Programmation de Modèles Linguistiques (I)

(L5SOPROG L3 Sciences du Langage)

2024-2025

Projet

Caroline Koudoro-Parfait, Sorbonne Université.
Crédit : Gaël Lejeune, Sorbonne Université

Date et Rendu

La date de rendu est fixée au **3 janvier 2024 à 23h59** sur MOODLE uniquement. Tout retard sera pris en compte dans la note. Pour tout souci sur ce dépôt envoyez un mail conjoint à : `caroline.parfait@sorbonne-universite.fr`

Le rendu doit être composé d'un rapport de 3-6 pages au format PDF et de votre code Python en `.ipynb` ET en `py`.

Vous devez commenter votre code, c'est à dire que l'on doit comprendre pour chaque fonction/chaque cellule quels sont les traitements que vous faites. Dans le rapport vous expliquerez pourquoi vous faites ces traitements.

Tout le rendu doit être archivé dans un seul fichier `.zip` et ensuite déposé sur le site Moodle (Rubrique: Projet).

Codes Python

Les codes Python attendus seront organisés dans un ou plusieurs fichiers. Chaque fichier devra être présent dans les deux formats `ipynb` ET `py` Factorisez vos codes Python: importer les bibliothèques et définir d'abord les fonctions, puis les appeler au fur et à mesure que vous avez besoin.

Rapport

Le rapport, composé de 3-6 pages, doit être organisé par les sections "Introduction", "Développement", "Conclusion et perspectives" et "Références bibliographiques".

N'oubliez pas de mettre sur la première page du rapport vos nom, prénom et numéro d'étudiant.

"Introduction" , expliquez le sujet et les problèmes traités (par ex. les avantages d'utiliser un moteur de recherche et ses limitations).

"Développement" , expliquez vos choix concernant les traitements effectués sur les données tels que : les choix de tokeniseur, le nettoyage ou non du vocabulaire, etc. Montrez les résultats intermédiaires et expliquez aussi les choix de traitements des résultats intermédiaires ou partiels s'il y a lieu.

"Conclusion et perspectives" , expliquez les difficultés que vous avez rencontrées et comment vous les avez résolues. Vos conclusions et les améliorations envisageables du programme que vous avez produit.

- Ne mettez pas de codes Python dans le rapport sauf si vraiment vous devez illustrer quelque chose d'important.
- Le projet consiste essentiellement à améliorer ce que nous avons fait dans les derniers TDs.

1 Extraction d'Entités Nommées, annoter et évaluer (Difficulté *)

Données :

1. Un jeu de données déjà annoté, au format CSV : *CSV_annote* sur Moodle
2. Un jeu de données à faire annoter : *CSV_a_annote* sur Moodle

Résultat attendu :

1. Sur le jeu de données 1: évaluer les résultats de `spacy_sm`, `spacy_md` et `spacy_lg` avec précision, rappel et f-mesure pour chaque catégorie (PERS, LOC, ORG et MISC)
2. Sur le jeu de données 2:
   - faire annoter par 3 annotateurs différents, calculer l'accord inter-annotateurs,
   - fusionner les annotations (choix majoritaire)
   - évaluer les résultats de spaCy sur ce jeu de données

Principaux outils nécessaires :

- Pandas
- Spacy
- Json
- Kappa de Cohen score `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.cohen_kappa_score.html`

2 Évaluation de la lexicalité (Difficulté *)

Données :

1. Corpus Littéraire, *Dumas*, sur Moodle

Résultats attendus :

1. Calculer le taux de lexicalité de chacun des textes "bruts"
2. Calculer le taux de lexicalité en enlevant les entités nommées
3. Donner des statistiques sur les mots inconnus restants

Principaux outils nécessaires :

- Glaff
- Json

3 Moteur de Recherche (Difficulté **)

Données :

1. Le corpus multilingue vu en cours

Résultats attendus :

1. Indexation du corpus
2. Interrogation par l'utilisateur
3. Tri des résultats (par similarité du document avec la requête[1])

Principaux outils nécessaires :

- Json
- input
- CountVectorizer/TfIdfVectorizer

4 Reconnaissance d'entités nommées des lieux pour une représentation de l'espace politique (Difficulté ***)

Données :

1. corpus Polux

Résultats attendus :

1. Reconnaissance d'entité nommées de lieux (EN LOC)
2. Indexation des EN LOC
3. Interrogation par l'utilisateur (penser que l'utilisateur peut approximer les noms de lieux)
4. Tri des résultats (par similarité entre les documents, réunir les documents qui mentionnent les mêmes noms de lieux)
5. Géocodage
6. Représentation sur une carte avec Googlemy map (`https://www.google.com/maps/d/`)

Principaux outils nécessaires :

- spaCy
- geopy (`https://geopy.readthedocs.io/en/stable/`)
- Json
- CountVectorizer/TfIdfVectorizer

[1] Dans un premier temps, trier simplement par nombre de mots en commun peut être une bonne approximation