POLYTECH PARIS-SUD

Et5-Info

Module : Traitement Automatique des Langues - Traduction Automatique

TP 2 - Analyse linguistique avec le framework NLTK

Contexte :

Une plateforme d'analyse linguistique standard se compose des modules suivants :

1. **Découpage (Tokenization):** Ce module consiste à découper les chaînes de caractères du texte en mots, en prenant en compte le contexte ainsi que les règles de découpage. Ce module utilise généralement des règles de segmentation ainsi que des automates d'états finis.
2. **Analyse morphologique (Morphological analysis):** Ce module a pour but de vérifier si le mot (token) appartient à la langue et d'associer à chaque mot des propriétés syntaxiques qui vont servir dans la suite des traitements. Ces propriétés syntaxiques sont décrites en classes appelées catégories grammaticales. La consultation de dictionnaires de formes ou de lemmes permet de récupérer les propriétés syntaxiques concernant les mots à reconnaître.
3. **Analyse morpho-syntaxqiue (Part-Of-Speech tagging):** Après l'analyse morphologique, une partie des mots restent ambigus d'un point de vue grammatical. L'analyse morphosyntaxique réduit le nombre des ambiguïtés en utilisant soit des règles ou des matrices de désambiguïsation. Les règles sont généralement construites manuellement et les matrices de bi-grams et tri-grams sont obtenues à partir d'un corpus étiqueté et désambiguïsé manuellement.
4. **Analyse syntaxique (Syntactic analysis ou Parsing):** Ce module consiste à identifier les principaux constituants de la phrase et les relations qu'ils entretiennent entre eux. Le résultat de l'analyse syntaxique peut être une ou plusieurs structures syntaxiques représentant la phrase en entrées. Ces structures dépendent du formalisme de représentation utilisé : un arbre syntagmatique, un arbre de dépendance ou une structure de traits. L'analyse en dépendance syntaxique consiste à créer un arbre de relations entre les mots de la phrase. Le module d'analyse syntaxique utilise des règles pour l'identification des relations de dépendance ou des corpus annotés en étiquettes morpho-syntaxiques et en relations de dépendance.
5. **Reconnaissance d'entités nommées (Named Entity recognition):** Ce module consiste à identifier les dates, lieux, heures, expressions numériques, produits, évènements, organisations, présentes sur un ou plusieurs tokens, et à les remplacer par un seul token.

<u>Travail demandé</u>

Vous allez installer et expérimenter la plateforme d'analyse linguistique NLTK (une boite à outils pour le traitement automatique de la langue utilisant des approches hybrides).

I. Installation de la plateforme d'analyse linguistique NLTK

Avant de démarrer l'installation de la plateforme NLTK sur un poste équipé d'une distribution Linux (de préférence Ubuntu 16.04 LTS), il faudrait vérifier la version installée de Python:

`python –version`

Si `Python 2.7.12`, utiliser `pip`

Si `Python3.5` (ou supérieur), utiliser `pip3`

Installation : http://www.nltk.org/install.html

1. Installation de NLTK

```
pip install --user -U nltk
```

2. Installation de Numpy

```
pip install --user -U numpy
```

3. Test de NLTK en interactif

```
python
>>> import nltk
>>> nltk.download('punkt')
>>> from nltk.tokenize import word_tokenize
>>> text = "It's works!"
>>> print(word_tokenize(text))
```

Résultat:

```
['It', '’', 's', 'works', '!']
```

1. Evaluation de l'analyse morpho-syntaxique de la plateforme NLTK

1. Ecrire un programme Python utilisant le package `pos_tag` pour désambiguïser morpho-syntaxiquement le texte du fichier `wsj_0010_sample.txt`. Le résultat de ce module sera mis dans le fichier `wsj_0010_sample.txt.pos.nltk`.

   Note :
   Un exemple d'utilisation du package `pos_tag` se trouve sur le lien https://www.guru99.com/pos-tagging-chunking-nltk.html.

2. **Evaluation à l'aide des étiquettes Penn TreeBank (PTB) :** Utiliser le programme Python « evaluate.py » pour évaluer l'analyseur morpho-syntaxique de la plateforme NLTK. L'évaluation se fait en utilisant le fichier contenant l'annotation de référence `wsj_0010_sample.pos.ref`

   ```
   python evaluate.py wsj_0010_sample.txt.pos.nltk
   wsj_0010_sample.pos.ref
   ```

Note:
Pour calculer la précision de l'analyseur morpho-syntaxique de la plateforme NLTK à l'aide du script Python evaluate.py, il faut que les tokens (1ère colonne) des fichiers wsj_0010_sample.txt.pos.nltk et wsj_0010_sample.pos.ref soient identiques. Pour ce faire, il faut écrire un programme Python qui permet de ne garder que les tokens identiques dans les deux fichiers wsj_0010_sample.txt.pos.nltk et wsj_0010_sample.pos.ref.

3. **Evaluation à l'aide des étiquettes universelles :**
   a. Remplacer à l'aide d'un programme Python les étiquettes Penn TreeBank des fichiers « wsj_0010_sample.txt.pos.nltk » et « wsj_0010_sample.txt.pos.ref » par les étiquettes universelles en utilisant la table de correspondance « POSTags_PTB_Universal.txt ».

      Note :
      Nommer les fichiers avec les étiquettes universelles comme suit :
      « wsj_0010_sample.txt.pos.univ.nltk » et « wsj_0010_sample.txt.pos.univ.ref ».

   b. Utiliser le programme Python « evaluate.py » pour évaluer l'analyseur morpho-syntaxique de la plateforme NLTK selon les étiquettes universelles

      ```
      python evaluate.py wsj_0010_sample.txt.pos.univ.nltk
      wsj_0010_sample.txt.pos.univ.ref
      ```

   c. Quelles conclusions peut-on avoir à partir de ces deux évaluations ?

      Note:
      L'analyseur morpho-syntaxique de la plateforme NLTK utilisent les étiquettes du Penn TreeBank.

2. Utilisation de la plateforme NLTK pour l'analyse syntaxique

1. Ecrire un programme Python utilisant le package `parse` pour extraire les mots composés (chunks) ayant la structure syntaxique Déterminant-Adjectif-Nom (`grammar = "Compound: {<DT>?<JJ>*<NN>}"`) présents dans le texte du fichier `wsj_0010_sample.txt`. Le résultat de ce module sera mis dans le fichier `wsj_0010_sample.txt.chk.nltk`.

   Note :
   Un exemple d'utilisation du package `parse` se trouve sur le lien https://www.guru99.com/pos-tagging-chunking-nltk.html.

2. Généraliser le programme Python précédent pour extraire les mots composés (chunks) compatibles avec les structures syntaxiques ci-dessous :

   Adjectif-Nom
   Nom-Nom
   Adjectif-Nom-Nom
   Adjectif-Adjectif-Nom

Note :
Il est recommandé d'utiliser un fichier déclaratif contenant ces structures syntaxiques.

3. Utilisation de la plateforme NLTK pour l'extraction d'entités nommées

1. Ecrire un programme Python utilisant le package ne_chunk pour extraire les entités nommées présentes dans le texte du fichier `wsj_0010_sample.txt`. Le résultat de ce module sera mis dans le fichier `wsj_0010_sample.txt.ne.nltk`.

   Note :
   Un exemple d'utilisation du package ne_chunk se trouve sur le lien https://pythonprogramming.net/named-entity-recognition-nltk-tutorial/.

2. Ecrire un programme Python permettant de convertir les étiquettes des entités nommées produites par le module en étiquettes suivantes :

   O: Words that are not named entities and referred to as 'Other'.
   B-PERS: Beginning of Person Name.
   I-PERS: Inside of Person Name.
   B-ORG: Beginning of Organization Name.
   I-ORG: Inside of Organization Name.
   B-LOC: Beginning of Location Name.
   I-LOC: Inside of Location Name.
   B-MISC: Beginning of MiscellaneousWord.
   I-MISC: Inside of MiscellaneousWord.

   Conversion des étiquettes NLTK en étiquettes standard:

ORGANIZATION - Georgia-Pacific Corp., WHO	=> **ORG**
PERSON - Eddy Bonte, President Obama	=> **PERS**
LOCATION - Murray River, Mount Everest	=> **LOC**
DATE - June, 2008-06-29	=> **MISC**
TIME - two fifty a m, 1:30 p.m.	=> **MISC**
MONEY - 175 million Canadian Dollars, GBP 10.40	=> **MISC**
PERCENT - twenty pct, 18.75 %	=> **MISC**
FACILITY - Washington Monument, Stonehenge	=> **ORG**
GPE - South East Asia, Midlothian	=> **LOC**

   Exemple:
   ORGANIZATION National/NNP Association/NNP

   Ce qui était prévu initialement:
   National **B-ORG**
   Association **I-ORG**

   Ce qu'il faut faire:
   National **ORG**