Recherche d'Information

M2 DAC

TME 4. Modèles Probabilistes

Au cours de ce TME, le travail à réaliser est d'implémenter et d'expérimenter deux modèles vus en cours: le modèle de langue d'une part et le modèle BM25 (ou Okapi) d'autre part. Ces modèles corresponderont à des éléments du package **modeles**.

1 Modèle de langue

Les modèles de langue pour la RI se basent sur une hypothèse générative qui suppose que chaque document est généré par un modèle qui lui est propre. On a aors un modèle génératif par document qui détermine une distribution des probabilités d'occurence de chacun de ses constituants. Ici, par soucis de simplicité, nous considèrerons des modèles de langue unigrammes, où la probabilité d'observation d'une séquence de mots correspond au produit des probabilités d'occurence de chacun des mots la composant. Ainsi, $P_M(s)$ exprimant la probabilité d'observer la séquence de n mots $s = s_1, s_2, ..., s_n$ selon le modèle M est définie par:

$$P_M(s) = \prod_{i=1}^{n} p_M(s_i)$$

où $p_M(s_i)$ correspond à la probabilité d'occurence du terme s_i selon le modèle M (on ne considère ici que les termes s_i appartenant au vocabulaire, i.e., ceux qui apparaissent dans l'index de la collection).

L'idée est alors de définir un modèle de langue M_d par document d, afin de déterminer un score d'appariement $f(d, q)$ de chaque document d avec une requête q dépendant de la probabilité $P_{M_d}(q)$ d'observer q selon le modèle M_d du document d. Puisque ce qui nous intéresse ici est un classement des documents selon leur score d'appariement (et non les probabilités exactes d'observation de la requête selon les différents modèles des documents), il est d'usage de simplifier le problème en considérant le logarithme de la probabilité $P_{M_d}(q)$:

$$f(q, d) = \log P_{M_d}(q) = \sum_{t \in q} tf(t, q) \log p_{M_d}(t)$$

La détermination d'un modèle de langue M_d se fait par maximum de vraissemblance: Le modèle M_d est celui qui rend le plus vraissemblable le document d. Ainsi, M_d est défini par:

$$M_d = \arg\max_M \log P_M(d) = \arg\max_M \sum_{t \in d} tf(t,d) \log p_M(t)$$

$$\Rightarrow p_{M_d}(t) = \frac{tf(t,d)}{L(d)}$$

Ce genre de modèle pour la RI présente de nombreux atouts:

- favorise les documents ayant plus de termes de la requête ;
- favorise les documents ayant des termes différents de la requête ;
- pénalise les documents longs ;

Néanmoins, il possède un défaut majeur: si un terme d'une requête q n'appartient pas à un document d, la probabilité d'observation de cette requête selon le modèle de langue du document d est nulle (et le score d'appariement $f(d,q)$ est alors égal à $-\infty$), indépendamment du nombre d'occurences des autres termes de la requête dans le document. Pour corriger ce problème, il est d'usage d'appliquer un "lissage" de $f(d,q)$, qui permet d'éviter qu'un terme absent annule toute la contribution des autres termes de la requête dans le score d'un document. Parmi de nombreux autres, un lissage possible est de considérer conjointement la probabilité d'observer le terme selon le modèle du document et celle de l'observer selon un modèle M_C défini sur l'ensemble du corpus C:

$$f(d,q) = \lambda \log P_{M_d}(q) + (1-\lambda) \log P_{M_C}(q)$$

avec $\lambda \in [0,1]$ un paramètre déterminant le poids du terme de lissage dans le score d'appariement du modèle de RI.

Définir un classe $LanguageModel$, héritant de la classe $IRmodel$, et définissant ce genre de modèle pour notre plateforme d'expérimentation pour la RI. Dans un premier temps on déterminera le paramètre de lissage λ à la main, selon les observations de performance que l'on pourra observer.

2 Modèle BM25 (Okapi)

La mesure Okapi constitue une des heuristiques de recherche les plus performantes. La mesure est issue d'une formulation probabiliste du problème de recherche d'information où les documents doivent être ordonnés selon leur probabilité d'être pertinent pour la requête considérée. Considérant une hypothèse d'indépendance des mots d'un documents, le score d'appariement pour un document d avec une requête q peut alors s'écrire de manière équivalente (avec $Pert(q)$ l'ensemble des documents pour la requête q):

$$f(d,q) = \log \frac{P\big(d' = d \big| d' \in Pert(q)\big)}{P\big(d' = d \big| d' \notin Pert(q)\big)}$$

$$= \sum_t \log \frac{P\big(tf(t,d') = tf(t,d) \big| d' \in Pert(q)\big)}{P\big(tf(t,d') = tf(t,d) \big| d' \notin Pert(q)\big)}$$

Considérant que $P\big(tf(t,d') = .\big|d' \in Pert(q)\big)$ suit une loi de poisson et après diverses simplifications et une prise en compte de la longueur des documents, le score d'appariement de la mesure Okapi (dans sa version la plus simple) est définie par:

$$f(d,q) = \sum_{t \in q} idf'(t) \frac{(k_1+1)tf(t,d)}{k_1\left((1-b)+bL(d)/L_{moy}\right)+tf(t,d)}$$

avec :

- $idf'(t) = \max\big(0, \log \frac{N-df(t)+0.5}{df(t)+0.5}\big)$ "probabilistic idf" ;
- L_{moy} : longueur moyenne des documents ;
- k_1 entre 1 et 2, $b \approx 0.75$
  valeurs à déterminer en fonction du corpus.

3 Optimisation des paramètres

Après avoir séparé les ensembles de requêtes en deux ensembles Q_{train} et Q_{test} distincts, rechercher de manière automatique les valeurs optimales des paramètres des différents modèles développés (λ pour les modèles de langue et $k1$ et b pour BM25) sur l'ensemble de requêtes d'entrainement de l'ensemble Q_{train} (les requêtes de Q_{test} seront réservées pour l'évaluation / la comparaison des modèles).

Il s'agit alors de mettre en place une recherche par "GridSearch", qui consiste à définir une grille de valeurs à tester pour chaque paramètre (implique de définir un intervale et un pas de déplacement entre chaque valeur à tester) et expérimenter chaque combinaison possible des paramètres d'un même modèle selon les grilles ainsi définies sur les requêtes d'apprentissage (on conservera les combinaisons permettant d'observer le meilleur MAP sur cet ensembe d'apprentissage).