Programmation de Modèles Linguistiques (II)

(L5SOPROG L3 Sciences du Langage)

2023-2024

TD1 Analyse en caractères

Caroline Koudoro-Parfait, Sorbonne Université

Objectifs

- Rappels de manipulations simples sur un texte avec l'environnement de travail spyder
- Identifier les propriétés morphologiques d'une langue
- Les comparer à d'autres

1 Identification de langue améliorée

Données :

1. corpus multilingue vu en cours

Énoncé

1. En vous appuyant sur la *baseline* en mots vue dans le TD4 du semestre 5, travailler avec des tri-grammes de caractères
2. ré-évaluer ce nouveau modèle
3. Donner le diagnostic de langue avec un score de confiance en pourcentage
   ➛ Évaluer le taux de réussite du programme en calculant les VP, FP, FN,VN par exemple.
4. Donner les autres langue possibles du document ➛ Langues proches
5. Représenter graphiquement avec Matplotlib[1] ou Seaborn[2] pour chaque langue quelles sont les langues les plus proches.

Attendus

1. Le programme doit être présenté selon *les bonnes pratiques de programmation*
2. Le programme doit être factorisé
3. Vous devez choisir des structures de données pertinente pour stocker vos données
4. Le programme doit être développé sous l'environnement spyder

Principaux outils nécessaires :

- Json
- CountVectorizer/TfIdfVectorizer

[1] `https://matplotlib.org/`
[2] `https://seaborn.pydata.org/`

2 Bonus : Reconnaissance d'entités nommées au format IOB

Données :

1. Un jeu de données déjà annoté, au format CSV : *CSV_annote* sur Moodle
2. des textes a annoter automatiquement

Résultat attendu :

1. Aligner les entités pour calculer la précision, le rappel et le f-score

Principaux outils nécessaires :

- csv
- Spacy
- Json

Programmer avec spaCy

`https://spacy.io/usage/linguistic-features#named-entities`

```
import spacy

nlp = spacy.load("en_core_web_sm")
doc = nlp("San Francisco considers banning sidewalk delivery robots")

# document level
ents = [(e.text, e.start_char, e.end_char, e.label_) for e in doc.ents]
print(ents)

# token level
ent_san = [doc[0].text, doc[0].ent_iob_, doc[0].ent_type_]
ent_francisco = [doc[1].text, doc[1].ent_iob_, doc[1].ent_type_]
print(ent_san)  # ['San', 'B', 'GPE']
print(ent_francisco)  # ['Francisco', 'I', 'GPE']
```

Le programme doit permettre de :

- Annoter les textes au format IOB avec spaCy : *Texte*. Vous devez récupérer :
  - les labels type de spaCy (PER : personne ; LOC : localisation ; MISC : Divers ; ....)
  - les labels *I : Inside – O : outside – B : beginning*
- Récupérer les Annotations dans les csv et les mettre au format IOB : *CSV_annotes*

- Sauvegarder les données pour chaque texte dans un fichier .bio selon la structure de données qui vous semble pertinente*. Les fichiers de sorties se nomment comme les fichiers d'entrées.
- indiquer le temps de travail de l'outil en l'affichant

Les données sont structurées par spaCy* comme suit :

```
['San', 'B', 'GPE']
['Francisco', 'I', 'GPE']
```

Devoir

- 1 script python .py
- quelques phrases de conclusion sur les résultats (qu'est-ce qui était attendu, qu'est-ce qui est inattendu ?)
- quelques phrases sur l'environnement de développement Spyder comparé à Jupyter notebook, point fort, faible ...

  Vous déposerez sur Moodle une archive zip nommée NUMETU.zip (où NUMETU est votre numéro d'étudiant) et contenant :
- Votre code exporté au format Python .py (et pas ipynb)
- le PDF du document que vous avez produit

  Date limite : indiquée sur le Moodle !