Démonstration spaCy

Projet tutoré FIE5 ISIS 2023

AYMARD Marion | JOLIVET Anaëlle | BOURDARIAS Cléa

Entrée [1]:
```
# Importe spaCy
import spacy

# Crée un objet nlp français vide
nlp = spacy.blank("fr")
```

Le paquet "fr_core_news_sm" est un petit pipeline pour la langue française qui propose toutes les fonctionnalités de base et qui a été entraîné sur des textes d'actualité. Ici on utilise le moyen (**fr_core_news_md**) car le petit ne suporte pas certaines fonctions que l'on va utiliser.

Entrée [2]:
```
# Charge le petit pipeline français
nlp = spacy.load("fr_core_news_md")
```

On utilise la phrase **"Je me suis fait mal au poignet hier après-midi."**.

Entrée [3]:
```
# Crée en traitant une chaine de caractères avec l'objet nlp
doc = nlp("Je me suis fait mal au poignet hier après-midi.")
```

Prédire les type de mot dans la phrase

Entrée [4]:
```
# Itère sur les tokens
for token in doc:
    # Affiche le texte et l'étiquette de partie de discours prédite
    print(token.text, token.pos_)
```

```
Je PRON
me PRON
suis AUX
fait VERB
mal ADV
au ADP
poignet NOUN
hier ADV
après-midi NOUN
. PUNCT
```

Prédire comment les mots sont reliés

Entrée [5]:

```
for token in doc:
    print(token.text, token.pos_, token.head.text)
```

```
Je PRON fait
me PRON fait
suis AUX fait
fait VERB fait
mal ADV fait
au ADP poignet
poignet NOUN fait
hier ADV fait
après-midi NOUN fait
. PUNCT fait
```

Prédire des similitudes entre les phrases

On prend les phrases **"J'ai mal aux dents."** et **"J'ai une douleur dentaire."**. On peut determiner un taux de similitude entres ces phrases.

Entrée [6]:

```
# Compare deux documents
doc1 = nlp("J'ai mal aux dents.")
doc2 = nlp("J'ai une douleur dentaire.")
print(doc1.similarity(doc2))
```

```
0.8645883336750065
```

On prend les phrases **"J'ai mal aux dents."** et **"J'ai une douleur aux pieds."**

Entrée [7]:

```
# Compare deux documents
doc1 = nlp("J'ai mal aux dents.")
doc2 = nlp("J'ai une douleur aux pieds.")
print(doc1.similarity(doc2))
```

```
0.927480481745581
```

On prend les phrases **"Je n'ai pas mal aux dents."** et **"J'ai mal aux dents."**

Entrée [8]:

```
# Compare deux documents
doc1 = nlp("Je n'ai pas mal aux dents.")
doc2 = nlp("J'ai mal aux dents.")
print(doc1.similarity(doc2))
```

```
0.7649048822216306
```

Cette fonction n'est **pas** forcément **pertinante** car on remarque que les phrases "J'ai mal aux dents." et "J'ai une douleur aux pieds." sont plus similaire que les phrases "J'ai mal aux dents." et "J'ai une douleur dentaire.". De plus, on remarque un taux de similitude important entre les phrases "Je n'ai pas mal aux dents." et "J'ai mal aux dents." alors qu'elles sont à caractère opposé.

Recupérer les mots qui nous intéresse

Toujours avec notre phrase **"Je me suis fait mal au poignet hier après-midi."**, on peut déterminer les mots qui nous semblent importants.

Entrée [9]:
```
# Initialise avec le vocabulaire partagé
from spacy.matcher import Matcher
matcher = Matcher(nlp.vocab)
```

Entrée [10]:
```
doc = nlp("Je me suis fait mal au poignet hier après-midi.")
```

Entrée [11]:
```
# Écris un motif qui recherche une forme de "mal" suivie d'un nom
pattern = [{"LEMMA": "mal"}, {"POS": "ADP"}, {"POS": "DET", "OP": "?"}]

# Ajoute le motif au matcher et applique le matcher au doc
matcher.add("DOULEUR", [pattern])
matches = matcher(doc)
print("Nombre de correspondances trouvées :", len(matches))

# Itère sur les correspondances et affiche la portion de texte
for match_id, start, end in matches:
    print("Correspondance trouvée :", doc[start:end].text)
```

```
Nombre de correspondances trouvées : 1
Correspondance trouvée : mal au poignet
```

Le format de ce pattern permet de trouver le même résultat même si la phrase est légèrement différente. Par exemple, essayons avec la phrases **"Je me suis fait mal à la main hier après-midi."**.

Entrée [12]:
```
doc = nlp("Je me suis fait mal à la main hier après-midi.")

matches = matcher(doc)
print("Nombre de correspondances trouvées :", len(matches))

for match_id, start, end in matches:
    print("Correspondance trouvée :", doc[start:end].text)
```

```
Nombre de correspondances trouvées : 1
Correspondance trouvée : mal à la main
```

On remarque qu'il repère le bout de phrase **"mal à la main"** même si **"mal"** et **"main"** sont séparés de deux mots et non d'un seul comme dans **"mal au poignet"**.

Récupérer les mots importants regroupés dans un fichier JSON

Il est possible de comparer les mots de notre doc avec ceux d'une liste d'un fichier json. Ici on récupère les mots que nous voulons trouver dans le texte depuis le fichier json **"vocabulaire.json"**.

Entrée [22]:

```
import json

with open("vocabulaire.json", encoding="utf8") as f:
    VOCABULAIRE = json.loads(f.read())

print(VOCABULAIRE)
```

```
['mal', 'poignet', 'hier', 'aïe', 'jambe', 'voiture']
```

Ensuite, dans la phrases **"Je me suis fait mal au poignet hier après-midi."** on ne recupère que les mots intéressants, c'est à dire, ceux qui sont dans le fichier json.

Entrée [23]:

```
nlp = spacy.blank("fr")
doc = nlp("Je me suis fait mal au poignet hier après-midi.")

# Importe le PhraseMatcher et initialise-le
from spacy.matcher import PhraseMatcher

matcher = PhraseMatcher(nlp.vocab)

# Crée des motifs objets Doc et ajoute-les au matcher
# C'est la version rapide de : [nlp(country) for country in COUNTRIES]
patterns = list(nlp.pipe(VOCABULAIRE))
matcher.add("VOCABULAIRE", patterns)

# Appelle le matcher sur le document de test et affiche le résultat
matches = matcher(doc)
print([doc[start:end] for match_id, start, end in matches])
```

```
[mal, poignet, hier]
```

On remarque que les mots séléctionnés ne sont que ceux qui sont listés dans le fichier **"vocabulaire.json"**.