Initiation à la programmation: Extraction de sujets des documents

Zhentao Li

5 avril 2017

Formalization

- Certaines notions humaines se traduisent facilement en langage informatique. Par exemple, les mathématiques.
- D'autre notions comme "similarité" et "sujet (d'un document)" sont plus flous.
- Il nous faut une définition « formelle » que l'on recherche avant de pouvoir écrire un programme qui le retrouve.
- *Une alternative:* Pour des concepts que les humains ont de la difficulté à énoncer mais nous pouvons identifier des exemples (par ex, si on nous montre un document et un sujet, on peut dire oui ou non si le document traite du sujet), nous pouvons utiliser des méthodes statistiques au lieu de définitions formelles, en acceptant une pénalité sur la précision.

nltk

Dans un terminal

```
virtualenv venv --system
. venv/bin/activate
pip install nltk gensim
```

nltk est un module Python pour le traitement de texte. Après avoir installé (ou téléchargé et rendu son chemin retrouvable par Python).

```
import nltk
nltk.download()
```

La deuxième command ouvre un fenêtre qui vous permet de télécharger bases de données (de texte, corpus, etc). Choisissez `book` et cliquez sur `Download`.

gensim

Nous allons maintement voir un façon d'identifier les sujets de documents. Il y en a d'autre (et cela dépend aussi de ce qu'on veut dire par « sujet » et ce qu'on veut en faire).
Dans la dernière étape, nous allons utiliser **gensim**, une bibliothèque de modélisation de sujets. Comme `nltk`, il faut aussi l'installer.

```
from gensim import corpora, models, similarities
```

Tokenization

Comme dans l'exercise du TD3, nous commençons par extraire les mots du texte.

```
document = """At eight o'clock on Thursday morning
Arthur didn't feel very good."""
mots = nltk.word_tokenize(document)
```

Suppression de mots commun

Nous gardons que les mots rares.

```
from nltk.corpus import stopwords
stopword_list = stopwords.words('english')
stopword_list += ['.', ',', ';', ':', '``', "''",
                  '--', '?', '!', "'s"]
mots_rares = []
for mot in mots:
    if mot not in stopword_list:
        mots_rares.append(mot)
```

Plusieurs documents

Nous allons extraire les mots rares de plusieurs documents. Supposons que nous avons une liste de documents dans la variable `documents`. Sinon, il est possible d'utiliser des exemples de nltk.

```
documents = []
for name in nltk.corpus.gutenberg.fileids()[:6]:
    documents.append(nltk.corpus.gutenberg.raw(name))
```

```
def rares(document):
    mots = nltk.word_tokenize(document.lower())
    mots_rares = []
    for mot in mots:
        if mot not in stopword_list:
            mots_rares.append(mot)
    return mots_rares

mots_rares = []
for document in documents:
    mots_rares.append(rares(document))
```

Construction de la matrice document-terme

Nous allons créer un histogramme des mots rares (un « bag of words ») pour chaque document.

```
id2word = corpora.Dictionary(mots_rares)
bag_of_words = []
for rare in mots_rares:
    bag_of_words.append(id2word.doc2bow(rare))
```

Le modèle Latent Dirichlet Allocation

Nous utilisons gensim pour extraire des sujets à partir de ce histogrammes. Le nom des mots lui est inconnu!

```
lda = models.ldamodel.LdaModel(corpus=bag_of_words,
                               id2word=id2word,
                               num_topics=3,
                               chunksize=10000)

for topic in lda.print_topics():
    print topic[1]
```

Télécharger le script

```
www.di.ens.fr/~zhentao/intropython/test_nltk.py
```

Autres idées

- Essayer d'autre paramètres et autre modèles
- Extraction de racines des mots (pour avoir moins de mots différents dans l'histogramme).
- Suppression des mots qui n'apparaissent qu'une seule fois.

Autre idée: utiliser un dictionnaire des synonymes

```
from nltk.corpus import wordnet

for i, meaning in enumerate(wordnet.synsets('dog')):
    print "Meaning", i, "NLTK ID:", meaning.name()
    print "Definition:", meaning.definition()
    print "Synonyms:", ", ".join(meaning.lemma_names())
    print
```

Pour chaque définition du mot chien (« dog ») on affiche sa définition et ses synonymes.

```
Meaning 0 NLTK ID: dog.n.01
Definition: a member of the genus Canis (probably descended from the common wolf) t
Synonyms: dog, domestic_dog, Canis_familiaris

Meaning 1 NLTK ID: frump.n.01
Definition: a dull unattractive unpleasant girl or woman
Synonyms: frump, dog
[...]
```

Autre idée: rechercher simplement les mots les plus fréquents

Pour afficher les 10 mots rares les plus fréquents du quatrième document:

```
from collections import Counter
comptes = Counter(mots_rares[3])
for mot, freq in comptes.most_common(10):
    print mot, freq
```

On obtient

```
said 652
turnbull 544
macian 425
man 336
like 326
one 316
n't 178
two 177
us 162
quite 144
```