Projet BàO
Présentation de TreeTagger
(mise en œuvre sous windows)

Sommaire

1. Préambule :

URL
http://www.ims.uni-stuttgart.de/projekte/corplex/DecisionTreeTagger-de.html

The TreeTagger is a tool for annotating text with part-of-speech and lemma information which has been developed within the TC project at the Institute for Computational Linguistics of the University of Stuttgart. The TreeTagger has been successfully used to tag German, English, French, Italian, Greek and old French texts and is easily adaptable to other languages if a lexicon and a manually tagged training corpus are available.

On se reportera aux informations en ligne sur le site du **TreeTagger** pour compléter les informations données *infra* (installation et utilisation sous Unix par exemple). On pourra aussi utiliser les ressources disponibles en ligne pour compléter l'installation de **TreeTagger.**

A noter (et à tester) :
Une version de **TreeTagger** *online* est disponible à cette adresse :

http://www.cele.nottingham.ac.uk/~ccztk/treetagger.php

Il s'agit d'une application en Flash permettant d'étiqueter des textes de moins de 1000 mots.

2. Les jeux d'étiquettes du TreeTagger

Observez d'abord le jeu d'étiquettes pour le français (*cf infra*) et le détail de ces étiquettes.

- Relevez les différentes catégories considérées par ce jeu d'étiquettes.
- Quelles informations plus détaillées (que juste la catégorie) peut-on avoir sur les noms ?
- Quelles informations plus détaillées (que juste la catégorie) peut-on avoir sur les verbes ?
- Quelles informations plus détaillées (que juste la catégorie) peut-on avoir sur les pronoms ?
- Y a-t-il une ou plusieurs étiquettes pour la ponctuation ? Si plusieurs, combien ?
- A quoi, d'après vous, correspond, dans les textes, l'étiquette SENT ?

3. Exploration du répertoire TreeTagger

Le répertoire **TreeTagger** contient deux fichiers : *INSTALL-treetagger* et *README-treetagger* qui contiennent des informations respectivement sur le **TreeTagger** et son installation.

Petites manipulations :
- Lire le fichier *INSTALL-treetagger*.
- Idem pour le fichier *README-treetagger*.
- Dans *INSTALL-tree,* allez au point 7 développé dans ce fichier. Testez la manipulation décrite. Pour cela, il vous faut vous rendre dans le répertoire dans lequel se trouve le fichier *INSTALL-tree* dans l'*Invite de commandes*.

Le répertoire `bin` contient : les programmes utilisés pendant l'exécution du **TreeTagger**.

Le répertoire `lib` contient : tous les fichiers de paramètres qui permettent l'étiquetage dans une langue particulière. Quelles langues est-il possible de traiter avec le **TreeTagger** installé sur votre machine ?

Le répertoire `cmd` contient :
- un programme Perl utilisé par le **TreeTagger** avant la phase d'étiquetage : le programme de *tokenisation* qui segmente le texte en *tokens* (un *token* par ligne). Vous pouvez ouvrir ce programme dans un éditeur de texte (*bloc-notes* ou *wordpad*) pour observer son contenu. Une version de ce programme est disponible pour le français (`tokenise-fr.pl`)
- Un programme Perl pour convertir les résultats de **TreeTagger** au format XML (programme `treetagger2xml.pl`)

4. Utilisation sous Windows *via* l'*Invite de commandes* DOS ou *via* Cygwin

Le **TreeTagger** est un programme qui ne possède pas d'interface graphique. Il faut donc lancer le programme à l'aide de commandes dans l'*Invite de commandes* DOS ou dans une fenêtre de commandes de type *Cygwin*.

Une fois que l'*Invite de commandes* est ouverte, vous devez spécifier le chemin des programmes du **TreeTagger** pour que ceux-ci soient exécutables quelque soit le dossier dans lequel vous vous trouvez. Pour cela, tapez la commande suivante suivie de « Entrée » :

```
set PATH=C:\TreeTagger\bin;%PATH%
```

5. Utilisation du TreeTagger sur un texte en français

5.1. Syntaxe générale

La commande d'étiquetage CLASSIQUE avec **TreeTagger** est la suivante :

```
tree-tagger [options] <parametres> <textein> <texteout>
```

- Le premier argument est le fichier paramètre (ici *french.par* dans le répertoire *lib*)…
- Le deuxième argument est le texte à étiqueter (**avec un mot par ligne**)…
- Le troisième argument est le nom du fichier de sortie…

5.2. En pratique

(*cf* fichier *README-ETIQUETAGE*)

Pour lancer **TreeTagger** sur un fichier texte (avec éventuellement des balises type SGML ou Lexico3) :

- **Lancer la fenêtre de commande** puis se placer dans le répertoire contenant **TreeTagger**.
- **Placer dans ce répertoire le fichier à étiqueter**.
- **Lancer la commande** :

(console unix type cygwin)

```
perl tokenise-fr.pl fichier-a-etiqueter.txt | ./bin/tree-tagger.exe
./lib/french.par -lemma -token -sgml > resultat-etiquetage.txt
```

(invite de commande DOS)

```
perl tokenise-fr.pl fichier-a-etiqueter.txt | bin\tree-tagger.exe
lib\french.par -lemma -token -sgml > resultat-etiquetage.txt
```

Dans cet exemple :

- Le fichier à étiqueter s'appelle : `fichier-a-etiqueter.txt`
- Le programme `tokenise-fr.pl` « segmente » le texte à étiqueter et produit un flux de sortie (avec un mot par ligne) qui est étiqueté par **TreeTagger**. Le résultat est stocké dans un fichier en sortie.

TreeTagger utilise ici les 3 options :

```
-lemma : Prints the lemma as well
-token : Prints the token as well
-sgml : Don't tag SGML annotations, i.e. lines
starting with '<' and ending with '>'
```

Remarque : On peut aussi ajouter l'option `-no-unknown` (après `-sgml`) pour ne pas avoir en sortie l'affichage <unknown> si le lemme n'est pas connu.

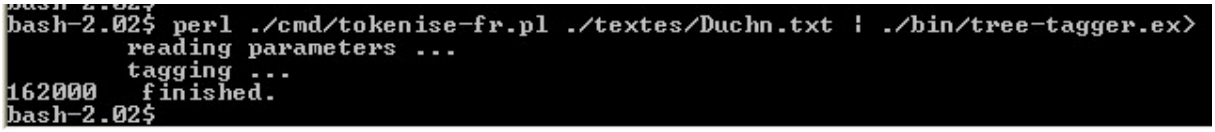
- Le résultat est contenu dans : `resultat-etiquetage.txt`

En sortie les zones textuelles hors balises seront constituées sur chaque ligne par : 1 mot, 1 tabulation, sa catégorie, 1 tabulation, son lemme.

5.3. Exemple

Exemple de traitement sur le fichier Duchn.txt contenu dans le répertoire de **TreeTagger** :

Commande lancée dans la fenêtre Cygwin (et à tester…) :




```
perl tokenise-fr.pl ./textes/Duchn.txt | ./bin/tree-tagger.exe
./lib/french.par -lemma -token -sgml > resultat-etiquetage.txt
```

- Le fichier de travail : `Duchn.txt`
- Le résultat de l'étiquetage : `resultat-etiquetage.txt`

5.1. Reformatage XML

On peut ensuite reformater le résultat d'étiquetage au format XML de la manière suivante :

```
perl treetagger2xml.pl resultat-etiquetage.txt
```

On obtiendra en sortie un fichier (au format XML) nommé `resultat-etiquetage.txt.xml`

6. Cas d'ambiguïtés et erreurs d'étiquetage

Ouvrez le fichier `resultat-etiquetage.txt` qui est la version étiquetée du texte initial.

Recherchez au moins deux exemples des erreurs d'étiquetage suivantes :

- Mots mal orthographiés ou inconnus : quelle étiquette est utilisée pour marquer ces formes ? Est-ce toujours la même qui est utilisée ? Quel est le lemme donné pour ces formes ?
- Erreurs de découpage : comment se comporte le **TreeTagger** ? A quel moment intervient l'erreur ?
- Cas où le sens est nécessaire à la désambiguïsation : comment se comporte le **TreeTagger**, comment résout-il le problème ?

A votre avis, quelles solutions peuvent être apportées à l'étiqueteur pour résoudre ces problèmes ?

Quelle est la conséquence de ces erreurs pour l'étiquetage du reste de la phrase dans laquelle elles apparaissent ?

7. Les options du TreeTagger

Les options du **TreeTagger** sont listées dans le fichier *README-treetagger* qui se trouve dans le dossier d'installation.

```
-token: Prints the token as well.
-lemma: Prints the lemma as well.
-sgml:  Don't tag SGML annotations, i.e. lines starting with '<' and ending
        with '>'.
-threshold <p>: Print all tags with a probability higher than <p> times the
        probability of the best tag.
-prob:  Print tag probabilities (requires option -threshold)
-no-unknown: Print the token rather than <unknown> for unknown lemmas
-quiet: Don't print status messages
-pt-with-lemma: If this option is specified, then each pretagging tag
        (see above) has to be followed by a whitespace and a lemma.
-pt-with-prob: If this option is specified, then each pretagging tag
        (see above) has to be followed by whitespace and a tag probability
        value. If -pt-with-prob and -pt-with-lemma have been specified,
        then each pretagging tag is followed by a probability and a lemma
        in that order.
-files f: Read the names of input and output files pairwise from the
        file f. The format of f is the lexicon file format described below.
-lex f: Read auxiliary lexicon entries from the file f.
-eos-tag <tag>: The SGML tag <tag> signals the end of a sentence.
        This option implies the option -sgml
```

8. TreeTagger – Les étiquettes pour l'anglais : *The Penn Treebank*

Tagset

CC Coordinating conjunction (and,but,or...)
CD Cardinal Number
DT Determiner
EX Existential *there*
FWForeign Word
IN Preposition or subordinating conjunction
JJ Adjective
JJR Adjective, comparative
JJS Adjective, superlative
LS List Item Marker
MD Modal (can, could, might, may...)
NN Noun, singular or mass
NNP Proper Noun, singular
NNPS Proper Noun, plural
NNS Noun, plural
PDT Predeterminer (all, both ... when they precede an article)
POS Possessive Ending (Nouns ending in 's)
PRP Personal Pronoun (I, me, you, he...)
PRP$ Possessive Pronoun (my, your, mine, yours...)
RB Adverb (Most words that end in -ly as well as degree words like quite, too and very)
RBR Adverb, comparative (Adverbs with the comparative ending -er, with a strictly comparative meaning)
RBS Adverb, superlative
RP Particle
SYM Symbol (Should be used for mathematical, scientific or technical symbols)
TO *to*
UH Interjection (uh, well, yes, my...)
VB Verb, base form (subsumes imperatives, infinitives and subjunctives)
VBD Verb, past tense (includes the conditional form of the verb to be)
VBG Verb, gerund or persent participle
VBN Verb, past participle
VBP Verb, non-3rd person singular present
VBZ Verb, 3rd person singular present
WDT Wh-determiner (which, and *that* when it is used as a relative pronoun)
WP Wh-pronoun (what, who, whom...)
WP$ Possessive wh-pronoun
WRB Wh-adverb (how, where why)

Punctuation Tags

#
$
''
(
)
,
.
:
``

9. TreeTagger – Les étiquettes pour le français

```
ABR Abreviation
ADJ Adjectif
ADV Adverbe
DET:ART Article
DET:POS Pronom Possessif (ma, ta, ...)
INT Interjection
KON Conjunction
NAM Nom Propre
NOM Nom
NUM Numéral
PRO Pronom
PRO:DEM Pronom Démonstratif
PRO:IND Pronom Indefini
PRO:PER Pronom Personnel
PRO:POS Pronom Possessif (mien, tien, ...)
PRO:REL Pronom Relatif
PRP Préposition
PRP:det Préposition + Article (au,du,aux,des)
PUN Ponctuation
PUN:cit Ponctuation de citation
SENT Balise de phrase
SYM Symbole
VER:cond Verbe au conditionnel
VER:futu Verbe au futur
VER:impe Verbe à l'impératif
VER:impf Verbe à l'imparfait
VER:infi Verbe à infinitif
VER:pper Verbe au participe passé
VER:ppre Verbe au participe présent
VER:pres Verbe au présent
VER:simp Verbe au passé simple
VER:subi Verbe à l'imparfait du subjunctif
VER:subp Verbe au présent du subjunctif
```