## I. Vocabulaire

- **Doc :** Encodage + format + langue
- **Mot :** chaine délimitée
- **Terme :** mot normalisé
- **Token :** occurrence dans un doc

**Normalisation / lemmatisation :** réduction des variantes au radical (conjugaison, accords, etc.). La normalisation peut être problématique (segmentation des mots comme San Francisco, les nombres, les espaces, les mots composés, accents, casse, homonyme (MIT vs mit))

Les stop-words (mots très courants) sont à éliminer, sauf si on veut traiter les phrase queries.

Classes d'équivalence par soundex ou thesauri possibles.

## II. Modèle booléen

### 1. Principe

Chaque mot est une variable booléenne qui dépend du document. Recherche des documents valide vis-à-vis de la requête booléenne.

### 2. Index inversé

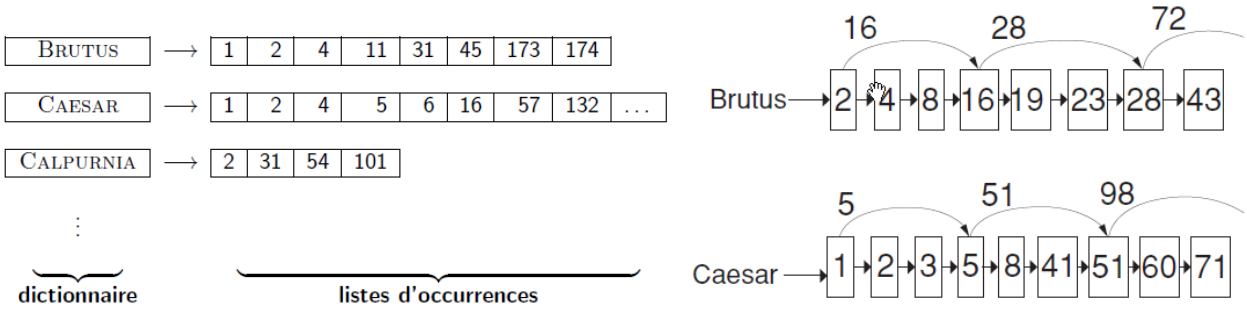
Pour chaque mot, on enregistre l'ensemble des documents qui le contiennent.

La requête revient à des combinaisons d'ensembles (intersection, union, …)

**Optimisation :** Traiter les termes dans l'ordre croissant de leurs fréquence, commencer avec les documents ayant la liste d'occurrence la plus courte. Si on a un OR, on peut estimer la taille de l'union par la somme du nombre d'occurrences.

**Skip pointers :** Permettent de réaliser la jointure plus rapidement. Pb du nombre et de leur longueur.

**Recherche de bouts de phrase ou de proximité :** indexer des paires de bi-termes ou les positions des termes par exemple.




### 3. Recherche tolérante (wildcard, correction orthographique, etc.)

On veut trouver un ou plusieurs termes de notre dictionnaire (terme, occurrences) correspondant à un token de la requête.

#### a. Wildcard sur arbre B

- mot* : facile
- *mot : besoin arbre B inversé
- mot*mot : intersection de mot* et *mot ou permuterm

**Permuterm :** on indexe toutes les permutations des mots (avec un caractère de fin de mot), et on permute mot*mot pour que * soit à la fin (mot$mot*)

**Index de k-gram :** On indexe les k-grams des lettres de chaque mot (ex : paires de lettres), et qui les associe aux mots correspondants possibles. On fait alors une intersection de plusieurs ensembles de mots (qui contient des faux positifs a filtrer)

### b. Distance d'édition (correction orthographique)

| | |
|---|---|
| coût de venir du voisin en haut à gauche (copie ou substitution) | coût de venir du voisin du dessus (suppression) |
| coût de venir du voisin de gauche (insertion) | minimum des 3 "mouvements"; coût minimal pour arriver ici |

| | | | f | | a | | s | | t | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | **0** | **1** | **1** | **2** | **2** | **3** | **3** | **4** | **4** |
| c | **1** | *1* | *2* | **2** | 3 | **3** | 4 | **4** | 5 |
| | **1** | *2* | *1* | **2** | **2** | **3** | **3** | **4** | **4** |
| a | **2** | **2** | **2** | *1* | *3* | *3* | *4* | 4 | 5 |
| | **2** | 3 | **2** | *3* | *1* | *2* | *2* | **3** | **3** |
| t | **3** | **3** | **3** | 3 | **2** | *2* | *3* | *2* | *4* |
| | **3** | 4 | **3** | 4 | **2** | *3* | *2* | *3* | *2* |
| s | **4** | **4** | 4 | **3** | **2** | 3 | *3* | *3* |
| | **4** | 5 | **4** | 5 | **3** | 4 | **2** | *3* | *3* |

**Distance de Levenstein :** Nombre d'opérations nécessaire pour passer de $s_1$ à $s_2$.

**Correction orthographique :** Remplacer chaque mot par le mot le plus proche dans un dictionnaire « bien orthographié » (à constituer…)

**Suggestion :** Recherche toutes les chaines à une distance inférieure à $C$ du mot entré, intersection avec le dictionnaire, et suggérer ces termes à l'utilisateur.

**Correction par k-grams :** On recherche les termes avec une forte proportion de k-grams communs avec le mot entré.

**Correction basée sur le nombre de documents :** Lancer plusieurs requêtes avec des variations des termes entrés et renvoyer celle proposant le plus de résultats.

**Correction basée sur l'historique des recherches**.

**Correction basée sur soundex** (p93 cours 3).

## III. TF-IDF & modèle vectoriel

Permet l'ordonnancement des résultats et la mesure de pertinence d'un document par rapport à la requête.

**Coefficient de Jaccard :** $\frac{|A \cap B|}{|A \cup B|}$

**Term frequency :** Fréquence du terme dans un document $tf_{t,d}^{log} = \max(0\,;1 + log\, tf_{t,d})$. On a pour chaque document sont vecteur de tf.

**Inverse document frequency :** Inverse du nombre de documents dans lequel un mot apparait $idf_t^{log} = \log \frac{N}{df_t}$ avec $N$ la taille du corpus et $df_t$ le nombre de documents contenant $t$.

**TF-IDF** est le produit $tf_{t,d}^{log} \times idf_t^{log}$. On normalise tous les vecteurs tf-idf.

**Requête :** On calcule le tf-idf de la requête et on fait le produit vectoriel avec tous les vecteurs tf-idf des documents. On ordonne les résultats.

## IV. Evaluation

Qualité d'un système RI : vitesse d'index et de recherche, cout de req.

| | pertinent | non pertinent |
|---|---|---|
| renvoyé | vrai positif (TP) | faux positif (FP) |
| non renvoyé | faux négatif (FN) | vrai négatif (TN) |

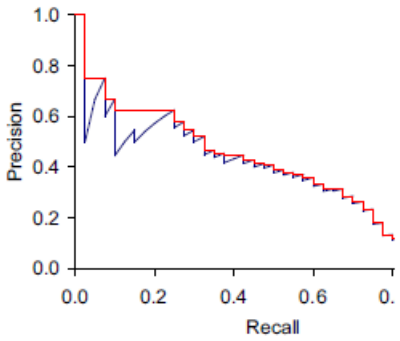
$$\text{Précision} = \frac{\#\text{ éléments pertinents renvoyés}}{\#\text{ éléments renvoyés}} = \mathbb{P}(\text{pertinent} \mid \text{renvoyé}) = \frac{\text{TP}}{\text{TP}+\text{FP}}$$

$$\text{Rappel} = \frac{\#\text{ éléments pertinents renvoyés}}{\#\text{ éléments pertinents}} = \mathbb{P}(\text{renvoyé} \mid \text{pertinent}) = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

**F-mesure :**

- La *F*-mesure permet un compromis entre rappel et précision
- $$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \quad \text{avec} \quad \beta^2 = \frac{1-\alpha}{\alpha}$$
- $\alpha \in [0,1]$ et donc $\beta^2 \in [0,\infty]$
- Le plus fréquemment on utilise l'équilibre entre précision et rappel avec $\beta = 1$ ou $\alpha = 0.5$
  - Il s'agit de la moyenne harmonique de *P* et *R*: $\frac{1}{F} = \frac{1}{2}\left(\frac{1}{P} + \frac{1}{R}\right)$

**Courbe précision rappel :**




Voir cours 8 pour la constitution de jeux de tests et la présentation des résultats.

## V. Retour de pertinence et expansion de requête

**Algo de Rocchio :** L'user indique les documents pertinents et la requête est recalculée en fonction.

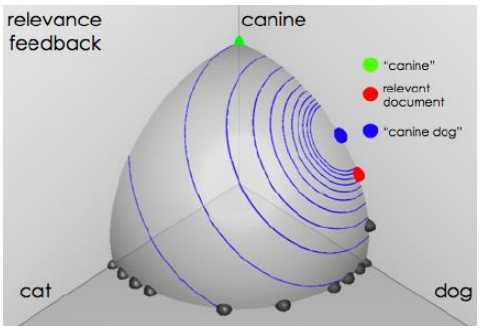
Pour cela on prend comme requête le centroïde des documents pertinents.

$$q_{opt} = \text{argmax}_q\left(q \cdot \mu_{D_r} - q \cdot \mu_{D_{nr}}\right) = \mu_{D_r} + \left(\mu_{D_r} - \mu_{D_{nr}}\right)$$

$D_r$ : relevent. $D_{nr}$ : non relevent.

- En pratique :
$$\begin{aligned}\vec{q}_m &= \alpha\vec{q}_0 + \beta\mu(D_r) - \gamma\mu(D_{nr}) \\ &= \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j\end{aligned}$$
  $q_m$ : vecteur requête modifiée ; $q_0$: vecteur requête initial ; $D_r$ et $D_{nr}$ : ensemble de documents identifiés comme pertinents et non pertinents; $\alpha$, $\beta$, et $\gamma$: poids
- Les requêtes ont tendance à se rapprocher des documents pertinents et à s'éloigner des non pertinents.
- compromis $\alpha$ vs. $\beta/\gamma$ : Si de nombreux documents sont marqués on veut un plus gros $\beta/\gamma$.
- Mettre les poids négatifs à 0
- "Les poids négatifs" associés à un terme n'ont pas de signification

Etiquetage positif $\beta = 0{,}75$ plus significatif que l'étiquetage négatif $\gamma = 0{,}25$.




**Pseudo-relevence feedback :** Pseudo-relevance feedback automatise la part manuelle du vrai retour de pertinence. Fournir une liste ordonnée de document à partir de la requête de l'utilisateur. Supposer que les k premiers documents sont pertinents Appliquer le retour de pertinence

# VI. Modèle probabiliste

Probabilité de pertinence : $R \sim \mathcal{B}$ $(1 :$ pertinent, $0 :$ non pertinent). On notera $R$ et $\bar{R}$.

Un document est pertinent si $\mathbb{P}(R|q,d) > \mathbb{P}(\bar{R}|d,q) \Leftrightarrow \mathbb{P}(R|q,d) > \frac{1}{2}$

## 1. Modèle Binaire Indépendant

$d \in \mathbb{B}^N$ vecteur où $d_i = 1$ si le terme $i$ est dans $d$ (0 sinon). Même principe pour $q$.

$$O = \frac{\mathbb{P}(R|d,q)}{\mathbb{P}(\bar{R}|d,q)} \propto \frac{\mathbb{P}(d|R,q)}{\mathbb{P}(d|\bar{R},q)} \propto \prod_{i,d_i=1} \frac{\mathbb{P}(d_i|R,q)}{\mathbb{P}(d_i|\bar{R},q)} \prod_{i,d_i=0} \frac{\mathbb{P}(\bar{d}_i|R,q)}{\mathbb{P}(\bar{d}_i|\bar{R},q)} = \prod_{i,d_i=1} \frac{p_i}{u_i} \prod_{i,d_i=0} \frac{1-p_i}{1-u_i}$$

- $p_i = \mathbb{P}(d_i|R,q)$ : proba que le terme $i$ soit dans un doc pertinent.
- $u_i = \mathbb{P}(d_i|\bar{R},q)$ : proba que le terme $i$ soit dans un doc non pertinent.
- On suppose que les termes absents de $q$ ont la même distrib dans $R$ et $\bar{R}$.

$$O \propto \prod_{i,d_i=1,q_i=1} \frac{p_i}{u_i} \prod_{i,d_i=0,q_i=1} \frac{1-p_i}{1-u_i} \propto \prod_{i,d_i=1,q_i=1} \frac{p_i(1-u_i)}{u_i(1-p_i)}$$

**Retrieval Status Value**

$$RSV = \sum_{i,d_i=1,q_i=1} c_i = \sum_{i,d_i=1,q_i=1} \log \frac{p_i}{1-p_i} + \log \frac{1-u_i}{u_i}$$

**Évaluation des $c_t$**

| Documents | pertinents | non pertinents | Total |
|---|---|---|---|
| $x_t = 1$ | $s$ | $\mathrm{df}_t - s$ | $\mathrm{df}_t$ |
| $x_t = 0$ | $S - s$ | $(N - \mathrm{df}_t) - (S - s)$ | $N - \mathrm{df}_t$ |
| Total | $S$ | $N - S$ | $N$ |

$$p_t = \frac{s}{S} \quad u_t = \frac{\mathrm{df}_t - s}{N - S}$$

$$c_t = \log \frac{s}{S-s} - \log \frac{\mathrm{df}_t - s}{(N - \mathrm{df}_t) - (S - s)}$$

$$\hat{c}_t = \log \frac{s+0.5}{S-s+0.5} - \log \frac{\mathrm{df}_t - s + 0.5}{(N - \mathrm{df}_t) - (S - s) + 0.5}$$

**Estimation en pratique**

- Le même raisonnement ne peut pas être tenu pour estimer $p_t$
- Différentes techniques :
  - Approximé par une fréquence d'apparition dans un ensemble de document connus comme pertinents
  - $p_t$ approximé par une constante (Croft et Harper, 1979)
  - $p_t$ proportionnel au log de la probabilité d'apparition de $t$ dans tout le corpus
  - $p_t$ estimé de façon itérative (approche probabiliste du retour de pertinence)

**Estimation en pratique**

- Les documents non pertinents sont ultra majoritaires
- Les statistiques sur les documents non pertinents sont approximées par les statistiques sur l'ensemble de la base

$$\log \frac{N - \mathrm{df}_t}{\mathrm{df}_t} \approx \log \frac{N}{\mathrm{df}_t}$$

- Revient au schéma de pondération idf

**Approche probabiliste du retour de pertinence**

1. Première description probabiliste de l'ensemble des documents (à partir d'une des autres techniques) ⇒ fournit un ensemble de documents $V$
2. Interaction avec l'utilisateur

$$VP \subset V \quad VNP \subset V$$

3. Réestimation de $p_t$ (et éventuellement $u_t$) par examen de $VP$ et $VNP$.

$$p_t = \frac{|VP_t|}{|VP|} \quad p_t^{(k+1)} = \frac{|VP_t| + \kappa p_t^{(k)}}{|VP| + \kappa}$$

**bilan**

$$RSV = \sum_{t:x_t=q_t=1} c_t \text{ avec } c_t = \log \left[ \frac{|V_i|}{|V| - |V_i|} \cdot \frac{N}{\mathrm{df}_t} \right]$$

- Indépendance des termes
- Les termes absents de la requête ne sont pas pris en compte
- La pertinence des documents est supposée binaire
- La pertinence des documents est supposée indépendante

## 2. Modèle de langue

On construit un modèle de langue $\mathcal{M}_d$ par document et on estime la proba que $\mathcal{M}_d$ ait généré $q$.

# VII. Indexation sémantique latente

$X \in \mathbb{R}^{n_{doc} \times n_{term}}$ matrice freq terme-doc. Décomposition SVD de $X$, compression, meilleurs résultats.