Travaux Dirigés d'informatique linguistique n°3

Cours d'Introduction à l'informatique linguistique

—Licence Informatique 3ème année—

Classification de documents et résumé automatique

Dans ce TP, vous aurez à implémenter un algorithme simple de classification de documents et de l'évaluer. Ensuite, vous devrez tester un outil simple de résumé automatique de textes.

---

1 Ressources

Toutes les ressources nécessaires pour cette séance, se trouvent dans le répertoire suivant :
`http://igm.univ-mlv.fr/~mconstan/enseignement/l3/infolingu/data/`

Vous y trouverez trois modules python (dont un avec des trous) :

- `textSpaceVector.py` qui permet de charger et manipuler des collections de documents textuels sous la forme de vecteurs,
- `categorisation.py` qui permet de classer des documents à partir d'une collection de documents déjà classés,
- `summarization.py` qui permet de résumer un texte.

Le répertoire contient également un ensemble de textes anglais (dépêches Reuter) se trouvant dans l'archive `reuter-collection.zip`

2 Classification de documents

Soient deux ensembles de textes D_{app} et D_{eval} auxquels on a assigné une liste de catégories thématiques (*coffee*, *gas*, *sugar*) et qui sont respectivement définis dans les fichiers `cat_apprentissage.txt` et `cat_evaluation.txt` du répertoire des ressources.

Chaque ligne comprend un chemin relatif de fichier, une tabulation, puis une séquence de catégories séparées par des virgules.

1. Dans votre répertoire de travail, créer un répertoire *data* dans lequel vous placerez le répertoire extrait de l'archive `reuter-collection.zip`
2. En utilisant la classe `catTextVectorCollection` du module `categorisation.py`, charger l'ensemble de documents D_{app} (`cat_apprentissage.txt`) et afficher le barycentre pour la catégorie *gas*.

3. Implanter la méthode `getBestCat(self,file)` dans la classe `catTextVectorCollection` qui calcule la meilleure catégorie pour un nouveau document donné (fichier `file`). L'algorithme utilisé sera le suivant : pour chaque catégorie, on calcule sa similarité avec le document (utilisation de la méthode `sim` de `textVector`). on garde la catégorie dont le barycentre est le plus proche du document en question.
4. Tester cet algorithme pour quelques documents de D_{eval} (`cat_evaluation.txt`)
5. Ecrire une procédure d'évaluation automatique de l'outil de classification en utilisant l'ensemble de documents D_{eval} et en calculant la précision générale. La précision est le nombre de documents bien classés sur le nombre de documents testés (ceux de D_{eval}).

Bonus Implanter la méthode `getBestCategories(self,file,ratio)` dans la classe `catTextVectorCollection` qui calcule les meilleures catégories pour un document `file`. Le paramètre `ratio` permet de déterminer le seuil à partir duquel une catégorie sera considérée comme bonne pour le document. Ce seuil est la multiplication de `ratio` par la similarité de la meilleure catégorie du document avec le document.

3 Résumé automatique

1. Charger un document en utilisant la classe `summarizationText` lui-même plongé dans une collection de textes (collection `Reuter` par exemple)
2. Faire le résumé automatiquement (méthode `summarize`)
3. Evaluer l'algorithme utilisé
4. Réitérer ces opérations pour plusieurs textes
5. Y-a-t-il des moyens d'améliorer la technique ?