Troisième partie : Etiquetage morphosyntaxique automatique

Le but de cette séance est de développer un étiqueteur morphosyntaxique permettant de déterminer la nature des mots d'un texte déjà tokenisé. Les étiquettes prédites (appelées *POS* pour *Part-Of-Speech*) correspondent à la catégorie morphosyntaxique d'un mot (nom, verbe, adjectif, préposition, etc.). Etant donné qu'à une même forme peut correspondre plusieurs *POS*, par exemple *verbe* ou *nom* pour la forme *président*, il faut développer un outils de désambiguisation permettant de choisir le *POS* correct en fonction du contexte d'apparition du mot dans la phrase. Pour effectuer cette désambiguisation nous utiliserons un classifieur spécialement développé pour pouvoir traiter du texte, implémentant un algorithme de *boosting*, il s'agit de l'outils *icsiboost* développé par Benoît Favre (chercheur en TAL et membre du LIS à l'AMU).

Mise en pratique

- Partie 1 : Prise en main du classifieur
  - Vous allez utiliser l'outil *icsiboost* disponible ici : https://github.com/benob/icsiboost
  - Récupérer l'archive ZIP de l'outil et installez le dans votre répertoire de travail.
  - Regardez la documentation dans le Wiki du projet : https://github.com/benob/icsiboost/wiki
  - Faites le tutorial : https://github.com/benob/icsiboost/wiki/Tutorial
- Partie 2 : Développement d'un classifieur pour l'étiquetage en POS
  - Vous allez maintenant utiliser *icsiboost* pour choisir un POS en contexte pour un mot donné. Pour cela vous allez construire des corpus d'apprentissage et de développement pour *icsiboost* contenant des exemples de mots en contexte et de POS de référence.
  - Ces corpus seront construit à partir de la ressource *corpus-pos* que vous pourrez télécharger ici : https://pageperso.lis-lab.fr/frederic.bechet/M1_TAL_TP_data/data_corpus_pos.tgz . Cette ressource contient du texte tokenisé avec un mot sur chaque ligne. La première colonne représente un indice du mot dans la phrase, la deuxième colonne contient la forme, et la troisième colonne l'étiquette morphosyntaxique (*POS*) qui a été attribuée par un annotateur humain.
  - Le travail va consister à choisir quels traits (*features*) vont être utilisés pour décrire les données dans les fichiers *.names* et *.data* pour obtenir les meilleures performances possibles de classification.
  - La version de base (*baseline*) de votre classifieur va utiliser uniquement 3 traits : le mot à étiqueter, le mot précédent et le mot courant. Par exemple, sur la première phrase du corpus d'apprentissage :

```
1	certes	adv
2	,	ponctw
3	rien	pro
```

```
4	ne	advneg
5	dit	v
6	qu'	csu
7	une	det
8	seconde	adj
9	motion	nc
10	de	prep
11	censure	nc
12	sur	prep
13	son	det
14	projet	nc
```

le fichier *.data* d'icsiboost contiendra les lignes suivantes :

```
1 , XX , certes , !VIRGULE , adv .
2 , certes , !VIRGULE , rien , ponctw .
3 , !VIRGULE , rien , ne , pro .
4 , rien , ne , dit , advneg .
5 , ne , dit , qu' , v .
6 , dit , qu' , une , csu .
7 , qu' , une , seconde , det .
8 , une , seconde , motion , adj .
9 , seconde , motion , de , nc .
10 , motion , de , censure , prep .
11 , de , censure , sur , nc .
12 , censure , sur , son , prep .
13 , sur , son , projet , det .
14 , son , projet , de , nc .
```

On voit dans cet exemple que le mot à étiqueter se trouve dans la 3e colonne, la première colonne contient l'identifiant (à ignorer durant l'apprentissage), la deuxième colonne contient le mot précédent, la quatrième colonne le mot suivant, et enfin la cinquième colonne l'étiquette *POS* à prédire. On remarque que les virgules et les points sont remplacés par des symboles pour éviter qu'ils ne soient interprétés comme des séparateurs de colonne par *icisboost*. Enfin, pour le premier mot, le mot précédent (qui n'existe pas) est remplacé par le symbole *XX*.

- Question 1 : écrivez le fichier *names* décrivant la version *baseline* des traits du classifieur pour *icsiboost*.
- Question 2 : écrivez un programme permettant de générer les fichiers de données nécessaire à l'apprentissage d'*icsiboost* avec le format *baseline* à partir des fichiers *corpus_pos_train.txt* (pour faire le fichier d'apprentissage *data*) et *corpus_pos_dev.txt* (pour le fichier *dev*).

    - Question 3 : Lancez un apprentissage en testant les capacité de généralisation de votre modèle sur le corpus de développement. Tester différentes valeurs pour le nombre d'itérations. Que constatez-vous ?
    - Question 4 : Proposez maintenant des ensembles de traits plus riche que le mode *baseline*, par exemple en augmentant la taille du contexte, en généralisant certains mots en les remplaçant par des informations morphologiques, en rajoutant des traits décrivant la morphologie (préfixe et suffixe par exemple) du mot à traiter. Testez vos nouveaux ensembles de traits et calculez les performances de chacun d'eux sur le corpus de développement.
    - Question 5 : Vous allez maintenant utiliser un lexique telle que le *Lefff* vu dans le dernier TP pour améliorer votre classifieur. Vous allez récupérer une nouvelle version de ce lexique ici : lex_tagg_pos.txt. Vous allez maintenant rajouter comme traits pour décrire vos données une colonne qui contiendra l'ensemble des valeurs de POS que peut prendre chaque mot dans le fichier lex_tagg_pos.txt, sinon le mot recevra le trait *OOV*. Est-ce que ce trait améliore les performances ?
- Partie 3 : Application de l'étiqueteur en POS
    - Vous allez maintenant enchaîner le programme de tokénisation et de découpage en phrase du TP précédent avec le classifieur en POS développé dans ce TP. Le but est de construire une chaine de traitement qui prend en entrée du texte libre et qui produit un corpus tokénisé, découpé en phrases et étiqueté en POS avec un mot par ligne au même format que le corpus corpus_pos_train.txt
    - Question 1 : Modifier vos programmes de tokénisation et découpage en phrase pour qu'ils puissent produire un corpus au format *icisboost* avec le meilleur ensemble de traits (le plus performant) que vous avez obtenu dans la partie 2.
    - Question 2 : écrivez un programme qui prend en entrée le corpus à étiqueter produit dans la question précédente et la sortie d'*icsiboost* pour produire un corpus annoté automatiquement en *POS* au format corpus_pos_train.txt
    -