TP2 : normalisation des mots, étiquetage morpho-syntaxique et mots outils[1]

Mise en place de l'environnement de travail

À partir de cette séance, vous utiliserez la bibliothèque scientifique `nltk` (*Natural Language ToolKit*).

Pour le TP, vous aurez besoin d'utiliser des ressources de `nltk`. Pour les installer, ouvrez un terminal et exécutez les commandes suivantes :

```
# lancement de python
python

# téléchargement des ressources de nltk pour le TP
>>> import nltk
>>> nltk.download()
```

Avec la dernière commande, un explorateur de fichiers s'ouvre : à partir de celui-ci, téléchargez `Corpora/wordnet` ainsi que l'étiqueteur `Models/maxent_treebank_pos_tagger`.

En cas de problème lors du téléchargement, vous pouvez télécharger ces ressources sur le site `http://www.nltk.org/nltk_data/` : vous les placerez ensuite dans votre répertoire `/Users/nom-etudiant/nltk_data`.

Pour ce TP, vous utiliserez les fichiers texte de l'archive `tokens.zip` présents sur Madoc, dans le répertoire `TP 2`.

Normalisation des mots : lemmatisation, racinisation

La normalisation morphologique consiste à rapprocher sous une même graphie les différentes formes fléchies des mots. Par exemple, les mots *étudiant*, *étudiante* et *étudiants* correspondent tous les trois à un même mot : *étudiant*. Comme vous l'avez vu en cours, deux types de normalisation sont couramment utilisés :

- la lemmatisation : trouver le lemme pour une forme donnée (*e.g.* parlais → parler) ;
- la racinisation (ou *stemming*) : supprimer les affixes (*e.g.* parlais → parl).

Pour réaliser la *lemmatisation*, `nltk` dispose du lemmatiseur de WordNet. Voici un exemple de code pour utiliser le lemmatiseur de WordNet sur un texte en anglais.

```
import nltk

wnl = nltk.WordNetLemmatizer()
lemma = wnl.lemmatize('words')

print lemma # affiche 'word'
```

1. D'après un sujet de Florian Boudin.

Reprenez le script `example1.py` du TP 1 pour créer le script `lemmatize_text.py` qui lemmatise les tokens d'un document dont le nom est donné en paramètre. Ce script doit afficher chaque mot suivi de son lemme (sans considérer les signes de ponctuation). Vous calculerez également la taille du vocabulaire du document en nombre de tokens ainsi qu'en nombre de lemmes (sans prendre en compte la casse des tokens) et vous afficherez ces informations.

Pour réaliser la *racinisation*, `nltk` dispose de deux raciniseurs : celui de Porter et celui de Lancaster. Voici un exemple de code pour utiliser ces raciniseurs sur un texte.

```
import nltk

porter = nltk.PorterStemmer()
lancaster = nltk.LancasterStemmer()

stemP = porter.stem('words')
stemL = lancaster.stem('words')

print stemP, stemL # affiche 'word word'
```

De la même façon que précédemment, écrivez un script `stemmatize_text.py` qui racinise les tokens d'un document dont le nom est donné en paramètre. Ce script doit afficher chaque mot suivi de sa racine d'après Porter puis de sa racine d'après Lancaster (sans considérer les signes de ponctuation). Vous calculerez également la taille du vocabulaire du document en nombre de tokens ainsi qu'en nombre de stems selon chaque raciniseur (sans prendre en compte la casse des tokens) et vous afficherez ces informations. Comparez également les stems trouvés par chaque raciniseur.

Étiquetage morpho-syntaxique

Un étiqueteur morpho-syntaxique est un outil permettant d'assigner automatiquement à chacun des mots (au sens tokens) d'un texte, sa catégorie grammaticale (e.g. déterminant, adjectif, nom). Un exemple de phrase tokenisée et étiquetée en catégories morpho-syntaxiques est présentée ci-dessous :

"`J' adore les sushis .`" → "`J'_CL adore_V les_D sushis_N ._.`"

La bibliothèque `nltk` dispose de plusieurs méthodes d'étiquetage morpho-syntaxique pour la langue anglaise. Voici un exemple de code vous montrant comment utiliser un des étiqueteurs.

```
import nltk

pos_tagged_words = nltk.pos_tag('Cooper Industries Inc asked the
    Federal Commission')
# affiche [(u'Cooper', 'NNP'), (u'Industries', 'NNPS'), (u'Inc', '
    NNP'), (u'asked', 'VBD'), (u'the', 'DT'), (u'Federal', 'NNP'), (u
    'Commission', 'NNP')]
```

Écrivez un script `pos-tagged_text.py` qui lit un document tokenisé en entrée et qui écrit le contenu étiqueté en catégories morpho-syntaxiques dans un fichier en sortie (dont le nom est également donné en paramètre du script).

Liste des mots outil

Un mot-outil est une catégorie de mot dont le rôle syntaxique est plus important que le rôle sémantique. En effet, les mots d'un texte n'ont pas tous le même rôle. Certains participent à la construction du sens (mots pleins), d'autres sont utilisés pour la construction du texte (mots outils). De nombreuses applications en TALN suppriment les mots outils afin d'améliorer les performances et/ou de réduire les temps de calcul.

Pour élaborer une liste de mots outil, une solution consiste à extraire les mots les plus fréquents dans un corpus. La bibliothèque `nltk` permet de calculer la distribution des tokens dans un texte. Voici un exemple de code vous montrant comment utiliser les distributions de fréquence d'un texte.

```
import nltk

fdist = nltk.probability.FreqDist(['it', 'is', 'it', 'that', 'is', '
    the', 'best'])
vocabulary = fdist.keys()

top_words = vocabulary[:2]
print top_words
# affiche ['it', 'is']
```

Créez un script `stop-words.py` qui extrait les 50 mots les plus fréquents de l'ensemble des documents tokenisés et écrit ceux-ci dans un fichier en sortie (un mot par ligne). Pour vous aider, voici un extrait de code permettant de lire tous les fichiers d'un répertoire.

```
import os, re ,codecs

path = 'tp2/data/tokens/'
for fichier in os.listdir(path):
  fh = codecs.open(path+fichier, "r", "utf-8")
  content = fh.read()
  fh.close()
```