Modèles de Recherche d'Information - EARIA 2012

Eric Gaussier

AMA/LIG
Université J. Fourier/Grenoble 1
UFR-IM^{2}AG
Informatique, Mathématiques et Mathématiques Appliquées de Grenoble
Eric.Gaussier@imag.fr

Octobre 2012

Table des matières

Objectifs du cours

- L'objectif de cours est d'introduire les principaux modéles (et algorithmes associés) utilisés en Recherche d'Information (RI).
- Nous nous intéresserons en particulier à :
  - L'indexation et la représentation des documents
  - Les modéles standard de la RI *ad hoc* :
    - Le modèle booléen
    - Le modèle vectoriel
    - Les modèles probabilistes

Domaines d'applications

- La Recherche d'Information
  - Module d'indexation des requêtes
  - Module d'indexation des documents
  - Appariement entre requêtes et documents (mesure de similarité/distance entre requêtes et documents
- La classification (clustering)
  - Similarité/distance entre documents joue un rôle central

Table des matières

Etapes de l'indexation textuelle

1. Segmentation
   - Découper un texte en mots :

     l'information donne sens à l'esprit
     l', information, donne, sens, à, l', esprit

     soit 7 mots mais seulement 6 types ; nécessité d'un dictionnaire (au moins léger) pour certains mots ou expressions
2. Filtrage par un anti-dictionnaire des mots vides
3. Normalisation
   - De la casse, des formes fléchies, des familles lexicales
   - Lemmatisation, racinisation

$\rightarrow$ Sac-de-mots : *inform, don, sens, esprit*

Représentation vectorielle des docs (1)

- L'ensemble des types forme le vocabulaire d'une collection. Soit M la taille de ce voc., et soit N le nombre de doc. dans la coll. On considére l'espace vectoriel à M dimensions ($\mathbb{R}^M$), dans lequel chaque axe correspond à un type
- Chaque document est alors représenté par un vecteur de $\mathbb{R}^M$ dont les coordonnées sont les poids des mots dans le document :
  - **Présence/absence** : $t_w^d = 1$ si w apparaît dans d, 0 sinon
  - Nbre d'occurrences : $t_w^d = \#\text{occ}(w, d)$
  - Nombre d'occurrences normalisé : $t_w^d = \frac{\#\mathbf{OCC}(w,d)}{\sum_{w'} \#\mathbf{OCC}(w',d)}$
  - Le *tf*idf* (voir plus loin)

Représentation vectorielle des docs (2)

La représentation vectorielle adoptée permet d'avoir directement accés aux outils mathématiques associés : distances, similarités, réduction de dimensions, ...

Exercices

- Chaque document est représenté par un tableau à M dimensions contenant les poids (coordonnées) des termes (types, mots) ; écrire un algorithme qui calcule le produit scalaire entre 2 documents ($\text{scal}(d, d') = \sum_w t_w^d t_w^{d'}$)
- Quelle est la complexité d'un algorithme qui calcule le produit scalaire entre un document et tous les autres documents de la collection ?

Une représentation creuse !

La majorité des termes de la collection n'apparaissent pas dans un document donné ; chaque document a donc la majeure partie de ses coordonnées nulles ; un gain d'espace peut être obtenu en ne représentant pas ces coordonnées nulles

Exemple de représentation creuse :

document d {

int l	(long. du doc. (types))
TabTermes int[l]	(indices des termes)
TabPoids float[l]	(poids des termes)
...	

Illustration

	programmation	langage	C	java	$\cdots$
d_1	0	1	0	1	$\cdots$
d_2	1	1	0	0	$\cdots$

Le fichier inverse

Possibilité d'accélérer le calcul dans le cas de représentations creuses, en utilisant un *fichier inverse* qui fournit, pour chaque terme, l'ensemble des documents dans lesquels il apparaît :

$$\text{terme } i \begin{cases} \text{int l} & \text{(nbre de docs assoc.)} \\ \text{TabDocs int[l]} & \text{(indices des docs)} \\ \ldots \end{cases}$$

On procède alors en 2 étapes :

1. Construction de l'ensemble des documents qui contiennent au moins un terme de la requête
2. Calcul de la similarité/distance entre requête et les documents de cet ensemble

Remarque Avantageux (gain de 3 à 5 ordres de grandeur) avec toute mesure (distance, similarité) qui ne fait pas intervenir les termes non présents dans un document. Produit scalaire ?, cosinus ?, distance euclidienne ?

Illustration

Collection

	programmation	langage	C	java	...
d_1	3 (1)	2 (1)	4 (1)	0 (0)	...
d_2	5 (1)	1 (1)	0 (0)	0 (0)	...
d_0	0 (0)	0 (0)	0 (0)	3 (1)	...

Fichier inverse

	d_1	d_2	d_3	...
programmation	1	1	0	...
langage	1	1	0	...
C	1	0	0	...
...	...	...	...	

Construction du fichier inverse

Dans le cadre d'une collection statique, 3 étapes principales régissent la construction du fichier inverse :

1. Extraction des paires d'identifiants *(terme, doc)*, passe compléte sur la collection
2. Tri des paires suivant les id. de terme, puis les id. de docs
3. Regroupement des paires pour établir, pour chaque terme, la liste des docs

Ces étapes ne posent aucun probléme dans le cas de petites collections où tout se fait en mémoire

Quid des grandes collections ?

Cas de mémoire insuffisante

Il faut dans ce cas stocker des informations temporaires sur disque
Trois étapes :

1. Collecte des paires TermId-DocId et écriture dans un fichier
2. Lecture par blocs de ce fichier, inversion de chaque bloc et écriture dans une série de fichier
3. Fusion des différents fichier pour créer le fichier inversé

Algorithme associé : *Blocked sort-based indexing* (BSBI)

L'algorithme BSBI (1)

1. $n \leftarrow 0$
2. while (tous les docs n'ont pas été traités)
3. do
4. $\quad n \leftarrow n + 1$
5. $\quad$ block $\leftarrow$ ParseBlock()
6. $\quad$ BSBI-Invert(block)
7. $\quad$ WriteBlockToDisk(block,f_n)
8. MergeBlocks(f_1, ..., f_n ;f_{merged})

L'algorithme BSBI (2)

L'inversion, dans BSBI, consiste en un tri sur les identifiants de termes (premiére clé) et les identifiants de documents (deuxiéme clé). Le résultat est donc un fichier inverse pour le bloc lu. Complexité en $O(T \log T)$ où T est le nombre de paires terme-document (mais étapes de lecture et de fusion peuvent être plus longues)

Exemple

w_1 = "brutus", w_2 = "caesar", w_3 = "julius", w_4 = "kill", w_5 = "noble"

$w_1 : d_1$	$w_2 : d_4$	$w_2 : d_1$
$w_3 : d_{10}$	$w_1 : d_3$	$w_4 : d_8$
$w_5 : d_5$	$w_2 : d_2$	$w_1 : d_7$

Table des matières

Les différents modéles standard

- Modéle booléen
- Modéle vectoriel
- Modéles probabilistes

Notations

x_w^q	Nbre occurrences du mot w dans q
x_w^d	Nbre occurrences du mot w dans le document d
t_w^d	Version normalisée de x_w^d (poids)
N	Nbre de documents dans la collection
M	Nbre de termes dans la collection
F_w	Nbre d'occ. total de w : $F_w = \sum_d x_w^d$
N_w	Fréquence documentaire de w : $N_w = \sum_d I(x_w^d > 0)$
l_d	Longueur du document d
m	Longueur moyenne dans la collection
L	Longueur de la collection
RSV	Retrieval Status Value (score)

Le modèle booléen (1)

Modèle simple fondé sur la théorie des ensembles et l'algèbre de Boole, caractérisé par :

- Des poids binaires (présence/absence)
- Des requêtes qui sont des expressions booléennes
- Une pertinence binaire
- Pertinence système : satisfaction de la requête booléenne

Le modèle booléen (2)

Exemple

$q =$ programmation $\wedge$ langage $\wedge$ (C $\vee$ java)

($q =$ [prog. $\wedge$ lang. $\wedge$ C] $\vee$ [prog. $\wedge$ lang. $\wedge$ java])

	programmation	langage	C	java	$\cdots$
d_1	3 (1)	2 (1)	4 (1)	0 (0)	$\cdots$
d_2	5 (1)	1 (1)	0 (0)	0 (0)	$\cdots$
d_0	0 (0)	0 (0)	0 (0)	3 (1)	$\cdots$

Score de pertinence

$\text{RSV}(d_j, q) = 1$ si $\exists\, q_{cc} \in q_{dnf}$ tq $\forall w, t_w^d = t_w^q$; 0 sinon

Le modèle booléen (3)

Considérations algorithmiques

Quand la matrice documents-termes est creuse (lignes et colonnes), utiliser un fichier inverse pour sélectionner le sous-ensemble des documents qui ont un score de pertinence non nul avec la requête (sélection rapidement réalisée). Le score de pertinence n'est alors calculé que sur les documents de ce sous-ensemble (généralisation à d'autres types de score).

	d_1	d_2	d_3	$\cdots$
programmation	1	1	0	$\cdots$
langage	1	1	0	$\cdots$
C	1	0	0	$\cdots$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	

Le modèle booléen (4)

Avantages et désavantages

+ Facile à développer

- Pertinence binaire ne permet pas de tenir compte des recouvrements thématiques partiels

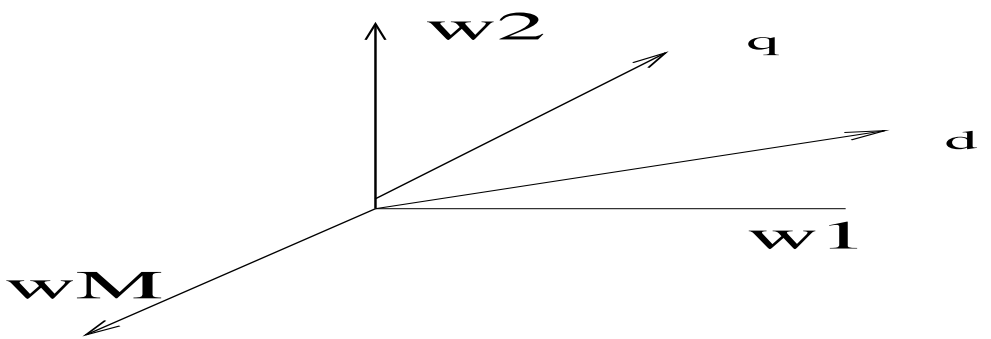
- Passage d'une besoin d'information à une expression booléenne

Remarque À la base de beaucoup de systèmes commerciaux

Le modèle vectoriel (1)

Revient sur deux défauts majeurs du modèle booléen : des poids et une pertinence binaires
Il est caractérisé par :

- Des poids positifs pour chaque terme dans chaque document
- Mais aussi des poids positifs pour les termes de la requête
- Une représentation vectorielle des documents et des requêtes



Espace vectoriel des terme

Le modèle vectoriel (2)

On considère donc que les documents et les requêtes sont des vecteurs dans un espace vectoriel de dimension M dont les axes correspondent aux termes de la collection

Similarité Cosinus de l'angle entre les deux vecteurs

$$\text{RSV}(q,d) = \frac{\sum_w t_w^d t_w^q}{\sqrt{\sum_w (t_w^d)^2}\sqrt{\sum_w (t_w^q)^2}} \quad (1)$$

Propriété Le cosinus est maximal lorsque document et requête contiennent exactement les mêmes termes, dans les mêmes proportions ; minimal lorsqu'ils n'ont aucun terme en commun (*degré de similarité*)

Le modèle vectoriel (3)

Calcul des poids : schéma *tf-idf*

Qu'est-ce qui décrit bien un document ?

$\rightarrow$ ses termes fréquents ($\text{tf}_w^d = \frac{x_w^d}{\max_{w'} x_{w'}^d}$)

Qu'est-ce qui distingue un document des autres ?

$\rightarrow$ ses termes spécifiques ($\text{idf}_w = \log \frac{N}{N_w}$)

$$\begin{cases} t_w = \text{tf}_w^d \times \text{idf}_w \\ \text{RSV}(q,d) = \frac{\sum_w t_w^d t_w^q}{\sqrt{\sum_w (t_w^d)^2}\sqrt{\sum_w (t_w^q)^2}} \end{cases}$$

Le modèle vectoriel (4)

Avantages et désavantages

+ Schémas de pondération permettant de prendre en compte différentes propriétés des index

+ Un appariement partiel qui permet de retrouver les documents qui répondent en partie à la requête

+ Un ordre total sur les documents qui permet de distinguer les documents qui abordent pleinement les thèmes de la requête de ceux qui ne les abordent que marginalement

- Difficulté d'aller plus avant dans le cadre vectoriel (modèle relativement simple)

Complexité : comme le modèle booléen, linéaire sur le nombre de documents qui contiennent les termes de la requête (similarité requête-document plus coûteuse)

Les différents modèles probabilistes

- *Binary Independence Model* et BM25 (S. Robertson & S. Walker)
- *Inference Network Model* (Inquery) - *Belief Network Model*
- *Statistical Language Models*
  - *Query likelihood* (Ponte & Croft)
  - *Probabilistic distance retrieval model* (Zhai & Lafferty)
- *Information models* (Amati & Van Rijsbergen, Clinchant & Gaussier)

Généralités

Modèle booléen $\rightarrow$ pertinence binaire
Modèle vectoriel $\rightarrow$ degré de similarité
Modèle BIM $\rightarrow$ degré de pertinence
probabilité d'un document d'être pertinent

- R variable aléatoire binaire qui indique la pertinence : $R = r$ (*relevant*) ou $\bar{r}$ (*not relevant*)
- $P(R = r|d, q)$: probabilité que R prenne la valeur r pour le document d et la requête q considérés
- $\text{RSV}(q, d) = \log \frac{P(R=r|d,q)}{P(R=\bar{r}|d,q)}$

Deux points de vue peuvent être adoptés ici pour ré-écrire ces quantités : le point de vue *génération du document* (BIM) ou le point de vue *génération de la requête* (LM)

Le modèle BIM (1)

Hypothèses

- La probabilité de pertinence dépend seulement des représentations de la requête et du document
- Les poids des termes dans les documents sont binaires : $d = (1010 \cdots 010 \cdots)$ ($t_w^d = 0$ ou 1)
- Chaque terme est caractérisé par une variable binaire A_w : $P(A_w = 1|q, r)$ probabilité que w apparaisse dans un document pertinent ($P(A_w = 0|q, r) = 1 - P(A_w = 1|q, r)$)
- Conditionnellement à R, les termes d'indexation sont indépendants
- $P(R = r|d, q) = \frac{P(R=r,q)P(d|R=r,q)}{P(d,q)}$

Le modèle BIM (2)

Avec ces hypothèses :

$$\begin{aligned}
\text{RSV}(q,d) &= \log \frac{P(R=r|d,q)}{P(R=\bar{r}|d,q)} \\
&= \log \frac{P(d|R=r,q)}{P(d|R=\bar{r},q)} + \log \frac{P(R=r,q)}{P(R=\bar{r},q)} \\
&=_{\text{rg}} \log \frac{P(d|R=r,q)}{P(d|R=\bar{r},q)}
\end{aligned}$$

Le modèle BIM (3)

En utilisant :

$$P(d|R=r,q) = P(d=(1010\cdots010\cdots)|q,R=r)$$

$$= \prod_w P(A_w=1|q,R=r)^{t_w^d t_w^q} P(A_w=0|q,R=r)^{(1-t_w^d)t_w^q}$$

Nous obtenons :

$$\text{RSV}(q,d) = \text{rang} \sum_w t_w^q t_w^d \left(\log\left(\frac{P(A_w=1|R=r,q)}{1-P(Aw=1|R=r,q)}\right) + \log\left(\frac{1-P(A_w=1|R=\bar{r},q)}{P(Aw=1|R=\bar{r},q)}\right)\right)$$

Le modèle BIM (4)

Comment calculer $P(A_w = 1|R = r, q)$ et $P(A_w = 1|R = \bar{r}, q)$?
Procédé itératif :

1. On définit des valeurs initiales (par exemple $P(A_w|R = r, q) = 0.5$, $P(A_w|R = \bar{r}, q) = \frac{N_w}{N}$)
2. On effectue une recherche (valeur courante paramètres)
3. On met à jour les paramètres en fonction des résultats retrouvés (éventuellement avec utilisateur). Par ex., V cardinal de l'ensemble des docs jugés pertinents :

$$P(A_w = 1|R = r, q) = \frac{V_w}{V}, \quad P(A_w = 1|R = \bar{r}, q) = \frac{N_w - V_w}{N - V}$$

Le modèle BIM (5)

Avantages et désavantages

+ Une notion claire et théoriquement fondée du degré de pertinence

+ Le processus de recherche d'information est un processus itératif qui implique l'utilisateur

- Sensibilité aux valeurs initiales

- Procédure d'estimation certes intéressante mais coutêuse

Complexité similaire au modèle vectoriel à chaque itération ; un peu plus complexe en général

Le modèle BM25

Permet de corriger les défauts du modèle BIM

$$\text{RSV}(q,d) = \log\left(\frac{N - N_w + 0.5}{N_w + 0.5}\right) \times \frac{(k_1+1)x_w^d}{k_1((1-b)+b\frac{l_d}{m}) + x_w^d} \times \frac{(k_3+1)x_w^q}{k_3 + x_w^q}$$

$k_1 \in [1;2], b = 0.75, k_3 \in [0;1000]$

Introduction à ML/QL : deux dés

Soient D_1 et D_2 deux dés tels que :

Pour D_1, $P(1) = P(3) = P(5) = \frac{1}{3} - \epsilon$, $P(2) = P(4) = P(6) = \epsilon$
Pour D_2, $P(1) = P(3) = P(5) = \epsilon$; $P(2) = P(4) = P(6) = \frac{1}{3} - \epsilon$

Supposons que l'on observe la séquence $Q = (1, 3, 3, 2)$. Quel dé est le plus susceptible d'avoir généré cette séquence ?

Réponse

$P(Q|D_1) = (\frac{1}{3} - \epsilon)^3\epsilon$; $P(Q|D_2) = (\frac{1}{3} - \epsilon)\epsilon^3$

Introduction à ML/QL : deux dés

Soient D_1 et D_2 deux dés tels que :

Pour D_1, $P(1) = P(3) = P(5) = \frac{1}{3} - \epsilon$, $P(2) = P(4) = P(6) = \epsilon$
Pour D_2, $P(1) = P(3) = P(5) = \epsilon$; $P(2) = P(4) = P(6) = \frac{1}{3} - \epsilon$

Supposons que l'on observe la séquence $Q = (1, 3, 3, 2)$. Quel dé est le plus susceptible d'avoir généré cette séquence ?

Réponse

$P(Q|D_1) = (\frac{1}{3} - \epsilon)^3\epsilon$; $P(Q|D_2) = (\frac{1}{3} - \epsilon)\epsilon^3$

Modèle ML/QL (1)

Les documents jouent le rôle des dés, la requête de la séquence.

On cherche alors quels sont les documents les plus susceptibles d'avoir "généré" la requête.

Chaque document d est associé à un modèle de document $\mathcal{M}_d$ et le score de pertinence est donné par la probabilité que q soit générée par ce modèle de document :

$$\text{RSV}(q,d) = P(q|\mathcal{M}_d) \; (\equiv P(q|R=r,d))$$

Modèle ML/QL (2)

Différentes lois de probabilité peuvent être considérées :

- **Multinomiale** : $P(q|\mathcal{M}_d) = \prod_w p(w|\mathcal{M}_d)^{x_w^q}$
  (Song & Croft)

- Processus de Bernoulli (t_w^q binaires) :
  $P(q|\mathcal{M}_d) = \prod_w P(t_w^q = 1|\mathcal{M}_d)^{t_w^q}(1 - P(t_w^q = 1|\mathcal{M}_d))^{1-t_w^q}$
  (Ponte & Croft)

- Poisson : $P(q|\mathcal{M}_d) = \prod_w \frac{(l_q\lambda_w)^{x_w^q} e^{l_q\lambda_w}}{x_w^q!}$
  (Mei *et al.*)

Modèle ML/QL (3)

Comment estimer les paramètres (du modèle multinomial) ?

Une approche simple : le maximum de vraisemblance - on cherche les valeurs de $p(w|\mathcal{M}_d)$ qui maximisent la probabilité d'observer le document d

$$\begin{aligned} \hat{p}(w|\mathcal{M}_d) &= \frac{x_w^d}{\sum_w x_w^d} \\ (\text{RSV}(q,d) &= \prod_w \hat{p}(w|\mathcal{M}_d)^{x_w^q}) \end{aligned} \quad (2)$$

q : (programmation, java) et d :(langage,java) $\rightarrow$ Problème !

Solution : lissage

Modèle ML/QL (4)

Pour lisser, on tient compte du modèle de la collection (fournit l'information qui manque au niveau du document)

$$p(w|\mathcal{M}_d) = (1 - \alpha_d)\hat{p}(w|\mathcal{M}_d) + \alpha_d \hat{p}(w|\mathcal{M}_C)$$

Avec : $\hat{p}(w|\mathcal{M}_d) = \frac{x_w^d}{\sum_w x_w^d}$, $\hat{p}(w|\mathcal{M}_C) = \frac{F_w}{\sum_w F_w}$

Modèle ML/QL (5)

Lissage de Jelinek-Mercer : $\alpha_{\boldsymbol{d}} = \lambda$

- $\mathcal{D}$ ensemble de développement (requêtes et jugements de pertinence)
- $\lambda = 0$, recherche sur $\mathcal{D}$ et évaluation des résultats
- Augmenter la valeur de λ de ϵ (e.g. 0.001), nouvelle recherche sur $\mathcal{D}$ et évaluation des résultats
- Répéter jusqu'à $\lambda = 1$
- Sélectionner la meilleure valeur de λ

Modèle ML/QL (6)

Lissage de Dirichlet : $\alpha_d = \frac{\mu}{\mu + l_d}$

μ est un paramètre appris comme précédemment (mais gamme de variation plus large - $\mathbb{R}^+$)

(Zhai & Lafferty)

Modèle ML/QL (7)

Avantages et désavantages

+ Un cadre théorique clair et bien fondé, facile à implanter et conduisant à de bons résultats

+ Facile à étendre à d'autes cadres (recherche d'information multilingue, boucle de pertinence, ...)

- Estimation des λs nécessite des données d'apprentissage

- Difficulté (conceptuelle) d'une boucle de rétro-pertinence (*(pseudo-)relevance feedback*)

Complexité similaire au modèle vectoriel

Une généralisation deML/ QL :
Kullback-Leibler divergence retrieval model

$$\text{RSV}(q,d) = -KL(\Theta_q||\mathcal{M}_d) = -\sum_w P(w|\Theta_q) \log \frac{P(w|\Theta_q)}{P(w|\mathcal{M}_d)}$$

En développant :

$$\text{RSV}(q,d) =_{\text{rang}} \sum_w x_w^q \log P(w|\mathcal{M}_d) =_{\text{rang}} logP(q|\mathcal{M}_d)$$

C'est en général le modèle KL qui est implanté dans les systèmes de recherche d'information (Lemur, Terrier)

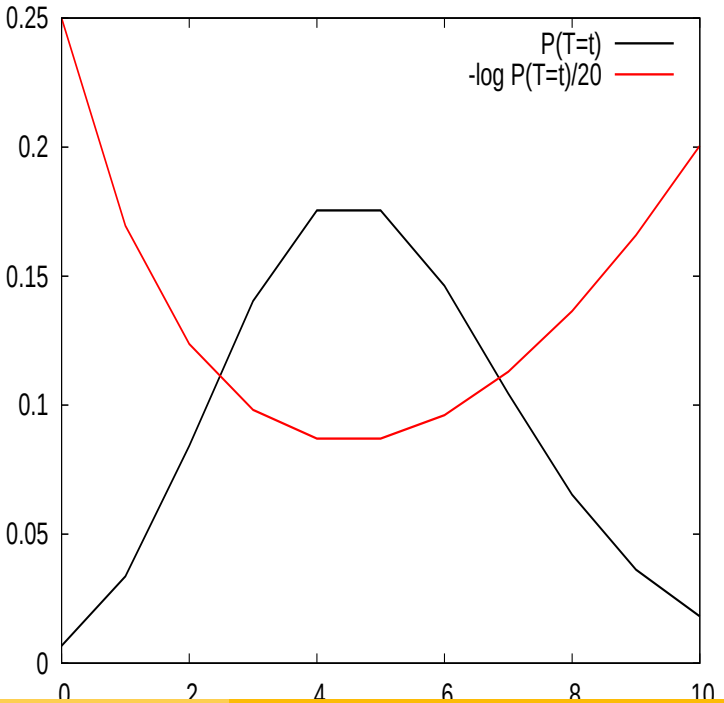
Modèles d'information (1)

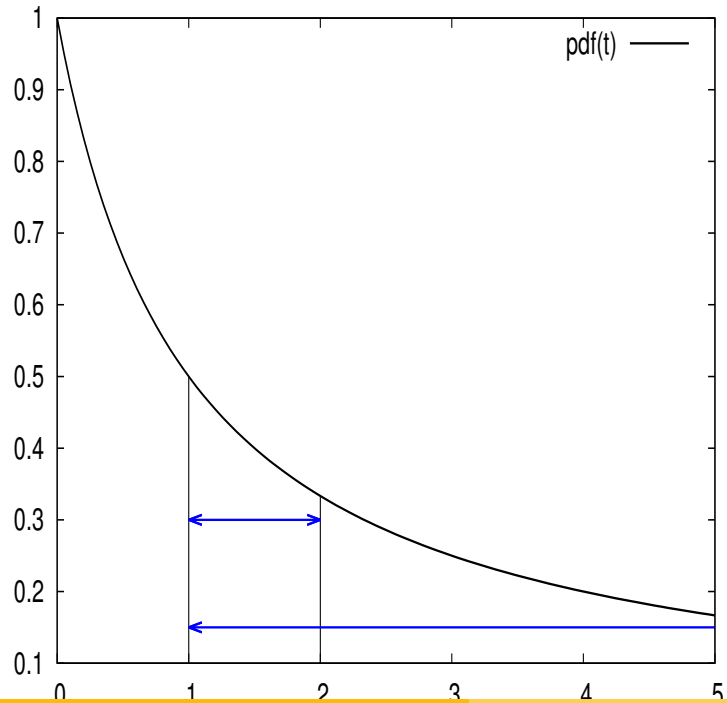
Approches informationnelles

Utilisation de l'information se Shannon (sur la fonction de survie) pour mesurer l'importance d'un mot dans un document

$$\text{Shannon Information} = -\log P(T_w = t|\lambda_w)$$

$$\text{Information} = -\log P(T_w > t|\lambda_w)$$





Modèles d'information (2)

Requêtes et documents sont comparés sur la base de l'information moyenne apportée par le document sur la requête :

$$\text{RSV}(q,d) = -\sum_{w \in q} t_w^q \log P(T_w > t_w^d | \lambda_w) \quad (3)$$

1. **Normalisation des occurrences** : $t_w^d = x_w^d \log(1 + c\frac{avg_l}{l_d})$ (utilisée aussi pour le modèle vectoriel)
2. **Distribution de probabilité** : continue, « bursty », support = $[0, +\infty)$; loi log-logistique ou SPL
3. **Paramètre** λ_w : fixé ($\frac{N_w}{N}$), ou estimé
4. **Fonction de score**

$$\text{RSV}_{LL}(q,d) = \sum_{w \in q \cap d} q_w \log \frac{\lambda_w + t_w^d}{\lambda_w}$$

Modèles d'information (2)

Requêtes et documents sont comparés sur la base de l'information moyenne apportée par le document sur la requête :

$$\mathrm{RSV}(q,d) = -\sum_{w \in q} t_w^q \log P(T_w > t_w^d | \lambda_w) \quad (3)$$

1. **Normalisation des occurrences** : $t_w^d = x_w^d \log(1 + c\frac{avg_l}{l_d})$ (utilisée aussi pour le modèle vectoriel)
2. **Distribution de probabilité** : continue, « bursty », support = $[0, +\infty)$; loi log-logistique ou SPL
3. **Paramètre** λ_w : fixé ($\frac{N_w}{N}$), ou estimé
4. **Fonction de score**

$$\mathrm{RSV}_{LL}(q,d) = \sum_{w \in q \cap d} q_w \log \frac{\lambda_w + t_w^d}{\lambda_w}$$

Collections utilisées

	N	l_{avg}	# Queries
TREC-3	741856	261.134	50
TREC-6	528155	295.976	50
TREC-7	528155	295.976	50
TREC-8	528155	295.976	50
CLEF-3	169477	300.789	60

TABLE: Description (statistique) des collections

Résultats

	TREC-3	TREC-6	TREC-7	TREC-8	CLEF-3
BM25	*25.10*	*23.55*	*18.42*	*25.30*	39.96
LM_{DIR}	**26.85**	*24.27*	18.88	*25.43*	*39.38*
LGD	*25.60*	*24.57*	18.92	*25.87*	*39.51*
SPL	26.77	**25.19**	**19.09**	**26.28**	**40.42**

TABLE: Comparaison de différents modèles (probabilistes) de RI (MAP avec validation croisée et t-test à 0.1)

Table des matières

Conclusion

La recherche sur les modèles de RI est un domaine foisonnant :

- Reposant sur la modélisation de problèmes fondamentaux (pertinence, poids des termes, distribution des mots dans les collections)
- Fort ancrage actullement sur les modèles probabilistes, mais :
  - De nouvelles propositions voient régulièrement le jour (approche quantique)
  - Evolution marquée vers l'apprentissage supervisé pur les moteurs sur le Web (voir cours « RI et Apprentissage »)
- La RI sur les images est très proche (mais représentation BOkP moins riche sémantiquement)
- La RI devient de plus en plus sociale (de nouoveaux ingrédients sont nécessaires)

Quelques références (1)

Les références « historiques »

- S. P. Harter. *A probabilistic approach to automatic keyword indexing*, Journal of the American Society for Information Science, 26, 1975.
- S. E. Robertson. *The Probability Ranking Principle in IR*, Journal of Documentation, 33(4) :294–304, 1977.

Booléen, vectoriel (et autres)

- G. Salton & M. J. McGill. *Introduction to Modern Information Retrieval*, McGraw-Hill, Inc., New York, NY, USA, 1983.
- C. D. Manning, P. Raghavan & H. Schütze. *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA, 2008.

Quelques références (2)

BIM et BM25

- S. E. Robertson & S. Walker. *Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval*, SIGIR 1994.

Modèles de langue

- J. M. Ponte & W. B. Croft. *A language modeling approach to information retrieval*, SIGIR 1998.
- D. Hiemstra, S. Robertson & H. Zaragoza. *Parsimonious language models for information retrieval*, SIGIR 2004.
- C. Zhai & J. Lafferty. *A study of smoothing methods for language models applied to ad hoc information retrieval*, SIGIR 2001/ACM Trans. Inf. Syst., 22(2), 2004.

Quelques références (3)

Modèles d'information

- G. Amati & C. K. van Rijsbergen 02. *Probabilistic Models of Information Retrieval Based on Measuring The Divergence from Randomness*, ACM Transactions on Information Systems, Vol. 20(4), 2002.
- S. Clinchant & E. Gaussier. *Bridging language modeling and divergence from randomness models : A log-logistic model for IR*, ICTIR 2009.
- S. Clinchant & E. Gaussier 10. *Information-based Models for AD-Hoc IR*, SIGIR 2010.