Rapport

Chassande Antoine – You Adrien

Hypothèses de départ

Choix du langage

Nous avons choisi d'utiliser **Python**, en particulier pour leur bonne gestion des objets de type dictionnaire.

Forme des requêtes

Afin de simplifier le fonctionnement du programme, nous avons fixé un format pour les requêtes.

- **Requêtes booléennes** : forme normale conjonctive
  `(A OR B OR NOT C) AND (D OR E)...`
- **Requêtes vectorielles** :
  `(A B C)`

Organisation du code

Nous avons organisé notre programme en modules.

InfoResearch

Contient le cœur du programme. Les fonctions sont appelées à partir de ce fichier. C'est aussi dans ce fichier que nous avons rentré nos requêtes.

DictioManager

Contient les fonctions de traitement des dictionnaires (création du dictionnaire contenant, pour chaque document, les mots indexés et leur nombre d'occurrence, création du dictionnaire inversé) et les fonctions d'accès.

VectorialModule

Contient la fonction retournant les documents répondant à une requête vectorielle, triés selon la similarité.

PonderatorModule
Contient les fonctions implémentant les méthodes de pondération en fonction de l'occurrence des mots.

BooleanModule
Contient la fonction retournant les documents répondant à une requête booléenne.

ProbabilistModule
Contient la fonction retournant les documents répondant à une requête vectorielle, en se basant sur le modèle probabiliste, triés selon la similarité.

EvaluationModule
Contient les fonctions permettant l'évaluation de la performance (temps d'exécution, taille sur le disque) et de la pertinence (précision, rappel, E-mesure, F-mesure, …).

Courbe
Contient les fonctions de création d'une liste de requêtes à partir de query.txt et d'un dictionnaire de documents pertinents étant donné l'Id d'une des requêtes ci-dessus.

Quelques résultats

Lors de nos tests, voici quelques temps d'exécution :

- **Création du dictionnaire** contenant, pour chaque document, les mots indexés et leur nombre d'occurrence : 8s maximum
- **Création du dictionnaire inverse** à partir du dictionnaire précédent : 1.7s max
- **Création du dictionnaire avec FreqNorm** à partir du dictionnaire : 1.7s max
- **Création du dictionnaire avec TF-IDF** à partir du dictionnaire : 3s max
- **Requête vectorielle** : 0.06s
- **Requête booléenne** : 1s
- **Requête probabiliste** : 1s

Et voici quelques données de taille :

- Taille du **dictionnaire** contenant, pour chaque document, les mots indexés et leur nombre d'occurrence : 196 704 octets
- Taille du dictionnaire inversé : 393 312 octets

Problèmes rencontrés

Nous n'avons pas réussi à obtenir la courbe précision-rappel. Cela doit probablement venir d'un problème d'implémentation de l'une des fonctions. Le temps très court de l'exécution du modèle vectoriel est étrange, et le problème peut donc venir de là. Cependant, nous ne sommes pas parvenus à corriger cette erreur.