POLYTECH PARIS-SUD

**Et5-Info**

TP Traitement Automatique de Langue

# Analyse linguistique

## Rappel sur les principales étapes et ressources de l'analyse linguistique à base de règles

L'analyse linguistique à base de règles repose sur des ressources généralement construites à la main. L'objectif étant le transfert de l'expertise des linguistes pour disposer des lexiques et des règles nécessaires au fonctionnement des outils de TAL.

Certains modules de l'analyse linguistique sont génériques dans la mesure où ils peuvent assurer le traitement de la majorité des langues traitées. D'autres, plus spécifiques, ne sont utilisés que dans des cas bien précis définis selon la langue à traiter. Une analyse linguistique standard se compose des modules suivants :

1. **Tokenisation :** 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 :** 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 :** 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 :** 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.

## Installation de la plateforme d'analyse linguistique open source LIMA (1 heure)

Avant de démarrer l'installation de la plateforme LIMA sur un poste équipé d'une distribution Linux (de préférence Ubuntu 16.04 LTS), il faudrait préparer l'environnement de développement en installant les outils suivants :

- Locales (locale-gen en_US.UTF-8)
- build-essential
- cmake
- g++
- python
- python-nltk
- gawk
- qt5-default
- libqt5xmlpatterns5
- libqt5xmlpatterns5-dev
- qttools5-dev
- libboost-all-dev
- libenchant-dev
- mesa-common-dev
- libgl1-mesa-dev
- libglu1-mesa-dev
- libasan0
- git

**Installation :**

1. Récupérer les source de LIMA: https://github.com/aymara/lima
2. Suivre les instructions pour l'installation de LIMA: https://github.com/aymara/lima/wiki

## Exercices

**Note :** Les exercices 2, 3 et 4 peuvent être faits sans avoir installé la plateforme LIMA (Exercice 1).

## 1. Utilisation de la plateforme LIMA pour l'analyse de textes

1. Editer le fichier « lima-lp-eng.xml » (../Dist/master/release/share/config/lima) et identifier les principaux modules composant la plateforme d'analyse linguistique LIMA (voir la section /* Definition of pipelines */).
2. Fixer les variables d'environnement : Sur le répertoire ../lima, lancer `source setenv-lima.sh -m release`
3. Lancer LIMA sur le fichier « wsj_0010_sample.txt » : `analyzeText --language=eng --pipeline=main wsj_0010_sample.txt`
4. Quels sont les résultats produits par cette analyse ?
5. Activer dans le fichier « lima-lp-eng.xml » les loggers "specificEntitiesXmlLogger" et "disambiguatedGraphXmlLogger"

6. Lancer à nouveau LIMA sur le fichier « wsj_0010_sample.txt » et observer les sorties produites : `analyzeText --language=eng --pipeline=main wsj_0010_sample.txt`
7. Rediriger le résultat d'analyse vers le fichier « wsj_0010_sample.txt.conll » : `analyzeText --language=eng --pipeline=main wsj_0010_sample.txt > wsj_0010_sample.txt.conll`

## 2. Extraction d'entités nommées

A partir des sorties de l'analyseur LIMA « wsj_0010_sample.txt.se.xml » ou « wsj_0010_sample.txt.conll », écrire un programme Python permettant de représenter les entités nommées sous le format suivant :

| Entité nommée | Type | Nombre d'occurrences | Proportion dans le texte (%) |
|---|---|---|---|

**Exemple :**

| | | | |
|---|---|---|---|
| Indianapolis | LOCATION | 1 | 14 (1/7) |

**Notes:**

- Mettre le résultat de la transformation du fichier « wsj_0010_sample.txt.disambiguated.xml » ou « wsj_0010_sample.txt.conll » dans le fichier « wsj_0010_sample.txt.ner.lima ».
- Appliquer cette transformation sur le fichier « formal-tst.NE.key.04oct95_sample.txt.conll » ou « formal-tst.NE.key.04oct95_sample.txt.se.xml ».

## 3. Analyse morpho-syntaxique

A partir de la sortie de l'analyseur LIMA « wsj_0010_sample.txt.disambiguated.xml » ou « wsj_0010_sample.txt.conll », écrire un programme Python permettant de représenter les étiquettes morpho-syntaxiques sous le format « Mot_Etiquette ».

**Exemple :**

Pour la phrase « When it's time for their biannual powwow, the nation's manufacturing titans typically jet off to the sunny confines of resort towns like Boca Raton and Hot Springs.", nous avons la representation suivante:

When_WRB it_PRP 's_VBZ time_NN for_IN their_PRP$ biannual_JJ powwow_NN ,_, the_DT nation_NN s_POS manufacturing_VBG titans_NNS typically_RB jet_VBP off_RP to_TO the_DT sunny_JJ confines_NNS of_IN resort_NN towns_NNS like_IN Boca_NNP Raton_NNP and_CC Hot_NNP Springs_NNP ._.

**Note:** Mettre le résultat de la transformation du fichier « wsj_0010_sample.txt.disambiguated.xml » ou « wsj_0010_sample.txt.conll » dans le fichier « wsj_0010_sample.txt.pos.lima ».

## 4. Evaluation de l'analyse morpho-syntaxique

1. **Evaluation à l'aide des étiquettes Penn TreeBank (PTB) :** Utiliser le programme Python « evaluate.py » pour évaluer l'analyseur morpho-syntaxique de la plateforme LIMA. L'évaluation se fait sur les fichier dans le nouveau format (wsj_0010_sample.txt.pos.lima ) (python evaluate.py wsj_0010_sample.txt.pos.lima wsj_0010_sample.txt.pos.ref)

   **Exemple :**
   python evaluate.py wsj_0010_sentence.pos.lima wsj_0010_sentence.pos.ref

   **Note:** Les deux fichiers « wsj_0010_sentence.pos.lima » et « wsj_0010_sentence.pos.ref » sont fournis pour illustrer leur format respectif et le résultat de l'évaluation.

2. **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.lima » 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.lima » et « wsj_0010_sample.txt.pos.univ.ref ».

   b. Utiliser le programme Python « evaluate.py » pour évaluer l'analyseur morpho-syntaxique de la plateforme LIMA selon les étiquettes universelles (python evaluate.py wsj_0010_sample.txt.pos.univ.lima wsj_0010_sample.txt.pos.univ.ref).
   c. Quelles conclusions peut-on avoir à partir de ces deux évaluations ?

**Note:** Certaines étiquettes de l'analyseur morpho-syntaxique de la plateforme LIMA ne font pas partie des étiquettes du Penn TreeBank. Il faut donc les remplacer dans le fichier « wsj_0010_sentence.pos.lima » avant de faire les deux évaluations.

- SCONJ => CC
- SENT => .
- COMMA => ,
- COLON => :

## Extension

Faire les mêmes exercices en utilisant les outils TAL du Stanford NLP Group (Stanford Named Entity Recognizer: https://nlp.stanford.edu/software/CRF-NER.html, Stanford Log-linear Part-Of-Speech Tagger: https://nlp.stanford.edu/software/tagger.html).

## Annexe: Les étiquettes morpho-syntaxiques universelles

Pour évaluer et évaluer les analyseurs morpho-syntaxiques, il est préférable d'utiliser des étiquettes universelles. Le tableau ci-dessous met en correspondance les étiquettes du Penn TreeBank et les étiquettes universelles pour l'anglais. Une seule étiquette est attribuée à chaque mot en fonction de son rôle dans la phrase et les mots sont classés à partir de huit catégories grammaticales : le verbe (VB), le nom (NN), le pronom (PR + DT), l'adjectif (JJ), l'adverbe (RB), la préposition (IN), la conjonction (CC), et l'interjection (UH).

| Etiquette Penn TreeBank | Description | Etiquette Universelle |
|---|---|---|
| CC | conjunction, coordinating | CONJ |
| CD | cardinal number | NUM |
| DT | determiner | DET |
| EX | existential there | DT |
| FW | foreign word | X |
| IN | conjunction, subordinating or preposition | ADP |
| JJ | adjective | ADJ |
| JJR | adjective, comparative | ADJ |
| JJS | adjective, superlative | ADJ |
| LS | list item marker | X |
| MD | verb, modal auxillary | VERB |
| NN | noun, singular or mass | NOUN |
| NNS | noun, plural | NOUN |
| NNP | noun, proper singular | NOUN |
| NNPS | noun, proper plural | NOUN |
| PDT | predeterminer | DET |
| POS | possessive ending | PRT |
| PRP | pronoun, personal | PRON |
| PRPDOL | pronoun, possessive | PRON |
| RB | adverb | ADV |
| RBR | adverb, comparative | ADV |
| RBS | adverb, superlative | ADV |
| RP | adverb, particle | PRT |
| SYM | symbol | X |
| TO | infinitival to | PRT |
| UH | interjection | X |
| VB | verb, base form | VERB |
| VBZ | verb, 3rd person singular present | VERB |
| VBP | verb, non-3rd person singular present | VERB |
| VBD | verb, past tense | VERB |
| VBN | verb, past participle | VERB |
| VBG | verb, gerund or present participle | VERB |
| WDT | wh-determiner | DET |
| WP | wh-pronoun, personal | PRON |
| WPDOL | wh-pronoun, possessive | PRON |
| WRB | wh-adverb | ADV |
| . | punctuation mark, sentence closer | . |
| , | punctuation mark, comma | . |
| : | punctuation mark, colon | . |
| ( | contextual separator, left paren | . |
| ) | contextual separator, right paren | . |