Ezriel Steinberg
Nathaniel Hayoun
IED – Paris 8

Analyse linguistique de comptes tweeters

Cahier des charges – Spécifications

Ce projet a pour but de réaliser un programme d'analyse linguistique de comptes tweeters, en calculant leur distance linguistique entre eux, afin de réaliser un *graphe de proximité linguistique*, mettant en lien les comptes les plus proches. L'idée serait de déterminer s'il est possible de visualiser graphiquement les diverses tendances politiques (gauche, droite, etc.) selon la proximité linguistique des comptes qui les représentent.

Ce projet se sépare en deux étapes, la première ayant été principalement réalisé par Ezriel et la seconde par moi. Voici la manière dont s'utilise ces deux programmes :

I dlTweets.py

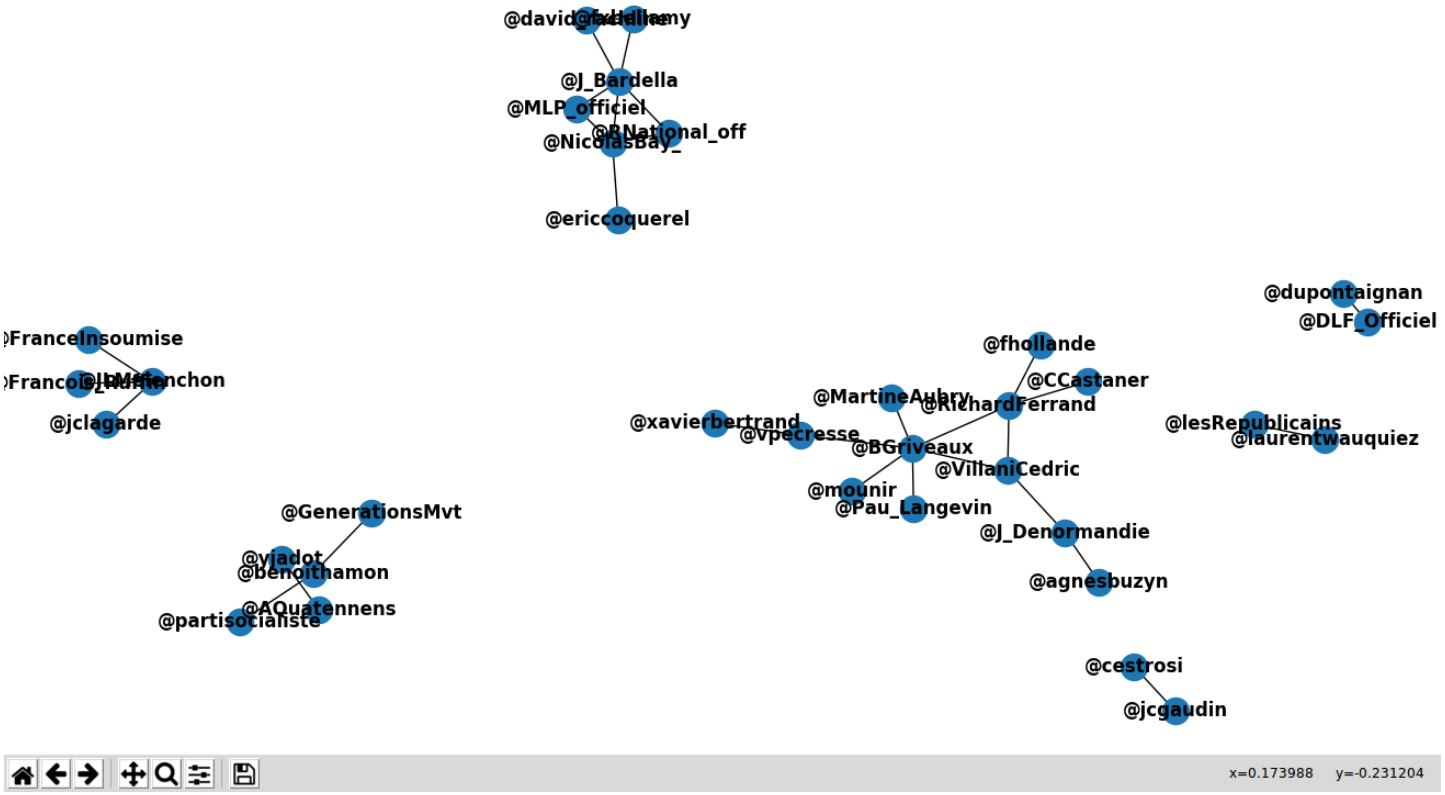
```
USAGE : dlTWeets.py [-f -a] [-n nb_tweets_per_account] [-o output_file] users_file or users_list
```

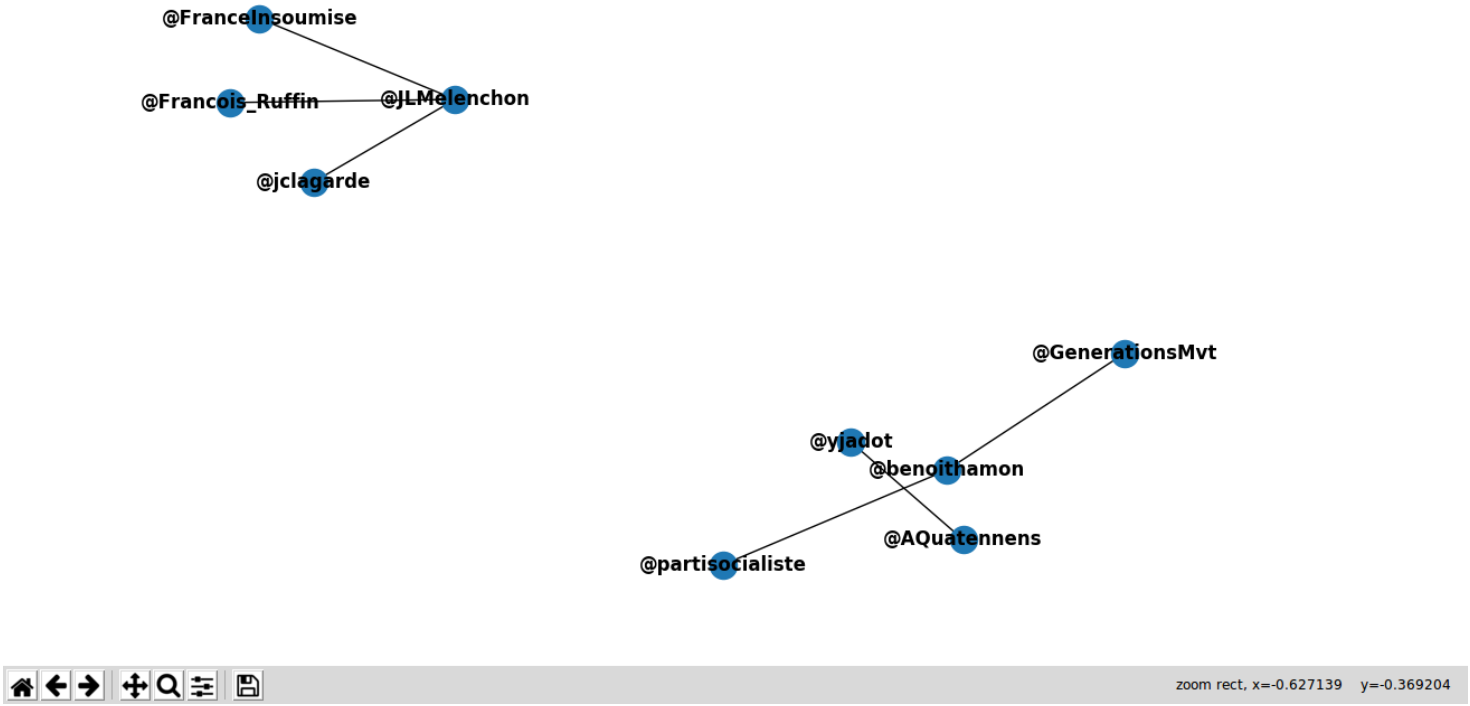
Un premier programme, nommé dm.py, prend en argument la liste des comptes tweeters dont on veut télécharger les tweets. Cette liste peut être écrite directement en argument sur la ligne de commande (option -a), mais dm.py peut aussi, avec l'option -f, récupérer les noms d'utilisateurs dans un fichier, séparés par des sauts de ligne.
L'option -n permet de spécifier le nombre de tweets à télécharger pour chaque compte (il peut y en avoir plusieurs milliers), enfin l'option -o spécifie le fichier output.
Ce programme génère en sortie un dictionnaire pickle, contenant deux objets dictionnaires :
- Un premier ("user"), qui, à chaque nom d'utilisateur, associe un dictionnaire où chaque mot lemmatisé est associé au nombre de ses occurrences dans ce compte.
- Et un deuxième ("allWords"), qui, à chaque mot, associe son nombre d'occurrences dans la *totalité* des comptes téléchargés (afin de faciliter la création d'une stoplist).

II analyseTweets.py

```
USAGE: analyseTweets.py [-d dict-file] [-o output-graph] [-s stopList-size] [-m min-occurences-number] [-p percentage-threshold]
```

Lors d'une deuxième étape, le programme nommé analyse.py prend en argument un fichier dict (par défaut « tweets.dict »), tel que celui écrit par le dm.py, et réalise dessus une analyse linguistique, afin de réaliser un graphe de proximité linguistique, qui sera affiché grâce au module networkx et mathplot.
Dans la page suivante, des images de ce à quoi peut ressembler un tel graphe, en entier ou en zoomant sur une partie.
Le programme sauve ce graphe dans un fichier au format gml, dont il est possible de spécifier le nom grâce à l'option -o. L'option -s permet de modifier la taille de la stoplist, afin d'éliminer plus ou moins de mots courants (par défaut, ce sont les 200 mots les plus courants qui sont éliminés), l'option -m permet de spécifier le minimum d'occurrences d'un mot pour qu'il soit pris en compte, et enfin l'option -p permet de spécifier le seuil (en pourcent) à partir duquel un lien de proximité sera considéré comme significatif (par défaut 33%).





Enfin, une fonction interne au programme, la fonction findNcommonWords, prend en argument une liste de noms de compte dont on veut voir les mots les plus fréquents, et en deuxième argument le nombre de mots en commun que l'on veut voir, et ce afin de pouvoir analyser les liens entre les différents *clusters* du graphe. Par exemple :

```
>>> findNcommonWords(["@NicolasBay_", "@MLP_officiel", "@J_Bardella",
"@RNational_off"],5)

emmanuel : 7.594409077242837
peuple : 7.081924640450012
union : 6.273054110155755
immigration : 5.92042774089543
arriver : 5.637502474462341
```