Travaux Dirigés d'informatique linguistique n°7

Cours d'Introduction à l'informatique linguistique

—Licence Informatique 3ème année—

Chunking

Ce TP est dédié à la segmentation en constituants simples non récursifs (ou "chunks").
En particulier, le but est de développer un petit chunker pour le français.

---

1 Ressources

Pour cette séance, nous vous fournissons un certain nombre de ressources se trouvant dans le répertoire suivant :
`http://igm.univ-mlv.fr/~mconstan/enseignement/l3/infolingu/data/`

- `tagging.py`, un module python qui permet de charger et manipuler des textes étiquetés au format `Treetagger`,
- `text-for-chunking.txt`, un texte encodé en UTF-8.

2 Introduction au chunking avec NLTK

1. Adapter pour le français l'exemple ci-dessous de repérage et de balisage de chunks nominaux.

   ```
   >>> from nltk_lite import chunk
   >>> tagged_text = "the/DT little/JJ cat/NN sat/VBD on/IN the/DT mat/NN"
   >>> input = chunk.tagstr2tree(tagged_text)
   >>> cp = chunk.Regexp("NP: {<DT>?<JJ>*<NN>}")
   >>> print cp.parse(input)
   ```

   Pour plus d'informations sur le chunking avec NLTK, vous pouvez lire le document suivant :

   `http://nltk.sourceforge.net/lite/doc/en/chunk.html`

2. Repérer à la main les chunks nominaux et verbaux du texte `text-for-chunking.txt`.
3. Repérer et baliser automatiquement ces chunks. Pour cela, vous utiliserez la classe `taggedText` du module `tagging.py` pour charger un texte brut en l'étiquetant avec `TreeTagger`. Vous utiliserez la méthode `getTextForChunking` pour récuperer le texte étiqueté au bon format pour le chunking. Puis adapter l'expression rationnelle précédente.

3 Chunking par une procédure en cascade

Une procédure de segmentation en chunks est en général itérative. Chaque étape dépend de la précédente. Par exemple, l'étape de reconnaissance des chunks nominaux (`XN`) est précédée de l'étape de reconnaissance des chunks adjectivaux (`XA`) :

```
XA-> Adv? Adj
XN-> Det XA XN
```

Le code en python ci-dessous montre un exemple d'une telle procédure en cascade.

```
>>> grammar = r"""
... NP: {<DT>?<JJ>*<NN.*>+} # noun phrase chunks
... VP: {<TO>?<VB.*>}       # verb phrase chunks
... PP: {<IN> <NP>}              # prepositional phrase chunks
... """
>>> cp = chunk.Regexp(grammar)
>>> print cp.parse(input)
```

L'expression `NN.*` représente tous les noms (`NN`, `NNS`, ...).

Le but de cet exercice est de construire un chunker pour le français.

1. Reconnaître les chunks adjectivaux (`grand`, `très grand`, `vraiment très grand`)
2. Reconnaître les chunks nominaux et les chunks prépositionnels
3. Élaborer une cascade pour reconnaître les chunks verbaux complexes (`a beaucoup mangé`, `a pu manger`)
4. Tester votre chunker sur des dépêches AFP. Quels problèmes pouvez-vous identifier ? Quelles en seraient les solutions ?