au json grâce à son "id"
                articles[article.id] = new_article

    #Sauvegarde du fichier avec la base mise à jour
    with open(database_path, 'w') as file:
        json.dump(articles, file)

    print('Done')
```

Testons la fonction de mise à jour.

[21]:
```
# On revérifie bien que "articles" contient 660 éléments d'abord
len(articles)
```

[21]:
```
660
```

[22]:
```
#Ensuite on fait la mise à jour...
maj_database(database_path='data/france-info-20200301.json', urls=urls)
```

```
Done
```

[23]:
```
#On recharge articles et on regarde sa taille
with open('data/france-info-20200301.json', 'r') as f:
    articles = json.load(f)
len(articles)
```

[23]:
```
717
```

[24]:
```
#Voyons le dernier élément pour s'assurer qu'il a bien été ajouté.
dict(list(articles.items())[-1:])
```

[24]:
```
{'https://www.francetvinfo.fr/sante/maladie/coronavirus/coronavirus-un-plan-
daide-aux-entreprises-et-aux-
salaries_3870517.html#xtor=RSS-3-[economie/entreprises]': {'title': 'Coronavirus
:\xa0un plan d'aide aux entreprises et aux salariés',
  'date': '2020-03-16T22:35:56',
  'author': [],
  'category': 'france',
  'content': 'La crise sanitaire du Covid-19 en France perturbe à la fois
salariés et entreprises, en proie à des déficits économiques déjà conséquents.
C'est pourquoi le gouvernement table sur une aide économique de dizaines de
milliards d'euros. Le but premier de cette aide est d'éviter le licenciement en
masse. C'est via le chômage partiel qu'une première ligne de défense va
s'opérer. « Concrètement, les salariés au SMIC vont conserver 100% de leur
rémunération, les autres salariés, 84% de leur salaire net » indique Christelle
Méral, journaliste. « Toutes les entreprises qui ont au moins un salarié ou un
apprenti peuvent mettre en place le chômage partiel. »\n\n\n\nUne souplesse
attendue des banques\n\nLe gouvernement aide aussi les employés à domicile « 80%
de leur rémunération est maintenue », rappelle Christelle Méral. Les
indépendants et artisans devraient être indemnisés par un fond de solidarité. Du
côté des banques, elles vont « décaler les remboursements des crédits jusqu'à 6
mois sans pénalité. Des nouveaux prêts seront accordés aux entreprises pour
payer les salariés », souligne la journaliste.',
  'image_link':
'https://www.francetvinfo.fr/image/75rw725tp-7874/500/281/21102757.jpg'}}
```

La base est effectivement mise à jour… Si on tente de recommencer la même maneouvre tout de

suite à l'instant, la base n'augmentera probablement pas, puisque de nouveaux flux ne sont pas encore disponible.

[25]:
```
#Ensuite on fait la mise à jour...
maj_database(database_path='data/france-info-20200301.json', urls=urls)

#On recharge articles et on regarde sa taille
with open('data/france-info-20200301.json', 'r') as f:
    articles = json.load(f)
len(articles)
```

```
Done
```

[25]:
```
717
```

Nous voyons bien que le contrôle d'identifiant est bien correct dans la base (pas d'ajout d'article qui existe déjà).

1.4 2. Extraction d'information

1.4.1 2.1. Extraction des entités nomées

[26]:
```
nlp = spacy.load("fr_core_news_sm")
```

Q5. Spacy utilise des modèles de données pré-entrainés sur de gros corpus de documents dans différentes langues. Tout démarre de la labelisation de certaines entités nomées à la main suivi de certaines règles grammaticales définies au préalable selon la langue sélectionnée. Une fois ces exemples données au modèle, ce dernier est entrainé sur des réseaux de neurones pour apprendre à identifier les entités nommées bien taggées dans la base d'apprentissage. La détection des entités nommées dépend donc fortement de la base d'apprentissage, c'est pourquoi SpaCy donne la possibilité d'entrainer son propre modèle à partir d'observations qu'on peut labeliser nous même à la main en suivant une structure bien déterminé.

A quoi correspondent les ´etiquettes IOB utilisées ?

[27]:
```
document = nlp("Jean Dupont est maire de Plouguemeur. Apple n'y a pas de locaux.
 ↪")
for token in document:
    print(token, token.ent_iob)
```

```
Jean 3
Dupont 1
est 2
maire 2
de 2
Plouguemeur 3
. 2
```

```
Apple 3
n' 2
y 2
a 2
pas 2
de 2
locaux 2
. 2
```

Comme bien détaillé dans la documentation (https://spacy.io/api/annotation#iob), les tags IOB sont des entiers qui nous permettent de connaitre la place d'un token vis à vis d'une entité. Un token peut être à l'intérieur d'une entité nomée (I pour intérieur –> 1), à l'extérieur (O pour Outside –>) ou au début (B pour Begin –> 3). Dans notre exemple, l'entité nomée est bien "Jean" d'où l'entier 3 associé au texte "Jean". L'entité continue avec "Dupont" et tous les autres tokens ne font pas parti de l'entité.

Q6.

[28]:
```python
def get_named_entities(doc):
    """[summary]

    Arguments:
        doc {[type]} -- [description]

    Returns:
        [type] -- [description]
    """

    return [(entity.text, entity.label_) for entity in doc.ents]
```

[29]:
```python
#Exemple d'article dont on veut connaitre les entités nomées
article.text
```

[29]:
```
'Le ministre de l\'Education nationale, Jean-Michel Blanquer, a précisé que ce
calendrier restait dépendant de l\'évolution de l\'épidémie.\n\nUn retour des
élèves en classe le 4 mai est le "scénario privilégié" mais reste "tributaire de
l\'évolution de l\'épidémie" en France, a indiqué dimanche 22 mars le ministre
de l\'Education Jean-Michel Blanquer. "Ce n\'est pas une annonce car tout sera
conditionné à l\'évolution du coronavirus et nous appliquons ce que nous dit le
ministère de la Santé", a précisé l\'entourage du ministre auprès de
l\'AFP.\n\n> Suivez l\'évolution de la pandémie de Covid-19 dans notre
direct.\n\n"On se prépare à tout. Le scénario privilégié est celui d\'un retour
en classe après les dernières vacances de printemps, le 4 mai, mais nous sommes
évidemment tributaires de l\'évolution de l\'épidémie", a indiqué Jean-Michel
Blanquer dans une interview au Parisien. Depuis lundi, tous les établissements
scolaires et universitaires sont fermés en France pour lutter contre la
propagation de l\'épidémie de coronavirus.\n\n"Le but n\'est pas de toucher aux
vacances"\n\nInterrogé pour savoir si l\'année scolaire pourrait être prolongée
```

```
et les congés d\'été rabotés, le ministre a indiqué que "le but n\'est pas de
toucher aux vacances". Si le scénario d\'un maintien du bac et du brevet en fin
d\'année reste "la principale option" à ce stade, le ministère travaille "sur
différents scénarios en fonction de la durée du confinement". Parmi ces
hypothèses, "il y a la prise en compte, importante ou partielle, du contrôle
continu", a expliqué également Jean-Michel Blanquer.\n\nConcernant les
inégalités entre enfants liées à l\'école à la maison, le ministre a assuré que
serait mis en place après la crise "des modules de soutien gratuits pour les
enfants les plus en difficulté", "au moins à la fin du mois d\'août".'
```

[30]:
```
get_named_entities(nlp(article.text))
```

[30]:
```
[("ministre de l'Education nationale", 'ORG'),
 ('Jean', 'PER'),
 ('Michel Blanquer', 'PER'),
 ('France', 'LOC'),
 ("ministre de l'Education Jean", 'ORG'),
 ('Michel Blanquer', 'PER'),
 ('ministère de la Santé', 'ORG'),
 ('AFP', 'ORG'),
 ('Suivez', 'LOC'),
 ('Covid-19', 'MISC'),
 ('Jean', 'MISC'),
 ('Michel Blanquer', 'PER'),
 ('Parisien', 'LOC'),
 ('France', 'LOC'),
 ('Jean', 'PER'),
 ('Michel Blanquer', 'PER')]
```

Six entités nommées ont été détectées dans ce texte. Le type de chaque entité est donné à sa droite. On sait donc que “Olivier” est une personne (PER) et “Chatenoy-en-Bresse” est un endroit (LOC comme Location). Quant au dernier type, la documentation officielle nous dit (MISC): “*Miscellaneous entities, e.g. events, nationalities, products or works of art.*”

La documentation de Spacy (https://spacy.io/models/fr) indique qu’il y 4 entités nommées détectable par ce modèle : PER, LOC, ORG et MISC. Dans l’exemple que nous venons d’exécuter, il y en a 3. Nous allons garder les 4 types d’entités. ORG signifie une organisation (privée ou gouvernementale).

Q7. Nous allons lancer ce calcul sur notre propre machine.

[31]:
```
%%timeit
result = []
for key in list(articles.keys())[:1]:
    doc = nlp(articles[key]['content'])
    for ent in doc.ents:
        result.append([ent.text, ent.start_char, ent.end_char, ent.label_])
```

```
22.2 ms ± 1.13 ms per loop (mean ± std. dev. of 7 runs, 10 loops each)
```

Le premier article prend en moyenne 22 millisecondes. Puisque nous avons 717 articles, la totalité du temps de traitement devrait être aux alentours de 16 secondes. Testons le.

[32]:
```
%%timeit
result = []
for key in articles.keys():
    doc = nlp(articles[key]['content'])
    for ent in doc.ents:
        result.append([ent.text, ent.start_char, ent.end_char, ent.label_])
```

```
28.8 s ± 735 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

Le temps moyen est de 27 secondes.

1.4.2 2.2. Analyse de entités nommées

Q8. Nous allons directement travailler avec les 717 articles (base déjà mise à jour).

[33]:
```
def plot_bar(series, title, top_N = 10):
    """ Fonction permettant d'afficher un bar plot horizontal à partir d'une␣
 ↪serie pandas.

    """

    series = series[::-1][-top_N-1:-1]
    to_plot  = [go.Bar(
                x=series.values,
                y=series.index,
                orientation = 'h'
    )]

    layout = go.Layout(title=title)
    fig = go.Figure(data=to_plot, layout=layout)
    iplot(fig)
```

[34]:
```
def get_top_entities(collection, top_N = 20):
    """[summary]

    Arguments:
        collection {list} -- La liste des documents text du corpus

    Keyword Arguments:
        top_N {int} -- Le nombre d'entités à afficher et à sauvegarder (default:
 ↪ {20})

    Returns:
```

```
        list of pandas Series -- La liste des top_N entités ordonnés par␣
↪occurence pour chaque type d'entité
    """

    entities_infos = []

    for article_content in collection:
        doc = nlp(article_content)
        for entity in doc.ents:
            entities_infos.append([entity.text, entity.label_])
    df = pd.DataFrame(entities_infos, columns=["entity_name", "entity_type"])
    saved_entities = []
    for _type in df.entity_type.unique():
        _series = df[df.entity_type == _type].entity_name.value_counts().
↪sort_values(ascending=False)
        plot_bar(_series, title = _type, top_N=top_N)
        saved_entities.append(_series[:top_N+1])
    return saved_entities
```

[36]:
```
# Création d'une collection totale des contenus articles
collection = []
for key in articles.keys():
    collection.append(articles[key]['content'])
```

[37]:
```
saved_entities = get_top_entities(collection)
```

Q9. Nous écrivons une fonction qui retourne le nombre de co-occurrence de deux entités dans un document donné. On considère qu'il y a co-occurrence entre deux entité dans un document lorsque les deux entités se succèdent dans la liste exhaustive des entités. Il faut comprendre par cela que les deux entités ne sont pas forcement collés l'un à l'autre dans le texte brut du document. Prenons un exemple dans notre collection :

[38]:
```
collection[47][:300] + "..."
```

[38]:
```
"Le site de General Electric à Belfort. (MELANIE JUVE / RADIOFRANCE)\n\nDeux
plans de sauvegarde de l'emploi, annoncés en 2019, commencent à se concrétiser
en ce début 2020 : celui de General Electric à Belfort et celui de Michelin à La
Roche sur Yon.\n\nfranceinfo : Peut-on parler de plans qui s'avèrent…"
```

En observant le 47e document de notre collection, on observe que l'entité "General Electric" et "Belfort" se succèdent dans la liste des entités de ce document. Pourtant la préposition "à" sépare ces deux entitiés à chaque fois mais on ne tient pas compte de cette préposition dans le calcul du nombre de co-occurence des deux entités. On voit donc bien qu'il y a 2 co-occurences de ces entités dans ce début de texte…

[39]:
```python
def get_entities_cooccurrence(e1, e2, doc):
    """[summary]

    Arguments:
        e1 {spaCy entity} -- La 1ère entité nomée
        e2 {spaCy entity} -- La 2eme entité nomée
        doc {spaCy doc}   -- Un document spaCy où chercher les co-occurrences

    Returns:
        int -- Le nombre total de co-occurence dans le document donné
    """
    nb = 0 #nombre total de cooccurrences
    entities = doc.ents
    for i in range(len(entities)-1):
        if (entities[i].text == e1 and entities[i+1].text == e2) or \
           (entities[i].text == e2 and entities[i+1].text == e1):
            nb += 1

    return nb
```

Jetons un oeil sur les entités nommées du document 47 de notre collection.

[40]:
```python
get_named_entities(nlp(collection[47]))
```

[40]:
```
[('General Electric', 'ORG'),
 ('Belfort', 'LOC'),
 ('MELANIE', 'MISC'),
 ('RADIOFRANCE', 'MISC'),
 ('General Electric', 'ORG'),
 ('Belfort', 'LOC'),
 ('Michelin', 'ORG'),
 ('La Roche', 'LOC'),
 ('Yon', 'LOC'),
 ('General Electric', 'ORG'),
 ('CFE', 'ORG'),
 ('CGC', 'ORG'),
 ('SUD Industrie', 'ORG'),
 ('PSE', 'ORG'),
 ('GE', 'ORG'),
 ('GE', 'ORG'),
 ('CDI', 'ORG'),
 ('CDD', 'MISC'),
 ('Michelin', 'ORG'),
 ('La Roche', 'LOC'),
 ('Yon', 'LOC'),
 ('Michelin', 'ORG'),
 ('Michelin', 'ORG'),
```

```
 ('Michelin', 'ORG'),
 ('France', 'LOC'),
 ('Michelin', 'ORG')]
```

[41]:
```
# Cherchons le nombre de co-occurrence des deux entités nomées précédemment␣
↪citées
get_entities_cooccurrence('General Electric', 'Belfort', nlp(collection[47]))
```

[41]: 2

Q10. Cherchons des co-occurences parmi la liste des entités nommées les plus fréquentes

[42]:
```
# Voyons voir le contenu du premier type d'entité sauvegardé (PER)
saved_entities[0]
```

[42]:
```
Emmanuel Macron               81
Jean                          68
Brexit                        47
Boris Johnson                 42
Angela Merkel                 36
Bruno Le Maire                26
Birkenau                      23
César                         20
Isabelle Kocher               19
Michel Barnier                15
Agnès Buzyn                   15
Vladimir Poutine              15
Claude Féraud                 14
Donald Trump                  14
président de la République    14
Kyriakos Mitsotakis           13
Ginette Kolinka               12
Christophe Dettinger          12
Aujourd'hui                   12
Eveline Szpirglas             12
Xavier Bertrand               12
Name: entity_name, dtype: int64
```

Il s'agit d'une pandas Series dont l'index est constitué des entités nomées et les valeurs sont le nombre d'occurence. On s'interessera donc aux index pour le calcul des co-occurences.

[43]:
```
#Stockage des articles de notre collection comme document nlp
docs = []
for doc in collection:
    docs.append(nlp(doc))
```

[44]:
```
occ_infos = defaultdict(int) #dictionnaire qui contiendra toutes les␣
↪coocurrences des entités
for doc in docs:
    for entity_type_1, entity_type_2 in it.combinations(saved_entities, 2):
        for entity_1 in entity_type_1.index.values:
            for entity_2 in entity_type_2.index.values:
                occ_infos[entity_1 + " -- " + entity_2] +=␣
↪get_entities_cooccurrence(entity_1, entity_2, doc)
```

[45]:
```
pd.Series(occ_infos).sort_values(ascending=False)[:40]
```

[45]:
```
JT -- JT\n\n                          359
Brexit -- Union européenne             42
Royaume-Uni -- Union européenne        36
Birkenau -- Auschwitz                  31
Brexit -- Brexit                       27
France -- JT\n\n                       26
Brexit -- Français                     24
Brexit -- UE                           22
Royaume-Uni -- Brexit                  21
Brexit -- C'                           21
Brexit -- Royaume-Uni                  21
Londres -- Brexit                      19
Royaume-Uni -- UE                      19
Brexit -- Londres                      19
Britanniques -- Brexit                 18
Brexit -- Britanniques                 18
Boris Johnson -- Brexit                16
Royaume -- Brexit                      13
Brexit -- Royaume                      13
Brexit -- Downing Street               12
Brexit -- Premier ministre             12
Britanniques -- Union européenne       12
Royaume -- Union européenne            12
France -- Covid-19                     11
Chine -- JT\n\n                        11
Chine -- Covid-19                      11
Brexit -- Parlement européen           10
Allemagne -- JT\n\n                    10
Wuhan -- Français                      10
France -- Français                     10
France -- Brexit                       10
Boris Johnson -- Premier ministre      10
Brexit -- France                       10
Agnès Buzyn -- ministre de la Santé     9
Uni -- Brexit                           9
Brexit -- Uni                           9
```

```
Emmanuel Macron -- la France              9
Birkenau -- Pologne                       9
Londres -- Union européenne               9
Eveline Szpirglas -- Auschwitz            8
dtype: int64
```

[46]:
```
plot_bar(pd.Series(occ_infos).sort_values(ascending=False),
         title="Cooccurrence des entités nommées les plus fréquentes",
         top_N=23)
```

[47]:
```
nlp("Brexit").ents[0].label_, nlp("UE").ents[0].label_, nlp("Union Européenne").
↪ents[0].label_
```

[47]:
```
('PER', 'ORG', 'ORG')
```

"Brexit" est identifié comme une personne tandis que "UE" et "Union Européenne". Nous pouvons remarquer que le sujet émergent est le "Brexit" et cette entité est intimement liée à une organisation (UE) ou à un endroit (UK ou Londres). Les relations semblent assez pertinentes.

Q11.

[48]:
```
def characterize_link(ent_list, docs):
    link = defaultdict(list)
    for i in range(len(docs)):
        doc_index = defaultdict(list)
        for e1, e2 in ent_list:
            for j in range(len(docs[i].ents) - 1):
                #Si les deux entités sesuivent
                if e1 == docs[i].ents[j].text and e2 == docs[i].ents[j+1].text:
                    # Capter le texte entre les deux entités
                    between_doc = docs[i].text[docs[i].ents[j].end_char :␣
↪docs[i].ents[j+1].start_char]
                    for token in nlp(between_doc): #vérifier s'il y a un verbe␣
↪entre les deux
                        if token.pos_ == 'VERB':
                            doc_index[i].append(token)
                    if len(doc_index[i])>0:
                        link[e1 + " -- " + e2].append(dict(doc_index))
    return link
```

[49]:
```
find_links = [
    ("Brexit", "UE"),
    ("France", "JT"),
    ("Chine", "Covid-19")
]
```

[50]:
```
pprint(dict(characterize_link(find_links, docs=docs)))
```

```
{'Brexit -- UE': [{37: [fait, est, négocier]},
                  {294: [doux, garantir]},
                  {296: [quittant]},
                  {298: [votés]},
                  {305: [Négocier, inquiètent]}],
 'Chine -- Covid-19': [{578: [>]},
                       {602: [eu, hospitalisé, testé]},
                       {632: [fait]}],
 'France -- JT': [{51: [jugés, convoqué]}]}
```

Comme nous pouvons constater le résultat, cette fonction nous permet de donner en entrée une liste de tuples d'entitées et retourne pour chaque tuple le numéro du document dans lequel il a été trouvé suivi de la liste de tous les verbes qui caractérisent la liaison entre ces deux entités. Par exemple, en regardant le tuple (Chine, Covid19), nous pouvons constater que les verbes les liant dans le document 602 sont "avoir", "hospitaliser" et "tester". Jetons un coup d'oeil à ce document.

[51]:
```
docs[602]
```

[51]:
```
Il est la première victime européenne du coronavirus : Adriano Trevisan, 78 ans.
Comment ce maçon à la retraite, habitant un petit village au nord de l'Italie,
a-t-il pu être contaminé ? Il n'a jamais été en Chine et n'a jamais eu de
contact avec des malades connus. L'homme était hospitalisé depuis 10 jours pour
une autre pathologie, mais il a été testé positif au Covid-19.

L'Italie du Nord se barricade

Une femme de 77 ans est également morte chez elle. Elle aurait pu contracter le
virus lors de son passage dans un hôpital où au moins un patient était infecté.
"Nous faisons des tests sur l'entourage des victimes, révèle Giulio Gallera,
conseiller médical dans la région de Lombardie.13% des tests sont positifs donc
l'épidémie est évidente." En Italie, les cas de contamination ont plus que
doublé en 24 heures et presque 60 personnes sont infectées. Alors, l'Italie du
Nord se barricade.

Le JT

Les autres sujets du JT
```

Ces verbes trouvés sont essentiellement dans le 1er paragraphe.

Q12.

[52]:
```
def get_entity_pairs(verb, docs):
    pairs = []
    for doc in docs:
        for i in range(len(doc.ents) - 1):
            e1, e2 = doc.ents[i], doc.ents[i+1]
            between_doc = doc.text[e1.end_char : e2.start_char]
```

```
            for token in nlp(between_doc): #vérifier s'il y a un verbe entre les deux
                #print(token.text)
                if token.pos_ == 'VERB' and token.text == verb:
                    pairs.append((e1, e2))
    return pairs
```

[53]:
```
get_entity_pairs("testé", docs[300:700])
```

[53]:
```
[(Logement, France), (Japon, Français), (Chine, Codogno), (Chine, Covid-19)]
```

Cet exemple montre que la nature de la relation définie uniquement par un verbe entre les deux entités n'est pas très bonne. On a pu retrouver l'exmple donné par le tuple (Chine, Covid-19) dans la question 11 mais les autres réponses ne sont pas assez pertinentes. Il faudrait plutot une analyse syntaxique plus approfondie pour s'attendre à des résultats plus convaincants, ceci permettrait de mieux définir la nature des relations entre les entités.

[54]:
```
get_entity_pairs("juger", docs)
```

[54]:
```
[(BlackRock, BlackRock),
 (CNCDH, Commission),
 (Syrie, Français),
 (CNCDH, France),
 (Peter Maurer, Comité international de la Croix),
 (Madame Isabelle Kocher, Julien Denormandie),
 (Tribunal militaire et la Cour de sureté de l'Etat, Constitution)]
```

Cet exemple reste quand même pas mal, les entités retournées ont un lien avec le verbe recherché.