Université Paul Sabatier
EMMAB1J1 - Extraction d'information dans les documents textes, audio, vidéo

Enseignant : José G. Moreno

13/10/2017

TP 2. Extraction d'information sur documents textuels (2/3)

III - L'usage de méthodes statistiques

Les règles peuvent nous rapporter des informations fausses ou ignorer certains usages (le dernier est le plus fréquent). Les méthodes d'apprentissage statistiques sont couramment utilisés pour identifier correctement les entités nommées. Pour le faire, ces méthodes utilisent plusieurs informations comme le contexte ou l'information syntactique dans laquelle un mot apparaît. L'avantage de ces méthodes est qu'ils n'utilisent pas des règles prédéfinies et ils s'adaptant plus facilement à des nouveaux documents. Leur principal inconvénient est le coût de générer des annotations fiables pour l'étape d'apprentissage.

Utilisation de méthodes statistiques

Nous allons utiliser la méthode de reconnaissances d'entités nommées implémenté sur nltk. Pour le faire, nous allons utiliser des phrases dans lesquelles au moins un pays est mentionné. Voici le code pour récupérer la liste de pays :

```
# -*- coding: utf-8 -*-

import pandas
import re

def cleanAndSave(text):
    f = open("listCountries.txt",'a+')
    for val in re.finditer('\[\[[A-Z](\w| )+\]\]', text):
        val = val.group(0)
        f.write(val[2:-2]+"\n")
    f.close()

def main():
    pages_to_show = ['List of countries']
    df = pandas.DataFrame.from_csv("simplewiki.csv", encoding='utf-8')
    for page in pages_to_show:
        cleanAndSave(df['text'][df['title'] == page].item())

if __name__ == "__main__":
    main()
```

Le code sauvegarde l'information dans le fichier listCountries.txt.

```
Afghanistan
Albania
Algeria
Andorra
```

```
Angola
Antigua and Barbuda
Argentina
Armenia
Aruba
Australia
Austria ...
```

Le code est téléchargeable sur : moodle - tp2_a.py

Une fois la liste est produite, nous pouvons rechercher des phrases avec des mentions de la liste. Voici le code pour rechercher des phrases qui contient des mentions des pays :

```
# -*- coding: utf-8 -*-

import pandas
import re

def findPhrases(text,l):
    parts = text.split("\n")
    for part in parts:
        for val in re.finditer('[A-Z](\w| |\,)+\.', part):
            if len(val.group(0).split(" "))<10:
                continue
            for country in l:
                if country in val.group(0):
                    print val.group(0)
                    break

def main():
    countries = [x.strip() for x in
open("listCountries.txt",'r').readlines()]
    df = pandas.DataFrame.from_csv("simplewiki.csv", encoding='utf-8')
    for page in list(df['text']):
        findPhrases(page,countries)

if __name__ == "__main__":
    main()
```

Les résultats obtenus sont :

```
Japanese and Korean can be written in phonetic scripts that do not
indicate tone.

This character is specific to the Tay people of northern Vietnam.

May has also dealt with the war in Iraq and Syria.

Iraq then informs UNSCOM that it will be able to resume its flights.

Indonesian president BJ Habibie announced on 12 September that he would
do so.
...
```

Le code est téléchargeable sur : moodle – tp2_b.py

Pour utiliser une des méthodes implémentées sur nltk, il suffit d'utiliser

```
print(nltk.ne_chunk(nltk.pos_tag(nltk.word_tokenize("May has also dealt
with the war in Iraq and Syria."))))
(S
  May/NNP
  has/VBZ
  also/RB
  dealt/VBN
  with/IN
  the/DT
  war/NN
  in/IN
  (GPE Iraq/NNP)
  and/CC
  (GSP Syria/NNP)
  ./.)
```

Notez que d'abord il faut trouver les « tokens », puis faire l'étiquetage morphe-syntaxique (POS tagging) et finalement la reconnaissances d'entités nommées. Regardez le support du cours pour des explications.

Maintenant c'est à vous !

1) Le code a fonctionné au premier coup ? Pourquoi pas ?
2) Pourquoi le code

```
print(nltk.ne_chunk(nltk.pos_tag("May has also dealt with the war in
Iraq and Syria.".split())))
```

marche ? Expliquez les différences entre les résultats.
3) Pourquoi le code

```
print(nltk.ne_chunk(nltk.word_tokenize("May has also dealt with the war
in Iraq and Syria.")))
```

ne marche pas ? Pourquoi faut-il utiliser nltk.pos_tag ?
4) Quel type système de reconnaissances d'entités nommées est implémenté par la méthode ne_chunk ? Justifiez la réponse.
5) Modifiez le code tp2_b.py pour pouvoir appliquer la reconnaissance d'entités à toutes les phrases identifiés par le code. Dans quels cas ne_chunk ne marche pas ?
6) Utilisez ne_chuck pour faire la reconnaissance d'entités nommées sur des pages complètes. Comparez les résultats entre pages de nature différent (Pays vs Personne). Trouver vous des différences significatives ?
7) Utilisez les instructions sur https://github.com/nltk/nltk/wiki/Installing-Third-Party-Software pour installer le système de reconnaissances d'entités nommées de stanford https://nlp.stanford.edu/software/CRF-NER.shtml . Appliquez le modèle sur les phrases que vous avez trouves avec le code tp2_b.py et comparez les résultats obtenus avec ne_chunck. Trouvez-vous des différences ? Quel facteur peut les générer ?
8) Après vous expériences avec ne_chunk et StanfordNERTagger. Listez les gros lignes pour implémenter un système statistique pour le NER.