UNIVERSITÉ DES SCIENCES ET TECHNOLOGIE HOUARI BOUMEDIENE

FACULTÉ D'INFORMATIQUE - MASTER 2 SII

RECHERCHE D'INFORMATION
INFORMATION RETRIEVAL

CHAPITRE 3: Pondération des termes

06 octobre 2025

I. INTRODUCTION

- La pondération des termes est une étape importante dans le processus d'indexation dans la RI
- Elle permet d'attribuer une valeur numérique à chaque terme pour mesurer son importance dans le document
- La 1 ère valeur qui peut être utilisée c'est **la fréquence du terme** dans le document
- Un terme est important dans le document s'il est fréquent dans le document et moins fréquent dans la collection
- Une autre mesure qui peut être ajoutée à la fréquence du terme pour bien mesurer son poids c'est **sa fréquence dans la collection**.

II. APPROCHES DE PONDÉRATION

Objectif : Déterminer l'importance d'un terme dans un document et sa capacité à discriminer entre documents.

Pouvoir discriminatoire d'un terme

- ✓ Évalue à quel point un terme est utile pour distinguer les documents pertinents des non-pertinents.
- ✓ Terme très fréquent dans tous les documents = faible pouvoir discriminatoire.
- ✓ Terme rare ou spécifique = fort pouvoir discriminatoire.
- ✓ L'approche de pondération dépend du modèle de RI employé

II. APPROCHES DE PONDÉRATION

Modèle	Méthode de pondération	Idée principale
Vectoriel	**TF, IDF, TF-IDF**	Mesurer l'importance locale d'un terme dans un document, ajustée par sa rareté dans la collection.
Probabiliste	**BM25 et variantes**	Pondération basée sur la probabilité qu'un terme apparaisse dans un document pertinent, avec ajustement de longueur et fréquence.
LSI (Latent Semantic Indexing)	**TF-IDF + réduction SVD**	Pondération initiale suivie d'une projection dans un espace latent pour capturer la sémantique et les synonymes.
Modèle de Langage	**Probabilités avec lissage**	Chaque document est vu comme un générateur de texte, la pondération reflète la probabilité qu'il produise un terme.

III. PONDÉRATION PAR LA MESURE TF-IDF

- ✓ L'approche TF-IDF est l'une des plus connues en **Recherche d'Information** .
- ✓ Elle repose sur l'idée que les termes importants doivent avoir un **pouvoir discriminant** :
  - **TF (Term Frequency)** : fréquence d'un terme dans un document → mesure l'importance locale.
  - **IDF (Inverse Document Frequency)** : fréquence inverse d'un terme dans la collection → mesure la rareté d'un terme et donc son pouvoir discriminant.
- ✓ **TF capte l'importance d'un terme dans un document, tandis que IDF pondère cet effet selon la rareté du terme dans la collection.**

III.1. Term Frequency - TF

Il existe plusieurs formules pour calculer la fréquence d'un terme t_i dans un document d_j.

1) **TF brut (fréquence absolue)** $tf(t_i, d_j) = freq(t_i, d_j)$

   où $freq(t_i, d_j)$ est le nombre d'occurrences du terme t_i dans le document d_j.

2) **TF normalisé (par la fréquence maximale du document)**

$$tf(t_i, d_j) = \frac{freq(t_i, d_j)}{\max_{t \in d_j}(freq(t, d_j))}$$

3) **TF logarithmique (réduction de l'effet des grandes fréquences)**

$$tf(t_i, d_j) = 1 + \log\left(freq(t_i, d_j)\right)$$

III.1. Term Frequency - TF

4) TF avec K (Normalisation par longueur)

$$tf(t_i, d_j) = \frac{freq(t_i, d_j)}{K + freq(t_i, d_j)}$$

K est une constante pour tenir compte de la longueur des documents. Plus le document est long, plus cette formule limite l'influence des termes très fréquents.

5) Okapi TF (BM25, ajustement longueur du document)

$$tf(t_i, d_j) = \frac{freq(t_i, d_j) \cdot (k_1 + 1)}{freq(t_i, d_j) + k_1 \cdot \left(1 - b + b \cdot \frac{|d_j|}{avgdl}\right)}$$

C'est une évolution de TF avec K, largement utilisée dans les modèles probabilistes.
L'objectif est de pondérer un terme en tenant compte à la fois de sa fréquence dans le document et de la longueur du document, mais avec un ajustement plus fin grâce aux paramètres du modèle BM25.

- $|d_j|$:longueur du document d_j (nb total de termes)
- $avgdl$: average document length
- k_1 :paramètre de saturation du TF, contrôle la montée en poids des termes très fréquents (typiquement 1.2–2)
- b: paramètre de normalisation de la longueur du

III.2. INVERSE DOCUMENT FREQUENCY – IDF

IDF mesure la capacité d'un terme à discriminer entre documents.

Deux Formules :

1. $idf_i = \log\left(\frac{N}{n_i}\right)$
2. $idf_i = \log\left(\frac{N}{n_i} + 1\right)$ pour éviter zéro.

- N= nombre total de documents
- n_i =nombre de documents contenant le terme t_i

- ✓ L'IDF diminue le poids des termes fréquents dans la collection et augmente celui des termes rares.
- ✓ La 1 ère formule retourne zéro pour un terme qui existe dans tous les documents de la collection. Pour éviter ce problème nous utilisons la 2 ème dans la suite de nos cours

III.3. PONDÉRATION COMBINÉE TF-IDF

- Pondération finale : weight $(t_i, d_j) = TF(t_i, d_j) \times IDF(t_i)$
- **Avantages :**
  - Balance fréquence locale et discrimination globale
  - Très efficace pour la recherche de documents textuels
- **Limites :**
  - Ne prend pas en compte la proximité des mots
  - Ne gère pas les synonymes

IV. EXEMPLE PRATIQUE

- Pour calculer le poids d'un terme par TF*IDF, il suffit de choisir une formule de TF est al multiplier par une formule de l'IDF
- Soit la collection suivante de 3 documents :

D1 : { langage de programmation python est très utilisé pour le traitement de texte }

D2 : { le langage JAVA est basé sur le langage C++ }

D3 : { un langage de programmation est un langage utilisé pour traduire un algorithme en un programme }

Stopwords: { de, est, très, pour, le, un, en }

- N= nombre total de documents
- n_i =nombre de documents contenant le terme t_i

Donner le fichier inverse de la collection avec la formule :

$$\text{weight}(t_i, d_j) = \frac{\text{freq}(t_i, d_j)}{\max(\text{freq}(d_j))} \cdot \log\left(\frac{N}{n_i} + 1\right)$$

IV. EXEMPLE PRATIQUE

- Documents :

D1 : { langage, programmation, python, utilisé, traitement, texte }

D2 : { langage, JAVA, basé, langage, C++ }

D3 : { langage, programmation, langage, utilisé, traduire, algorithme, programme }

- Mots vides supprimés.

- Nombre de documents $N = 3$

1. Fréquence des termes dans chaque document

Terme	D1	D2	D3	ni
langage	1	2	2	3
programmation	1	0	1	2
python	1	0	0	1
utilisé	1	0	1	2
traitement	1	0	0	1
texte	1	0	0	1
JAVA	0	1	0	1
basé	0	1	0	1
C++	0	1	0	1
traduire	0	0	1	1
algorithme	0	0	1	1
programme	0	0	1	1
Max freq(dj)	1	2	2	-

2. TF normalisé (TF_norm)

$$TF_{\text{norm}}(t_i, d_j) = \frac{\text{freq}(t_i, d_j)}{\max(\text{freq}(d_j))}$$

Terme	D1	D2	D3
langage	1/1=1	2/2=1	2/2=1
programmation	1/1=1	0/2=0	1/2=0.5
python	1/1=1	0/2=0	0/2=0
utilisé	1/1=1	0/2=0	1/2=0.5
traitement	1/1=1	0/2=0	0/2=0
texte	1/1=1	0/2=0	0/2=0
JAVA	0/1=0	1/2=0.5	0/2=0
basé	0/1=0	1/2=0.5	0/2=0
C++	0/1=0	1/2=0.5	0/2=0
traduire	0/1=0	0/2=0	1/2=0.5
algorithme	0/1=0	0/2=0	1/2=0.5
programme	0/1=0	0/2=0	1/2=0.5

PONDÉRATION DES TERMES

3. IDF $IDF(t_i) = \log\left(\frac{N}{n_i} + 1\right)$

Terme	ni	IDF
langage	3	log(3/3 +1)=log2 ≈ 0.301
programmation	2	log(3/2 +1)=log2.5 ≈ 0.398
python	1	log(3/1 +1)=log4 ≈ 0.602
utilisé	2	log2.5 ≈ 0.398
traitement	1	log4 ≈ 0.602
texte	1	log4 ≈ 0.602
JAVA	1	log4 ≈ 0.602
basé	1	log4 ≈ 0.602
C++	1	log4 ≈ 0.602
traduire	1	log4 ≈ 0.602
algorithme	1	log4 ≈ 0.602
programme	1	log4 ≈ 0.602

PONDÉRATION DES TERMES

4. TF*IDF $\text{weight}(t_i, d_j) = TF_{\text{norm}}(t_i, d_j) \times IDF(t_i)$

Index ou Fichier Inverse

Terme	D1	D2	D3
langage	1 × 0.301 = 0.301	1 × 0.301 = 0.301	1 × 0.301 = 0.301
programmation	1 × 0.398 = 0.398	0 × 0.398 = 0	0.5 × 0.398 ≈ 0.199
python	1 × 0.602 = 0.602	0 × 0.602 = 0	0 × 0.602 = 0
utilisé	1 × 0.398 = 0.398	0 × 0.398 = 0	0.5 × 0.398 ≈ 0.199
traitement	1 × 0.602 = 0.602	0 × 0.602 = 0	0 × 0.602 = 0
texte	1 × 0.602 = 0.602	0 × 0.602 = 0	0 × 0.602 = 0
JAVA	0 × 0.602 = 0	0.5 × 0.602 ≈ 0.301	0 × 0.602 = 0
basé	0 × 0.602 = 0	0.5 × 0.602 ≈ 0.301	0 × 0.602 = 0
C++	0 × 0.602 = 0	0.5 × 0.602 ≈ 0.301	0 × 0.602 = 0
traduire	0 × 0.602 = 0	0 × 0.602 = 0	0.5 × 0.602 ≈ 0.301
algorithme	0 × 0.602 = 0	0 × 0.602 = 0	0.5 × 0.602 ≈ 0.301
programme	0 × 0.602 = 0	0 × 0.602 = 0	0.5 × 0.602 ≈ 0.301