Rapport de la semaine bloquée ADA -

Groupe de Haozhou Dai et Gros Alexis

Notre projet contient les paquetages suivants:

- p_chaine: pour la gestion et le découpage des chaînes de caractère
- p_classification: pour la classification des dépêches
- p_depeche: pour la gestion des dépêches
- p_generation: pour la génération automatique des lexiques
- p_lexique: pour la gestion des lexiques

Ainsi que le fichier depeches.txt qui contient les 500 dépeches de journal.

Introduction

Ce travail porte sur la création d'un classificateur automatique de dépêche. Nous devions faire un programme qui grâce à des fichiers de lexiques inférait la catégorie de l'article. Dans un deuxième temps, il était question de créer un programme d'apprentissage très simple qui déduisait la manière optimale de remplir les fichiers de lexiques en examinant la fréquence des mots dans les dépêches en fonction de la catégorie.

Nos réalisations:

Nous avons essayé d'être efficace. Ainsi, nous sommes avions terminés le projet en 8 heures. Nous avons réalisé tous les énoncés du projet, mais n'avons pas procédés aux optimisations de tri optionnelles. Nous sommes ainsi capables de générer automatiquement des fichiers de lexiques de la taille désirée au besoin.

Pour la partie 1, nous avons remplis les fichiers lexiques en parcourant le fichier depeches.txt. Ensuite, nous avons écrits une fonction Init_Lexique qui ouvrait le fichier lexique pour le remplir. Enfin, nous avons terminés en créant la fonction run qui générait un fichier de réponse respectant la consigne.

Comme conseillé, nous avons créé dans la partie 2 une fonction de génération de lexique (Generation_Lexique) qui calculait un score avant de le ramener à une valeur entre 0 et 3 via les fonctions Calcul_Scores et Poids_Score.

Nous avons créé des programmes de tests, de p1 à p8 pour valider les différentes fonctions au fur et à mesure de notre progression dans le projet.

Nous avons eu quelques soucis et bugs intéressants: les plus notables avaient traits à la gestion des fichiers. Nous avions oubliés par exemple de fermer les fichiers ouverts, ce qui amenait des bugs quand nous essayons de rassembler les programmes de classification et de génération car le même fichier était demandé plusieurs fois dans plusieurs fonction alors qu'il n'était pas fermé. Nous avons eu quelques problèmes également avec les modes d'ouverture.

Nous avons à chaque fois essayé d'écrire des versions des fonctions qui optimisent le temps processeur, en évitant les parcours de boucles inutiles. La génération des lexiques prends tout de même un certain temps: 2.3 secondes. La classification est en revanche extrêmement rapide puisqu'elle prends 1 milliseconde.

Organisation:

Pour garantir l'efficacité, nous nous sommes répartis les tâches: Alexis a commençé directement la deuxième partie tandis qu'Haozhou avançais

dans la première partie. Pour maximiser notre efficacité, nous avons utilisé un logiciel de gestion de version, git via la plateforme web github pour synchroniser nos progrès et garder une trace des différentes étapes de développement.

Résultats

Nous sommes assez mauvais pour rentrer les fichiers de lexique à la main: en effet, nous faisons moins bien que le hasard avec nos lexiques de 11 mots dans toutes les catégories sauf la politique, qui est la catégorie par défaut. En revanche, notre algorithme de classification fait 67% de recherche fructueuse avec seulement 20 mots dans les lexiques, ce qui est un score bien meilleur que le hasard.

Résultat manuel:

POLITIQUE: 98
SPORTS: 56
CULTURE: 52
ECONOMIE: 46
SCIENCE: 35
Moyenne : 57.4

Résultat automatique:

POLITIQUE: 91
SPORTS: 67
CULTURE: 64
ECONOMIE: 60
SCIENCE: 54
Moyenne : 67.2

Nous avons créé le programme de calcul du temps d’exécution pour vérifier l’efficacité de notre programme. Nous arrivons à un temps d'exécution tells que:

- La durée de génération des lexiques: 2.3 s
- La durée de la classification: ~1 milliseconde

Conclusion

Notre projet a été plutôt court, mais nous avons pu expérimenter tous les points clés de l’énoncé. Nous avons appris à travailler ensemble et à distance sans nous marcher sur les pieds en utilisant des outils de gestion de projets très importants dans le monde professionnel. Notre programme est perfectible: nous pourrions mettre plus de 20 mots dans nos lexiques pour améliorer nos statistiques et revoir notre algorithme de calcul des scores pour ne rentrer dans le lexique que les mots qui ciblent spécifiquement cette catégorie. Et évidemment, procéder au tri des fichiers comme indiqué dans l’extension.