# Programmation de Modèles Linguistiques (II)

(L5SOPROG L3 Sciences du Langage)

2024-2025

**TD3 Clustering avec un algorithme d'affinité de propagation**

Caroline Koudoro-Parfait, Sorbonne Université

## Objectifs

- Lire des fichiers selon une architecture de dossier données
- Utiliser la vectorisation de texte et la distance cosinus
- Construire une matrice
- Produire une représentation graphique des résultats

# 1 Clusters

**Données :**

1. Un jeu de données déjà annoté automatiquement avec spaCy au format IOB2

**Énoncé** Vous formerez des clusters à partir des sorties de reconnaissances d'entités nommées, pour représenter les formes les plus proches des entités, *par ex.* pour réunir dans un même cluster *"Morlincourt", "Morlincourtl", "Mlorlincourtl"*.

**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 pertinentes pour stocker vos données

**Principaux outils nécessaires :**

- TD3_VECTORIZER_CLUSTER.py (à télécharger sur Moodle)
- scikit learn
- Matplotlib ou Seaborn

## 1.1 Réflexions et conception en amont

Un cluster en traitement automatique des langues naturelles est un groupe de tokens réunis autour d'une ressemblance soit de type mophosyntaxique (tous les tokens du cluster sont des verbes par exemple), soit parceque les tokens ont des suites de caractères en communs, etc . Les termes réunis dans un cluster le sont autour d'un terme qu'on dit centroïde. Par exemple pour le cluster suivant le centroïde est "France" et le cluster comporte 10 tokens.

```
"ID 8": {
    "Centroide": "France",
    "Termes": [
      "Blancs",
      "Fance",
      "Fran—",
      "France",
      "Franceaetuelle",
      "Frnce",
      "Ionce",
      "Iranche",
      "laFance",
      "laFrance"
    ]
  },
```

Pour créer la matrice qui va permettre de former les groupes on peut utiliser :

- différents types de vectorisation : au grain mot ou au grain caractère (n-gram),
- différents type de distances permettant de déterminer les tokens les plus proches,
- plusieurs type d'algorithme pour grouper les tokens les plus proches.

Dans ce TD nous utiliserons un algorithme d'affinité de propagation, ce qui permet de créer automatiquement des clusters sans avoir du définir au préalable un nombre de cluster attendu.
En amont de la suite du TD vous réfléchirez à quels types de groupes d'entités vous voulez former ?
Vous Rédigerez quelques lignes précisant vos idées et proposerez un plan pour développer votre programme.

## 1.2 Développement du programme de mise en forme des données

Vous utiliserez les fichiers annotés automatiquement avec spaCy au format IOB2, disponibles sur Moodle, pour préparer une entrée adéquate pour le programme de clusterisation des entités nommées : TD3_VECTORIZER_CLUSTER.py. Le programme attend en entrée un ensemble (set) des tokens.

## 1.3 Commenter le programme TD3_VECTORIZER_CLUSTER.py

Vous commenterez de manière précise et pédagogique[1] le programme, en décrivant, par exemple, le type de données attribué pour chaque variable, la fonction des boucles, des conditions s'il y en a, l'usage des packages, etc.
Vous devez décrire chaque étape du programme et les expliciter.

Les commentaires doivent figurer dans le script.

[1] Comme si vous expliquiez à quelqu'un qui ne connaît rien à la programmation python)

### 1.4 Utiliser le programme TD3_VECTORIZER_CLUSTER.py

Vous ferez tourner le programme avec les données préparées tel que dans l'étape 1.2.
Préparez un format de sortie pour les clusters qui soit réutilisable tel qu'attendu en section 2.
Commentez les clusters que vous obtenez en sortie. Quel(s) paramètres pourriez-vous changer pour changer le contenu des clusters obtenus en sortie ? N'hésitez pas à faire des tests.

## 2 Représenter graphiquement les résultats

Représenter graphiquement les clusters. Vous pouvez par exemple représenter les centroïdes avec des points plus ou moins gros selon que le cluster comprend plus ou moins de termes. Par exemple un centroïde dont le cluster comprends 10 tokens aura un point plus gros que celui d'un centroïde dont le cluster ne compte que 5 tokens.

## 3 Bonus : explorer d'autres manières de calculer les clusters

En consultant les solutions proposées sur cette page, `https://scikit-learn.org/stable/modules/clustering.html` choisissez une autre manière de calculer les clusters.

## Devoir

- 1 ou plusieurs script(s) python .py, commentés,
- 1 PDF présentant :
  - La rédaction et le plan attendus en partie 1.1
  - quelques phrases de conclusion sur les résultats (qu'est-ce qui était attendu, qu'est-ce qui est inattendu ?)

  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 !