TC3 : Traitement Automatique des Langues
Introduction au TAL

Adrien Pavao

Septembre 2017

Les analyses

Voyons différentes analyses possible en Traitement Automatique des Langues (TAL), avec pour chacune une brève description et un exemple d'implémentation en Python avec la librairie nltk. Dans les exemples suivants, la variable *string_test* est une chaîne de caractères contenant le texte que l'on souhaite analyser.

- **Tokenisation :** On sépare le texte en tokens. Cela revient à peu près à diviser le texte en mot, en plus large. Par exemple, une ponctuation peut constituer un token.

```
tokens = nltk.word_tokenize(string_test)
```

- **Segmentation en phrase :** Si nous n'avons pas d'information sur la segmentation des phrases, le plus simple est de faire un split en utilisant le point comme séparateur.

```
sentences = string_test.split(".")
```

- **Etiquetage :** On souhaite attribuer à chaque mot sa fonction grammaticale (nom, adverbe...).

```
tagged = nltk.pos_tag(tokens)
```

- **Lemmatisation :** La lemmatisation s'applique sur un mot. On enlève les flexions du mot afin de la mettre dans sa forme la plus simple. Par exemple, "voitures" devient "voiture". Les flexions sont les accords et les conjugaisons. Le code suivant affiche "cat" à l'écran.

```
from nltk.stem import WordNetLemmatizer

lemmatizer = WordNetLemmatizer()
print(lemmatizer.lemmatize("cats"))
```

Figure 1: Exemple de lemmatisation sur des mots du corpus.

```
findings : finding
contributions : contribution
memoirs : memoir
fellows : fellow
was : wa
temperatures : temperature
weeks : week
emoticons : emoticon
```

On remarque qu'en anglais, la lemmatisation consiste principalement à enlever des "s" en fin de mot.

- **Racinisation :** A partir d'un mot on récupère sa racine. Par exemple, la racine du mot "échangeable" est "échang". Il ne faut pas confondre ce procédé avec la lemmatisation. Le code suivant affiche "run" à l'écran.

```
from nltk.stem.snowball import SnowballStemmer

stemmer = SnowballStemmer("english")
print(stemmer.stem("running"))
```

Figure 2: Exemple de racinisation sur des mots du corpus.

```
findings : find
I : i
contributions : contribut
Great : great
Thanks : thank
```

- **Reconnaissance d'entités nommées :** Une entité nommée est une expression faisant référence à une entité tel qu'un lieu précis ou une personne. Ce sont souvent des noms propres. Nous ne détaillerons pas l'implémentation ici car elle est un peu longue. Voici un exemple d'utilisation:

```
>>> get_named_entity("My name is Adrien and I live in Orsay.")
['Adrien', 'Orsay']
```

- **Résolution de coréférence :** L'objectif de cette analyse est de lier différentes expressions qui font référence à la même chose. Cette analyse

n'est pas prévue par la libraire nltk, cependant elle est possible avec la libraire Java CoreNLP.

- **Analyse syntaxique :**
  - **En constituants :** Chaque mot est représenté.
  - **En dépendances :** On représente le discours par un arbre, dans lequel certains mots sont importants (un nom par exemple) et d'autres servent principalement de lien, ou de noeud de l'arbre (un déterminant par exemple).

  Ces deux approches de la syntaxe présentent des structures différentes.