Moteurs de Recherche

Cours 2

Heuristiques de Recherche

Université Pierre et Marie Curie (UPMC)
Laboratoire d'Informatique de Paris 6 (LIP6)

Plan

- Quelques rappels ;
- Fonctions d'appariement ;
- Modèles de recherche :
  - $tf.idf$;
  - Modèle vectoriel ;
  - Modèles de langue pour la RI ;
  - *Probability Ranking Principle* et BM25 ;
- Prochain cours : évaluation des Moteurs de Recherche.

Rappels : recherche sur des documents plats

- Un MR aide un utilisateur à trouver l'information qui l'intéresse
  définition pratique: un MR *ordonne* des documents selon leur *pertinence* par rapport à un *besoin d'information* ;

- un *besoin d'information* est exprimé sous forme d'une *requête* ;
  la requête est composée de *mots-clefs*
  (formule booléenne, liste de mot-clefs, phrase) ;

- un *index* est créé à partir des *termes* utilisés dans les documents
  (a) la première difficulté est de déterminer le vocabulaire
  $\Leftrightarrow$ déterminer les règles de *tokenisation* ;
  (b) l'index est indépendant de la requête ;

- une *fonction d'appariement* donne un *score* à chaque document
  (a) dépend des termes présents dans la requête et dans le document ;
  (b) les documents sont ensuite triés par ordre de scores décroissants.

Fonctions d'appariement (requête, document) 1/4

Fonction d'appariement:

- fonction f : $f(q, d) \in \mathbb{R}$ (q : requête, d : document) ;
- le score représente la pertinence prédite d'un document (*relativement aux autres documents*) par rapport au besoin d'information exprimé par la requête.

Quantités observées, prises en compte dans le calcul du score :

- fréquence des termes de la requête dans le document ;
- pouvoir de discrimination d'un terme ;
- longueur du document.

Note : d'autres quantités peuvent être ajoutés en prenant en compte les métadonnées, la structure de la collection, ...

Fonctions d'appariement (requête, document) 2/4

- Fréquence d'un terme t de la requête dans le document d
  noté tf pour *term frequency* :

$$tf(t,d) = \text{nb de fois que } t \text{ apparaît dans } d.$$

- Pouvoir de discrimination d'un terme t
  noté idf pour *inverse document frequency* :

$$idf(t) = \log\left(\frac{N}{df(t)}\right) \quad \text{avec :}$$

  - N : nombre de documents dans la collection ;
  - $df(t)$: nombre de documents contenant le terme t.

- Longueur d'un document d (notée $L(d)$) : $L(d) = \sum_w tf(w,d)$.

Note : implicitement, on perd la notion "document = séquence de termes" pour arriver à une représentation *sac-de-mots*.

Fonctions d'appariement (requête, document) 3/4

Les fonctions d'appariements sont des *heuristiques* réalisant un *compromis* entre les critères suivants :

[un document d est *supposé* plus pertinent que d' lorsque :]

- d contient plus d'occurrences des termes de la requête que d' ;
- d contient plus de termes de la requête différents que d' ;
- d contient des termes de la requête plus *discriminants* que d' ;
- d contient les mêmes termes de la requête que d', et d est plus court que d'.

"A Formal Study of Information Retrieval Heuristics", Fang et al. SIGIR 2004

Note : ces desiderata expriment des scores de pertinence *relatifs*
$\Rightarrow$ l'objectif n'est pas d'estimer la pertinence d'un document *dans l'absolu*.

Fonctions d'appariement (requête, document) 4/4

Plus précisément :

[on souhaite $f(q,d) > f(q,d')$ lorsque :]

- d contient plus d'occurrences d'un terme t de la requête que d'

  sous les contraintes : (a) $L(d) = L(d')$

  et : (b) $\forall t'$ tel que $t' \in q$ et $t' \neq t$: $tf(t,d) = tf(t,d')$;

- d contient plus de termes différents de la requête que d'

  soient deux termes de la requête t et t' et deux documents d et d' tels que :

  (a) $\sum_{t'' \in q, t'' \notin \{t,t'\}} tf(t'',d) = \sum_{t'' \in q, t'' \notin \{t,t'\}} tf(t'',d')$;

  (b) $tf(t,d) + tf(t',d) = tf(t,d')$ et $tf(t',d') = 0$;

  (c) $idf(t) = idf(t')$;

- d contient des termes de la requête plus *discriminants* que d'

  i.e. on souhaite $f(q,d) > f(q,d')$ dans le cas où :

  (a) $t, t' \in q$ et $tf(t,d) + tf(t',d) = tf(t,d') + tf(t',d')$;

  (b) $idf(t) > idf(t')$;

- d contient les mêmes termes de la requête que d', et d est plus court que d'.

Modèles de recherche sur les documents plats

- Les règles précédentes donnent une *spécification partielle* d'une fonction d'appariement ;
- les règles définissant le compromis à réaliser sont difficiles à respecter sur toutes les valeurs possibles de $tf(t,d)$, $idf(t)$ et $L(d)$;
- les heuristiques de recherche utilisent différents cadres formels pour guider la définition complète d'une fonction d'appariement :
  - modèle vectoriel ;
  - modèle de langue ;
  - principe probabiliste.

Modèles de recherche (préambule) : scoring $tf.idf$

- Fonction d'appariement : somme des $tf.idf$

$$f(q,d) = \sum_{t \in q} tf(t,d) idf(t)$$

- un des premiers modèles de recherches ;

- propriétés :
  - favorise les documents qui ont plus de termes de la requête ;
  - favorise les documents qui ont des termes plus discriminants.

- Inconvénients :
  - pas de prise en compte de la taille des documents ;
  - ne favorise pas l'apparence de termes différents.

Modèles de recherche : modèle vectoriel 1/3

- Extension naturelle du $tf.idf$;
- principe : la requête *et* le document sont représentés comme un vecteur dans un espace *sac-de-mots* :
  - notations :
    $\overrightarrow{v}(q)$: vecteur représentatif de la requête q ;
    $\overrightarrow{V}(d)$: vecteur représentatif du document d;
  - fonctions d'appariement :

$$f(q,d) = sim\left(\overrightarrow{v}(q), \overrightarrow{V}(d)\right)$$

où sim est une fonction de similarité entre vecteurs.
Exemples : $sim(\overrightarrow{v},\overrightarrow{v}') = \overrightarrow{v}.\overrightarrow{v}'$ ou $sim(\overrightarrow{v},\overrightarrow{v}') = cos(\overrightarrow{v},\overrightarrow{v}') = \frac{\overrightarrow{v}.\overrightarrow{v}'}{||\overrightarrow{v}||\,||\overrightarrow{v}'||}$.

Modèles de recherche : modèle vectoriel 2/3

- Fonctions d'appariement :

$$f(q,d) = sim\left(\overrightarrow{v}(q), \overrightarrow{V}(d)\right)$$

où sim est une fonction de similarité entre vecteurs
par exemple $sim(v,v') = v.v'$ ou $sim(v,v') = cos(v,v') = \frac{v.v'}{||v||\,||v'||}$

- Exemples :
  1. $sim =$ produit scalaire, et :
     $\overrightarrow{V}_w(d) = tf(w,d)$ et $\overrightarrow{v}_w(q) = \begin{cases} idf(w) & \text{si } t \in q \\ 0 & \text{sinon} \end{cases}$
     on retrouve la somme des tf.idf ;
  2. $sim =$ cosinus et même représentation pour $\overrightarrow{v}(q)$ et $\overrightarrow{V}(d)$
     $\approx$ prise en compte de la longueur du document (attention à la norme 2) ;
  3. $sim =$ produit scalaire, et :
     $\overrightarrow{V}_w(d) = \begin{cases} 1+\log(tf(w,d)) & \text{si } tf(w,d)>0 \\ 0 & \text{sinon} \end{cases}$ et $\overrightarrow{v}_w(q) = \begin{cases} idf(w) & \text{si } t \in q \\ 0 & \text{sinon} \end{cases}$
     favorise les termes différents dans un même document ;

Modèles de recherche : modèle vectoriel 2/3

- Fonctions d'appariement :

$$f(q,d) = sim\left(\overrightarrow{v}(q), \overrightarrow{V}(d)\right)$$

où sim est une fonction de similarité entre vecteurs
par exemple $sim(v,v') = v.v'$ ou $sim(v,v') = cos(v,v') = \frac{v.v'}{||v||\,||v'||}$

- Exemples :
  1. $sim =$ produit scalaire, et :
     $\overrightarrow{V}_w(d) = tf(w,d)$ et $\overrightarrow{v}_w(q) = \begin{cases} idf(w) & \text{si } t \in q \\ 0 & \text{sinon} \end{cases}$
     on retrouve la somme des tf.idf ;
  2. $sim =$ cosinus et même représentation pour $\overrightarrow{v}(q)$ et $\overrightarrow{V}(d)$
     $\approx$ prise en compte de la longueur du document (attention à la norme 2) ;
  3. $sim =$ produit scalaire, et :
     $\overrightarrow{V}_w(d) = \begin{cases} 1+\log(tf(w,d)) & \text{si } tf(w,d)>0 \\ 0 & \text{sinon} \end{cases}$ et $\overrightarrow{v}_w(q) = \begin{cases} idf(w) & \text{si } t \in q \\ 0 & \text{sinon} \end{cases}$
     favorise les termes différents dans un même document ;

Modèles de recherche : modèle vectoriel 2/3

- Fonctions d'appariement :

$$f(q,d) = sim\left(\overrightarrow{v}(q), \overrightarrow{V}(d)\right)$$

où sim est une fonction de similarité entre vecteurs
par exemple $sim(v,v') = v.v'$ ou $sim(v,v') = cos(v,v') = \frac{v.v'}{||v||\,||v'||}$

- Exemples :
  1. $sim =$ produit scalaire, et :
     $\overrightarrow{V}_w(d) = tf(w,d)$ et $\overrightarrow{v}_w(q) = \begin{cases} idf(w) & \text{si } t \in q \\ 0 & \text{sinon} \end{cases}$
     on retrouve la somme des tf.idf ;
  2. $sim =$ cosinus et même représentation pour $\overrightarrow{v}(q)$ et $\overrightarrow{V}(d)$
     $\approx$ prise en compte de la longueur du document (attention à la norme 2) ;
  3. $sim =$ produit scalaire, et :
     $\overrightarrow{V}_w(d) = \begin{cases} 1+\log\left(tf(w,d)\right) & \text{si } tf(w,d)>0 \\ 0 & \text{sinon} \end{cases}$ et $\overrightarrow{v}_w(q) = \begin{cases} idf(w) & \text{si } t \in q \\ 0 & \text{sinon} \end{cases}$
     favorise les termes différents dans un même document ;

Modèles de recherche : modèle vectoriel 3/3

Modèle vectoriel : conclusion

- un premier cadre formel pour définir des fonctions d'appariement
  fondements : similarités entre vecteurs dans un modèle sac-de-mots ;
- permet d'obtenir plusieurs fonctions différentes
  nécessité de faire des expériences pour déterminer les bons paramètres ;
- mauvaise prise en compte de la longueur des documents
  la norme 2 des vecteurs ne correspond pas à la notion intuitive de "longueur d'un document".

Modèles de recherche : Modèles de langue 1/4

- Modèle de langue = distribution de probabilité sur les mots ;
- Utilisation en RI, traduction automatique, etc. ;
- Modèles de langue *unigram* :
  la probabilité (connaissant la longueur) d'une séquence de mots est le produit des probabilités des mots ;
- Exemple : considérons modèle de langue suivant :

mot	le	la	et	chat	souris	mange	court	après
probabilité	0.1	0.1	0.1	0.2	0.2	0.1	0.1	0.1

on a alors : $P(\text{"le chat court après la souris et la mange"})$

$$= 0.1 * 0.2 * 0.1 * 0.1 * 0.1 * 0.2 * 0.1 * 0.1 * 0.1$$

Modèles de recherche : Modèles de langue 2/4

Modèles de langue pour la RI:

- Principe :
  1. hypothèse générative : chaque document est généré par un unique modèle de langue (on a donc un modèle génératif par document) ;
  2. le score d'un document est établi selon le *query likelihood model* :

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

où $p_{doc=d}(t)$: probabilité de générer t selon le modèle associé à d.

- Définition de $p_{doc=d}(.)$: maximum de vraisemblance

$p_{doc=d}(.)$ est le modèle rendant le plus vraisemblable possible le document d

$$p_{doc=d} = \underset{\substack{p:\text{proba} \\ \text{sur les termes}}}{\arg\max} \log P(d) = \underset{\substack{p:\text{proba} \\ \text{sur les termes}}}{\arg\max} \sum_t tf(t,d) \log p(t)$$

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

Modèles de recherche : Modèles de langue 3/4

Modèles de langue pour la RI:

$$f(q,d) = \sum_{t \in q} \log \frac{tf(t,d)}{L(d)} \quad \text{(en supposant } tf(t,q) \in \{0,1\}\text{)}.$$

Propriétés :

- favorise les documents ayant plus de termes de la requête ;
- favorise les documents ayant des termes différents de la requête ;
- pénalise les documents longs ;
- *pas de prise en compte du pouvoir de discrimination des termes*
  correction par *lissage de Laplace* :

$$p_{doc=d}(t) = \frac{tf(t,d) + \mu\, p_{collection}(t)}{L(d) + \mu} \text{ avec } p_{collection}(t) = \frac{\sum_{d'} tf(t,d')}{\sum_{t'} \sum_{d'} tf(t',d')}.$$

Modèles de recherche : Modèles de langue 4/4

- Cadre formel adapté à la RI ;
- paramètres estimés par des outils statistiques ;
- modélisation très flexible
  modèles bi-grams, mélange de modèles de langues, ...
- après lissage :
  possède les propriétés essentielles des fonctions d'appariement.

Modèles de recherche : BM25 1/3

- "Okapi BM25" : développé au cours des années 90
  décrit dans *"A Probabilistic Model of Information Retrieval: Development and Comparative Experiments", Spärck Jones et al., IPM (2000) ;*
- heuristique de recherche la plus performante ;
- inspirée par le *probability ranking principle* de S. Robertson :

  idéalement, les documents doivent être ordonnés par :

$$P\big(\text{"document pertinent pour } q\text{"} \big| \text{document } = d\big)$$

  ou, de façon équivalente :

$$f(q,d) = \log \frac{P\big(\text{document } = d \big| \text{"document pertinent pour } q\text{"}\big)}{P\big(\text{document } = d \big| \text{"document non pertinent pour } q\text{"}\big)} .$$

Modèles de recherche : BM25 2/3

$$f(q,d) = \log \frac{P\left(\text{document } = d \middle| \text{"document pertinent pour } q\text{"}\right)}{P\left(\text{document } = d \middle| \text{"document non pertinent pour } q\text{"}\right)} .$$

- + hypothèse d'indépendance entre les mots sachant la pertinence
  similaire à Naive Bayes :

$$f(q,d) = \sum_w \log \frac{P\left(TF_w = tf(w,d) \middle| \text{"document pertinent pour } q\text{"}\right)}{P\left(TF_w = tf(w,d) \middle| \text{"document non pertinent pour } q\text{"}\right)} .$$

  TF_w : variable aléatoire, représente le nb. de fois que w est dans un document ;
- + loi de poisson pour $P\left(TF_w = . \middle| \text{"document pertinent pour } q\text{"}\right)$
  non-linéarité par rapport à $tf(w,d)$ (et même saturation);
- + simplifications + prise en compte de la longueur du document...

Modèles de recherche : BM25 3/3

Formule BM25 :

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

avec :

- $idf'(t) = \max\left(0, \log \frac{N-df(t)+0.5}{df(t)+0.5}\right)$ “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.

Remarque :
BM25 possède (à peu près) toutes les caractéristiques souhaitées.

Heuristiques de recherche : conclusion

- Les heuristiques de recherche réalisent un compromis complexe
  nécessité de prendre en compte
  - la fréquence d'apparition des termes dans le document ;
  - le pouvoir de discrimination des termes ;
  - la longueur des documents.

- Les heuristiques ont des paramètres qui dépendent du corpus
  $\Rightarrow$ nécessité de créer des *corpus d'évaluation* et des *mesures d'évaluation* pour déterminer des paramètres optimaux.

- Peu de recherche aujourd'hui sur ces heuristiques
  BM25 est considéré comme réalisant un excellent compromis ;
  beaucoup de recherche pour la prise en compte de critères additionnels.

- L'apprentissage est très utilisé dans les fonctions d'appariement
  estimation des paramètres optimaux pour des mesures d'évaluation spécifiques,
  combinaisons d'heuristiques prenant en compte des critères hétérogènes.