POLYTECH PARIS-SUD

Et5-Info

Module : Traitement Automatique des Langues - Traduction Automatique

Projet – Evaluation d'outils de TAL

Le sujet : Evaluation de deux plateformes open source d'analyse linguistique

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 évaluer les deux plateformes d'analyse linguistique que vous avez utilisées en TPs: Stanford Core NLP et NLTK.

Stanford Core NLP : est une boite à outils linguistiques utilisant l'apprentissage statistique à partir de corpus annotés.

NLTK : est une boite à outils linguistiques utilisant des approches hybrides combinant l'apprentissage automatique et des ressources linguistiques.

Consignes générales

Le but de ce projet est de montrer que vous pouvez installer, évaluer et rédiger un rapport décrivant les principaux résultats, points forts et limitations d'une plateforme open source d'analyse linguistique.

Vous pouvez utiliser un dépôt ***git*** à partager entre les membres de votre groupe pour assurer la compatibilité entre les modifications que vous faites. Ce répertoire git devra être structuré comme suit :

1. Un fichier texte nommé ***README*** contenant les indications nécessaires pour exécuter les scripts Python que vous avez développés. Vous indiquerez notamment les options pour la ligne de commande ainsi que des exemples d'utilisation.
2. Un dossier ***src*** pour les codes source de vos scripts Python. La lisibilité du code sera l'un des critères de notation (commentaires précis, concis, clairs et bien orthographiés).
3. Un dossier **doc** pour votre rapport écrit en LaTeX ou en Word et compilé en un PDF de 5 `a 7 pages nommé ***nom1-nom2-nom3.pdf*** qui décrit votre travail. Vous y décrirez l'objectif du projet, les résultats d'évaluation des deux plateformes d'analyse linguistique, les points forts et les limitations de chaque plateforme. Si vous savez comment résoudre ces limitations, mentionnez-le et décrivez vos idées en les présentant comme des pistes pour un travail futur. Vous devez également inclure une petite section qui décrit la contribution respective de chaque membre du groupe. Le rapport peut être écrit en anglais ou en français.
4. Un dossier **data** pour les données manipulées. Veillez à utiliser les mêmes noms de fichiers utilisés en TPs et en projet.
5. Un dossier **tp** pour les codes source de vos scripts Python des deux TPs. Il faut ajouter à ce dossier un ***README*** contenant les indications nécessaires pour exécuter ces scripts Python.

Vous m'enverrez par mail (nasredine.semmar@cea.fr) un lien vers votre projet dès que vous aurez créé le répertoire git. Ensuite, lorsque le répertoire est prêt pour être noté, vous m'informerez par mail également **au plus tard le dimanche 7 mars 2021**.

Remarques :

1. Pour pouvoir répondre à la 2ème question de la partie I (Evaluation de l'analyse morpho-syntaxique) vous aurez besoin de la table de correspondance entre les POS tags de la plateforme LIMA et les POS tags PTB (POSTags_LIMA_PTB_Linux.txt) ainsi que la table de correspondance entre les POS tags PTB et les POS tags Universels (POSTags_PTB_Universal_Linux.txt).
2. Le script « evaluate.py » ne fonctionne que si les deux fichiers à comparer ont les mêmes entrées (la 1ère colonne doit être la même), c'est-à-dire, le fichier résultat et le fichier de référence doivent avoir les mêmes mots sur la colonne de gauche.

I. Evaluation de l'analyse morpho-syntaxique

1. Utiliser le corpus annoté « pos_reference.txt.lima » pour extraire les phrases ayant servi pour produire ce corpus annoté et sauvegarder le résultat dans le fichier « pos_test.txt ». ***Dans ce corpus, une ligne vide indique la fin de la phrase courante.***

2. Convertir les tags du corpus annoté « pos_reference.txt.lima » en tags universels et sauvegarder le résultat dans le fichier « pos_reference.txt.univ ».

3. Lancer les deux POS taggers sur le fichier « pos_test.txt ». Les résultats doivent avoir le format du corpus annoté « pos_reference.txt.lima » (2 colonnes séparées par une tabulation) et doivent être sauvegardés respectivement dans les fichiers suivants :
   ```
   pos_test.txt.pos.stanford
   pos_test.txt.pos.nltk
   ```

4. Convertir les résultats des deux POS taggers en utilisant les étiquettes universelles (Annexe 1). Les résultats de cette conversion doivent être sauvegardés respectivement dans les fichiers suivants :
   ```
   pos_test.txt.pos.stanford.univ
   pos_test.txt.pos.nltk.univ
   ```

5. Lancer l'évaluation des deux POS taggers :

```
python evaluate.py pos_test.txt.pos.stanford.univ pos_reference.txt.univ
python evaluate.py pos_test.txt.pos.nltk.univ pos_reference.txt.univ
```

6. Quelles conclusions vous pouvez avoir à partir des résultats d'évaluation des deux POS taggers.

II. Evaluation de la reconnaissance d'entités nommées

1. Utiliser le corpus annoté « ne_reference.txt.conll » pour extraire les phrases ayant servi pour produire ce corpus annoté et sauvegarder le résultat dans le fichier « ne_test.txt ». ***Dans ce corpus, une ligne vide indique la fin de la phrase courante.***

2. Lancer les deux NE recognizers sur le fichier « ne_test.txt ». Les résultats doivent avoir le format du corpus annoté « ne_reference.txt.conll » (2 colonnes séparées par une tabulation) et doivent être sauvegardés respectivement dans les fichiers suivants :
   ```
   ne_test.txt.ne.stanford
   ne_test.txt.ne.nltk
   ```

3. Convertir les résultats des deux NE recognizers en utilisant les étiquettes CoNLL-2003 (https://www.clips.uantwerpen.be/conll2003/ner/ -Annexe 2-). Les résultats de cette conversion doivent être sauvegardés respectivement dans les fichiers suivants :
   ```
   ne_test.txt.ne.stanford.conll
   ne_test.txt.ne.nltk.conll
   ```

4. Lancer l'évaluation des deux NE recognizers :

```
python evaluate.py ne_test.txt.ne.stanford.conll ne_reference.txt.conll
```

```
python evaluate.py ne_test.txt.ne.nltk.conll ne_reference.txt.conll
```

5. Quelles conclusions vous pouvez avoir à partir des résultats d'évaluation des deux NE recognizers.

Annexe 1 : Correspondances entre les étiquettes du Penn TreeBank et les étiquettes universelles pour la désambiguïsation morpho-syntaxique.

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	.

Annexe 2 : Etiquettes CoNLL-2003 pour la reconnaissance es entités nommées

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.