TP Word Sense Disambiguation

Ce devoir a pour but de maîtriser et comparer différentes techniques de désambiguïsation lexicale. Elles seront évaluées sur le corpus anglais **TWA sense tagged data** (Mihalcea, 2003), un corpus étiqueté divisé en 6 fichiers, correspondant respectivement à 6 mots cibles ("bass", "crane", "motion", "palm", "plant", et "tank").
Chacun de ceux-ci est ambigu entre deux sens possibles (p.ex., "plant" est ambigu entre "plant%factory" et "plant%living").
Chacun des 6 sous-corpus est composé d'exemples (balise « instance ») qui comportent:

(i) un ID
(ii) un lemme
(iii) un contexte
(iv) et une étiquette-sense pour le lemme dans le contexte.

Pour évaluer sur des données non vues à l'entraînement, vous utiliserez un split 80/20 sur ce corpus: 80% des données serviront à l'entraînement/développement des différents systèmes, alors que 20% servira au test (voir méthode fournie `utils/data_split`).

Une API pour la lecture et manipulation des exemples TWA (twa.py), ainsi que certaines méthodes utilitaires (utils.py) vous sont fournis pour ce TP. Un squelette de système de désambiguïsation est également à votre disposition dans wsd.py.
(Libre à vous d'utiliser ce code ou pas)

A rendre par mail **avant le 25 novembre** : archive tar.gz avec votre code, tableau de résultats et discussion.

1 Baselines

Implémenter deux systèmes "baseline" non contextuels, à savoir la baseline qui utilise le sens le plus fréquent (dans le corpus d'entraînement) et la baseline qui consiste à prendre le premier sens de Wordnet (le premier sens de chaque lemme est fourni dans utils.py).

2 Simplified Lesk

Implémenter l'algorithme de Lesk simplifié en utilisant comme "signatures" les mots non vides appartenant aux définitions et aux exemples présents pour chaque sens dans Wordnet. Voir doc nltk + une table de correspondance entre les sens taggés dans le corpus et les sens Wordnet fournie dans utils.py.

Optionnel:
- utiliser aussi les données d'apprentissage pour enrichir les signatures

3 Naïf bayes

Implémenter un système de désambiguïsation lexicale basé sur un classifieur bayésien en utilisant comme traits de collocations les mots dans une fenêtre de 3 mots à gauche et à droite (en ne considérant pas les mots vides, cf. liste fournie dans utils.py).

Optionnel:
- faites varier la largeur de la fenêtre

4 Discussion

Reprenez les résultats des différents systèmes dans un tableau récapitulatif, comportant le nom du système et son score d'exactitude.
Quel est le meilleur système? Pourquoi pensez-vous que ce système donne de meilleures performances que les autres? Quel type d'erreurs sont commises par les différents systèmes comment pensez-vous que l'on puisse encore améliorer les performances de ces systèmes?