REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE MOULOUD MAMMERI TIZI-OUZOU

FACULTE GENIE ELECTRIQUE ET INFORMATIQUE

DEPARTEMENT D'INFORMATIQUE

Mémoire

De fin d'études

En vue de l'obtention du diplôme de master en informatique

Option : Conduite de Projets Informatique

Thème

Expansion du modèle de pertinence avec la caractéristique entropie

Réalisé par :

M[r] SALMI Toufik.

Proposé par :

M[r] HAMMACHE. A

Promotion

2013/2014

Remerciements

Je remercie le Tout Puissant, qui m'a donné la force et la patience pour l'accomplissement de ce travail.

Je présente mes remerciements les plus sincères à mon promoteur Mr Hammache.A, pour avoir été très disponible, pour ses conseils, et pour m'avoir soutenue et encouragé tout le long de l'année.

Je tiens ensuite à remercier Chacun des membres du jury pour m'avoir fait l'insigne honneur d'accepter d'examiner mon travail.

Merci à ma famille et mes amis pour leurs soutiens et leurs encouragements. Je remercie également toute personne ayant contribué de prés ou de loin à l'aboutissement de cette quête.

Toufik

Dédicaces

Je dédie ce modeste travail A:

Mes très chers parents.

Mes sœurs et mes frères

Mes ami(e)s et mes collègues

Ma très chère fiancée et future épouse Kahina

La mémoire de mon ami kamel smahi

Toufik

Table des matières

Chapitre III : Présentation de l'approche et expérimentation

Liste des figures

Liste des tableaux

Introduction générale

Jusqu'à aujourd'hui, la croissance considérable du volume de données a considérablement évolué grâce à l'informatique. Réunir suffisamment de données était donc le principal problème de la recherche d'information, mais avec la masse de données, l'objectif principal est devenu de localiser et d'interpréter les informations afin d'en extraire des connaissances. Pour cela, des approches et outils sont nécessaires afin de faciliter l'accès à l'information.

Dans un système de recherche d'information, nous retrouvons principalement deux éléments : les documents qui peuvent être : un texte, un morceau de texte, une page Web ou bien une image et une requête qui exprime le besoin d'information de l'utilisateur.

Si un système de recherche d'information est un système qui permet de retrouver une information pertinente par rapport à une requête dans une grande collection de documents, alors, un document pertinent est un document qui doit contenir l'information que l'utilisateur recherche.

Les résultats de la recherche dépendent de l'utilisateur et la requête soumise et reflètent la qualité d'un système de recherche d'information.

Notre approche rentre dans le cadre de l’expansion de la requête initiale soumise au système de recherche d’information par l’utilisateur- car c’elle ci est souvent incomplète- par des termes d’expansion afin de l’améliorer pour retourner les documents pertinents contenant le besoin en information de l’utilisateur.

Pour ce la nous avons implémenté le modèle de pertinence dans le système terrier puis nous avons fait l’extension de ce modèle avec des formules en ce basant sur deux paramètres : la taille des documents et leur entropie.

Nous avons organisé notre mémoire comme suit :

Le premier chapitre est consacré à la recherche d’information en générale en décrivant principalement les concepts de base, les modèles de RI, les processus d’indexation et d’expansion de requêtes ainsi que l’évaluation des SRI.

Le second chapitre est consacré à l’expansion de requêtes et à ces principales étapes pour la sélection des termes ainsi que les différents types de méthodes d’expansion.

Le dernier chapitre est consacré à la présentation et l’évaluation de notre approche.

Chapitre I

Recherche d'information

1. Introduction

Le stockage et l'accès à l'information a toujours suscité un grand intérêt pour les informaticiens, notamment avec la croissance considérable du volume documentaire. C'est là que la recherche d'information attribut afin d'organiser et d'indexer ces informations pour faciliter l'accès pour l'utilisateur à ces bases documentaire.

Ce chapitre a pour but de présenter les principaux concepts et modèles de la recherche d'information (RI), ainsi que les approches d'évaluation de ces systèmes. Nous commençons d'abord par donner quelques définitions, puis nous décrivons les étapes d'un processus de RI, nous passons ensuite en revue les différents modèles de RI. Enfin, nous présentons les plus importants critères d'évaluation d'un SRI.

2. Définition de la recherche d'information (RI) : [65]

La recherche d'information (RI) est un ensemble d'outils (programmes informatiques) permettent de sélectionner dans une collection de documents ceux qui sont susceptibles de répondre au besoin en information d'un utilisateur. Ce besoin peut être exprimé de manière active sous forme d'une requête, nous parlons alors de recherche active, ou bien de manière passive, en se basant sur le profil utilisateur, nous parlons alors d'un système de filtrage d'information.

3. Concept de base de la recherche d'information [65]

Pour interroger une base documentaire, l'utilisateur exprime son besoin en information sous forme d'une requête qu'il transmet au système de recherche d'information (SRI). Le rôle de SRI, est de sélectionner et de retourner à l'utilisateur les documents jugés pertinents à la requête parmi tous les documents de la base documentaire.

La problématique majeure du SRI, est de retrouver dans un temps raisonnable, quelques centaines ou milliers de documents pertinents parmi des collections qui peuvent contenir des millions de documents. Pour ce faire, la requête, ainsi que tous les documents de la collection sont au préalable représentés sous une forme intermédiaire, propre au SRI, définie par le modèle de représentation. La façon dont ensuite ces représentations sont utilisées pour retrouver les documents pertinents est, quand à elle, est définie dans le modèle de recherche.

Dans ce qui suit, nous définissons les éléments clés de la recherche d'information que nous venons d'évoquer.

- **Le document [1]**

Le terme « document », du latin « documentum », a connu plusieurs définitions au cours de l'histoire. Le sens principal de ce terme a été « enseignement » jusqu'au dix-huitième siècle ou il devint « preuve » puis au vingtième siècle « document : toute base de connaissance, fixée matériellement susceptible d'être utilisée pour consultation, étude ou preuves ».De façon générale, le document est une trace d'activité humaine véhiculant une

information pouvant être lu et interprétée par une personne non nécessairement à l'origine de cette trace.

Avec l'avènement de l'informatique, le document a quitté son support matériel natif pour une représentation binaire dans les mémoires des ordinateurs. Suite de quoi, les SRI sont devenus une nécessité pour gérer ces documents.

- **Collection de documents**

La collection de documents (base documentaire) est constituée d'un ensemble de documents. Elle contient la totalité des informations exploitables et accessibles pas l'utilisateur.

La gestion de la collection des documents est assurée par le SRI. En générale, et pour une gestion optimale, le SRI représente l'ensemble des documents sous une représentation intermédiaire appelée index.

- **La requête**

La requête exprime le besoin en information de l'utilisateur, c'est elle qui initie le processus de recherche d'information.

La requête doit contenir des éléments clés suffisamment discriminants afin d'exprimer au mieux le besoin en information de l'utilisateur. La requête est écrite dans un langage d'interrogation compréhensible par le SRI, parmi les langages d'interrogation les plus utilisés :

- Le langage booléen : La requête est un ensemble de mots clés (termes de recherche) séparés par des opérateurs logiques.
- Le langage naturel : la requête est exprimée avec un ensemble de mots clés dans un langage libre.
- Le langage graphique : La requête est construite à partir d'une liste de mots clés connus pour se faire une vue d'ensemble représentant le contenu. Des documents sont offerts à l'utilisateur pour formuler sa requête.

4. *Le système de recherche d'information*

Nous définissons un système de recherche d'information (SRI) comme étant un système permettant de retrouver les documents pertinents à une requête d'utilisateur écrite dans un langage libre, à partir d'une base de documents volumineuse. Il repose sur les trois fonctions suivantes: stocker, organiser (indexer) et rechercher des données.

4.1. Architecture d'un système de recherche d'information [65]

Le système de recherche d'information repose donc sur les trois processus suivants : l'indexation, l'appariement et enfin la reformulation de requêtes que nous allons présenter un par un.

4.1.1. Processus d'indexation

Le processus d'indexation consiste à créer une représentation interne de la requête utilisateur et de tous les documents de collection de façon à refléter aussi fidèlement que possible leur contenu. Cette représentation consiste en un ensemble de descripteur, composés de mots et de groupes de mots souvent associés à des poids, qui forme le langage d'indexation. Ces descripteurs représentent le résultat du processus d'indexation et sont rangés dans une structure appelée dictionnaire ou encore index. Pour avoir une réponse satisfaisante du système de recherche d'information SRI l'indexation doit être efficace.

4.1.2. Processus d'appariement

Le processus d'appariement ou de comparaison consiste à mettre en correspondance la représentation de la requête avec les représentations des documents. Un degré de pertinence ou de similarité, noté « RSV » (Retrieval status value), entre la requête et chaque document de la collection est ainsi calculé, en ce basant sur ce degré, les documents qui sont jugés similaires par le SRI à la requête sont, par la suite, retournés à l'utilisateur. Ceux-ci sont généralement classés par ordre de pertinence

4.1.3. Processus de la reformulation de la requête

En plus des deux processus de base décrit précédemment, un SRI peut intégrer un processus de reformulation de la requête. Ce processus a pour but d'améliorer la performance du système en offrant un mécanisme de raffinement de la requête utilisateur. Les techniques utilisées durant ce processus se classent en deux catégories : les méthodes locales et les méthodes globales.la reformulation est détaillée dans le deuxième chapitre.

4.1.3.1. Les méthodes locales

Elles s'appuient sur la technique dite de réinjection de pertinence. En se basant sur La requête initiale, une liste de documents sera présentée à l'utilisateur, qui va ensuite choisir les documents qu'il juge pertinents. La requête initiale sera ensuite enrichie avec les termes des documents ainsi choisis.

4.1.3.2.Les méthodes globales

La méthode est réajustée en se basant sur des ressources externes telles que les bases lexicales, les thésaurus et les ontologies. Ainsi, des termes non nécessairement présents dans la requête initiale, sont suggérés à l'utilisateur pour raffiner sa requête.

5. Modèles de recherche d'information

Le modèle de recherche d'information repose sur les représentations de la requête et des documents issues de l'indexation. Il permet de leur donner une interprétation afin de déterminer les documents qui sont similaires à la requête.

Au cours du développement de la RI, plusieurs modèle ont été proposés. Dans ce qui suit, nous présentons les principaux modèles utilisés.

5.1.Le modèle booléen

5.1.1. Le modèle booléen de base

Le modèle booléen est le premier modèle utilisé en RI. Il se base sur la théorie des ensembles et l'algèbre de Boole. Les documents y sont représentés par l'ensemble des termes qu'ils contiennent. Et la requête y est représentée sous frome d'une équation logique contenant des termes reliés par des connecteurs logiques (ET, OU, NON).

Formellement ;

- Un terme est une requête.
- Si **q** est une requête alors **NON q** est une requête
- Si $\mathbf{q_1}$ et $\mathbf{q_2}$ sont des requêtes alors $\mathbf{q_1}$ **ET** $\mathbf{q_2}$ est une requête.
- Si $\mathbf{q_1}$ et $\mathbf{q_2}$ sont des requêtes alors $\mathbf{q_1}$ **OU** $\mathbf{q_2}$ est une requête.

La fonction d'appariement **RSV** (**d**, **q**) qui détermine si un document **d** répond à une requête **q** est calculée comme suit :

- RSV (d, t) = 1 si le terme t ∈ d, 0 sinon.
- RSV (d, NON q_i) = 1- RSV (d, q_i).
- RSV (d, q_i **ET** q_i)= RSV (d, q_i) × RSV (d, q_j).
- RSV (d, q_i **OU** q_i)= RSV (d, q_i) + RSV (d, q_j)- RSV (d, q_i) × RSV (d, q_j).

q_i et q_j étant des requête quelconques.

C'est donc une fonction assez simple caractérisée par un mode d'appariement exact (1 ou 0).

Ceci fait du modèle booléen un modèle facile à implémenter tout en étant tout à fait fonctionnel. Par ailleurs, c'est aussi un modèle très expressif qui offre à l'utilisateur la possibilité d'effectuer une sélection exhaustive et non ambigüe.

Cependant, ce modèle nécessite de l'utilisateur la connaissance du langage booléen, qui est assez complexe. De plus, l'utilisateur doit pouvoir exprimer très précisément son besoin en information, sans quoi, la réponse du système risque d'être insatisfaisante. Par ailleurs, l'inconvénient majeur de ce modèle est de ne pas pouvoir classer les documents par ordre de pertinence. Cet inconvénient se fait d'autant plus ressentir si la requête retourne un grand nombre de documents.

5.1.2. Le modèle booléen étendu

Le modèle booléen étendu a été introduit pour compléter le modèle booléen de base en tenant compte du poids des termes issus de l'indexation.

La requête reste la même que celle du modèle booléen de base, par contre, la fonction d'appariement utilise le poids des termes, qui sont préalablement normalisés pour être compris entre 0 et 1, afin de calculer un score représentant le degré de similarité entre un document et une requête. Pour des requêtes à un ou deux termes, elle est définie comme suit :

- RSV (d, t_i) = a_i.
- RSV (d, **NON** t_i) = 1- RSV (d, t_i)
- RSV (d, t_1 **ET** t_2) = $1-\sqrt{\frac{(1-RSV(d,t_1))^2+(1-RSV(d,t_2))^2}{2}}$
- RSV (d, t_1 **OU** t_2) = $\sqrt{\frac{RSV(d,t_1)^2+RSV(d,t_2)^2}{2}}$

T_i étant un terme quelconque et a_i son poids associé dans le document d.

Les requêtes complexes -contenants plusieurs termes- sont traitées récursivement en appliquant les règles précédentes, néanmoins, le modèle *p-norm* propose une méthode permettant de calculer le score de similarité d'un document par rapport à la disjonction et la conjonction de plusieurs termes. Cette méthode se base sur les p-distances définie comme suit :

- $RSV(d, t_1\ ET\ t_2\ ET\ \dots ET\ t_m) = \sqrt[p]{\frac{a_1^p+a_2^p+\cdots+a_m^p}{m}}$
- $RSV(d, t_1\ OU\ t_2\ OU\ \dots OU\ t_m) = 1 - \sqrt[p]{\frac{(1-a_1)^p+(1-a_2)^p+\cdots+(1-a_m)^p}{m}}$

t_1, t_2… t_m étant des termes quelconques et $a_1, a_2 \dots\ a_m$ leurs poids associés dans le document d le paramètre entier p $(p \geq 1)$ est indiqué au moment de la formulation de la requête.

L'avantage principal du modèle booléen étendu par rapport au modèle booléen de base est de pouvoir classer les documents suivant leur pertinence. Cependant, tout comme le modèle booléen de base, les requêtes restent complexes et difficilement formulées par l'utilisateur.

5.1.3. Le modèle des ensembles flous

Le modèle des ensembles flous est une extension au modèle booléen de base. Il se base sur la théorie des ensembles flous où l'appartenance d'un élément à un ensemble est considérée comme probable et non certaine.

La requête y est décrite dans le langage booléen. Mais contrairement au modèle booléen de base, les termes se voient associés des poids (normalisés entre 0 et 1). Ceux-ci indiquent, dans le cadre de la théorie des ensembles flous, le degré d'appartenance d'un terme à un document. Ils sont utilisés par la fonction d'appariement pour calculer le degré de similarité d'un document à une requête de la façon suivante :

- RSV (d, t_i) = a_i.
- RSV (d, NON t_i) = 1- RSV (d, t_i).
- RSV (d, t_i ET t_j) = min (RSV (d, t_i), RSV (d, t_j))
- RSV (d, t_i OU t_j) = max (RSV (d, t_i), RSV (d, t_j))

t_i et t_j étant des termes quelconques, et a_i le poids associé à t_i

L'avantage principal de ce modèle est d'offrir la possibilité de classer les documents par ordre de similarité nul à la tautologie t ou (NON t) et un score non nul à l'antilogie t ET (NON t) (t étant un terme quelconques). Ceci fait qu'un système utilisant ce modèle peut retourner des résultats inattendus pour certaines requêtes.

5.2. Le modèle vectoriel

5.2.1. Le modèle vectoriel de base

Le modèle vectoriel est le modèle qui le plus inspiré la recherche d'information après le modèle booléen. C'est un modèle qui utilise une représentation géométrique en considérant les termes d'indexation comme les démentions d'un espace d'information multidimensionnel. La requête, qui est exprimée en langage naturel, et les documents sont alors représentés avec des vecteurs. La pertinence d'un document par rapport à une requête est relative aux positions de leurs vecteurs respectifs dans cet espace. La figure illustre de façon graphique la représentation de deux documents ($d_1$et d_2) et d'une requête q dans un espace associé à deux termes.

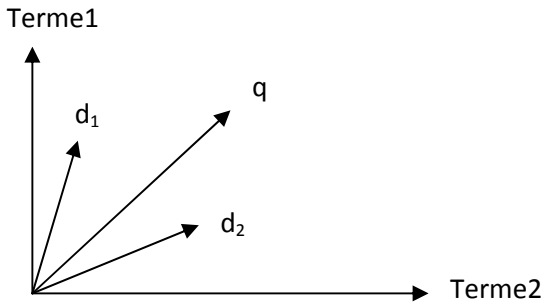

Figure I.1. Exemple d'une représentation du modèle vectoriel.

Dans un espace d'information à m terme T= $\{t_1,\ldots, t_m\}$, chaque composante W_i d'un vecteur $\vec{v}$ = $\{w_1,\ldots, w_m\}$ désigne le poids d'indexation ou terme t_1 dans le document ou la requête représenté(e) par ce vecteur. Ces composantes sont utilisé la fonction d'appariement pour mesurer la similarité d'un vecteur document $\vec{d}$ = ($w_{d,1} \ldots w_{d,m}$) à un vecteur requête

$\vec{q}$ = ($w_{q,1} \ldots w_{q,m}$). Les mesures de similarité les plus utilisé sont

- Le produit interne(ou produit scalaire) :

$$RSV\ (\vec{d}, \vec{q}) = \sum_{i=1}^{m} w_{q,i}.w_{d,i}$$

- Le cosinus de l'angle entre les deux vecteurs

$$RSV\ (\vec{d}, \vec{q}) = \cos(\vec{d}, \vec{q}) = \frac{\sum_{i=1}^{m} w_{q,i} \cdot w_{d,i}}{\sqrt{\sum_{i=1}^{m} w_{q,i^2} \cdot \sum_{i=1}^{m} w_{d,i^2}}}$$

- La mesure de similarité de Dice (rapport sur la moyenne arithmétique des normes)

$$\text{RSV}\left(\vec{d},\vec{q}\right) = dice\left(\vec{d},\vec{q}\right) = \frac{2 \cdot \sum_{i=1}^{m} w_{q,i} . w_{d,i}}{\sum_{i=1}^{m} w_{q,i}{}^2 + \sum_{i=1}^{m} w_{d,i}{}^2}$$

- La mesure de similarité de Jaccard (rapport de l'intersection sur l'union)

$$\text{RSV}\left(\vec{d},\vec{q}\right) = Jaccard\left(\vec{d},\vec{q}\right) = \frac{\sum_{i=1}^{m} w_{q,i} . w_{d,i}}{\sum_{i=1}^{m} w_{q,i}{}^2 + \sum_{i=1}^{m} w_{d,i}{}^2 - \sum_{i=1}^{m} w_{q,i} \cdot w_{d,i}}$$

- Le coefficient de superposition (overlap) :

$$\text{RSV}\left(\vec{d},\vec{q}\right) = overlap\left(\vec{d},\vec{q}\right) = \frac{\sum_{i=1}^{m} w_{q,i} . w_{d,i}}{\min\left(\sum_{i=1}^{m} w_{q,i}{}^2, \sum_{i=1}^{m} w_{d,i}{}^2\right)}$$

En plus de ces formules, il existe un grand nombre d'autres formules qui peuvent être utilisées pour calculer le degré de similarité d'un document à une requête. Le modèle vectoriel ne définit pas la meilleure formule, et le choix de la formule à implémenter est laissé au programmeur.

Le calcule du produit scalaire présente l'avantage d'être la méthode la plus rapide, mais retourne une valeur de similarité non normalisée. Quand au calcul de cosinus, c'est une méthode plus lente mais qui retourne une valeur de similarité normalisée (comprise entre 0 et 1).

L'avantage du modèle vectoriel est de pouvoir classer les documents par ordre de pertinence par rapport à une requête. Il offre aussi la possibilité de limiter le nombre de documents retournés, les SRI se basant sur ce modèle présentant en générale des résultats plus satisfaisants que ceux qui se basent sur le modèle booléen.

Néanmoins, ce modèle présente un inconvénient non négligeable, car il considère que les termes d'indexation forment une base, et donc qu'ils sont indépendants. Or, ceci est rarement le cas.

5.2.2. Le modèle vectoriel généralisé

Le modèle vectoriel généralisé a été proposé pour corriger les lacunes du modèle vectoriel de base, en considérant qu'il peut exister des dépendances entre les termes d'indexation. De ce fait on ne peut plus représenter les documents et les requêtes dans un même espace que celui du modèle vectoriel de base où les axes représentent les termes d'indexation, car ceux-ci sont orthogonaux et impliquent l'existence d'une indépendance entre termes.

Pour résoudre le problème, un nouvel espace est défini où les axes ne représentent plus les termes d'indexation, mais un ensemble de descripteur virtuels, appelés aussi *min-terms*, construit à partir des termes d'indexation en tenant compte de leurs cooccurrences dans les documents. Un document est alors représenté dans cet espace par un vecteur d définit comme suit :

$$\vec{d} = \sum_{i=1}^{m} w_{d,i} . \overrightarrow{x_i}$$

Où :

- m est le nombre de descripteurs virtuels présent dans cet espace
- $\overrightarrow{x_i}$ est un vecteur de base, associé au $i^{ème}$ descripteur virtuel
- $w_{d,i}$ est le degré de pertinence dans le document $\vec{d}$ di descripteur virtuel défini par $\overrightarrow{x_i}$

Le vecteur associé à la requête est défini de la même manière, et le degré de similarité entre une requête et un document est calculé en appliquant les même formules que c'elles utilisées dans le modèle vectoriel de base mais en utilisant les nouveaux vecteurs requête et document ainsi définis. De ce fait, le calcul du degré de similarité est plus couteux dans ce modèle, comparé au modèle vectoriel de base, mais il introduit la notion de dépendance entre les termes.

5.3. Le modèle probabiliste

5.3.1. Le modèle probabiliste de base

Le modèle probabiliste repose sur le principe de classement probabiliste (probability ranking principle) qui stipule qu'un système retournant les documents dans l'ordre décroissant de leur probabilité de pertinence par rapport à la requête, opère de manière optimale. Ainsi, le système doit estimer de manière aussi précise que possible la probabilité de pertinence d'un document par rapport à une requête exprimée en langage naturel.

Cette probabilité est notée p_q (pert\d_i), et exprime la probabilité que le document d_i fournisse des informations pertinentes pour la requête q. c'est une possibilité conditionnelle qui n'est pas directement calculable, est d'abord composée en utilisant le théorème de bayes :

$$p_q(\text{pert}\backslash d_i) = \frac{p_q(d_i\backslash pert)\,.\,p(pert)}{p_q(d_i\backslash pert)\,.\,p(pert) + p_q(d_i\backslash npert)\,.\,p(npert)}$$

Où $p(d_i\backslash pert)$ (respectivement p (d_i\npert)) indique la probabilité que d_i fasse partie de l'ensemble des documents pertinents (respectivement non pertinents) par rapport à la requête q. Quand à p (pert) et p (npert), elles indiquent respectivement la probabilité qu'un document quelconque soit pertinent ou non pertinent. Comme ces deux dernières probabilités sont fixes, on se limite aux deux valeurs p_q (d_i\pert) et p_q (d_i\npert) pour calculer le score de pertinence d'un document à une requête. Celles-ci sont calculées en se basant sur l'indexation binaire des documents où on considère que l'information la plus utile est la présence ou non d'un terme dans un document. De plus, on considère que les événements liés à la présence ou l'absence de termes d'indexation sont mutuellement indépendants. De par ces hypothèses, la fonction d'appariement définie comme suit :

$$RSV(d_i, q) = \log\frac{p_q(d_i\backslash pert)}{p_q(d_i\backslash npert)} = \sum_{j=0}^{m} x_{i,j}\cdot\left(\log\frac{p(t_j\backslash pert)}{1-p(t_j\backslash pert)}\right) + \left(\log\frac{p(t_j\backslash npert)}{1-p(t_j\backslash npert)}\right)$$

Où

- t_j représente un terme d'indexation, tel que $t_j \ \epsilon$ q, et q = $\{t_1, \dots, t_2\}$
- m représente le nombre total de termes d'indexation présents dans la requête q
- $x_{i,j} \ \epsilon \{0,1\}$ indique si le terme t_j est présent (1) ou non (0) dans le document d_j
- $p_q\left(t_j \backslash pert\right)$ indique la probabilité d'apparition du terme t_j sachant que le document appartient à l'ensemble des documents pertinents. Comme l'ensemble des documents pertinents n'est pas connu d'avance, une valeur fixe lui est généralement attribuée

$p_q\left(t_j \backslash npert\right)$ Indique la probabilité d'apparition du terme t_j sachant que le document appartient à l'ensemble des documents non pertinents. Cette probabilité correspond à df_j/n, Où n indique le nombre de documents de la collection, et df_j le nombre de documents indexés par le termet_j.

Le modèle probabiliste de base est un modèle tout à fait fonctionnel qui a démontré son efficacité en recherche d'information. Cependant il présente l'inconvénient de considérer les termes d'indexation comme étant indépendants les un des autres, ce qui n'est généralement pas le cas. En outre, c'est un modèle qui génère des calculs de probabilités relativement complexes par rapport à d'autres modèles.

5.3.2. Les modèles de langue

Les modèles de langue utilisent une approche différente de celles des modèles présentés jusqu'ici. Dans cette approche, la pertinence d'un document par rapport à une requête est calculée en se basant sur la probabilité que la requête puisse être générée à partir du document.

On crée un modèle statistique M_d pour chaque document *d*. la correspondance du document *d* à une requête *q* est déterminée par la probabilité p ($q\backslash M_d$) que la requête *q* puisse être générée à partir du modèle M_d.

Pour une requête *q* contenant les m termes {t_1, …, t_m} cette probabilité est calculée comme suit :

RSV (d, q) = p (q= (t_1, …, t_m)\$M_d$) = $\prod_{j=1}^{m} p(t_j \backslash d)$

Avec :

$$P\left(t_j \backslash d\right) = \frac{tf(t_j,d)}{\sum_{t \epsilon d} tf(t,d)}$$

Ou *tf* (t_j,d) indique la fréquence d'occurrence du terme t_j dans d.

L'inconvénient de cette méthode est que si un seul terme t_j de la requête n'apparait pas dans le document *d*, alors la probabilité p (q\M_d) sera nulle. Pour remédier à cela, on a recours à des techniques de lissage, qui consistent à affecter des valeurs non nulles aux termes qui n'apparaissent pas dans un document.

6. *Evaluation des systèmes de recherche d'information*

6.1. *Mesure d'évaluation*

L'évaluation est une étape importante dans l'élaboration d'un SRI. Elle permet de mesurer les caractéristiques du système et de fournir une base de comparaison entre différents modèles. Parmi ces caractéristiques on trouve notamment le temps de réponse du système et la qualité de réponse du système. La qualité d'un système est mesurée en comparant les réponses du système correspondent à celle que l'utilisateur espère, mieux est le système.

La comparaison des réponses d'un système pour une requête avec les réponses idéales nous permet d'évaluer les deux métriques suivantes:

$$Précision = \frac{documents\ pertinents\ retrouvés}{documents\ retrouvés}$$

$$Rappel = \frac{documents\ pertinents\ retrouvés}{documents\ pertinents\ dans\ la\ base}$$

En générale, on peut obtenir un taux de précision et de rappel aux alentours de 30%. Les deux métriques ne sont pas indépendantes. Il y a une forte relation entre elles: quand l'une augmente, l'autre diminue. Il ne signifie rien de parler de la qualité d'un système en utilisant seulement une des métriques. Ainsi, pour un système, on a une courbe de précision-rappel qui a en général la forme suivante :

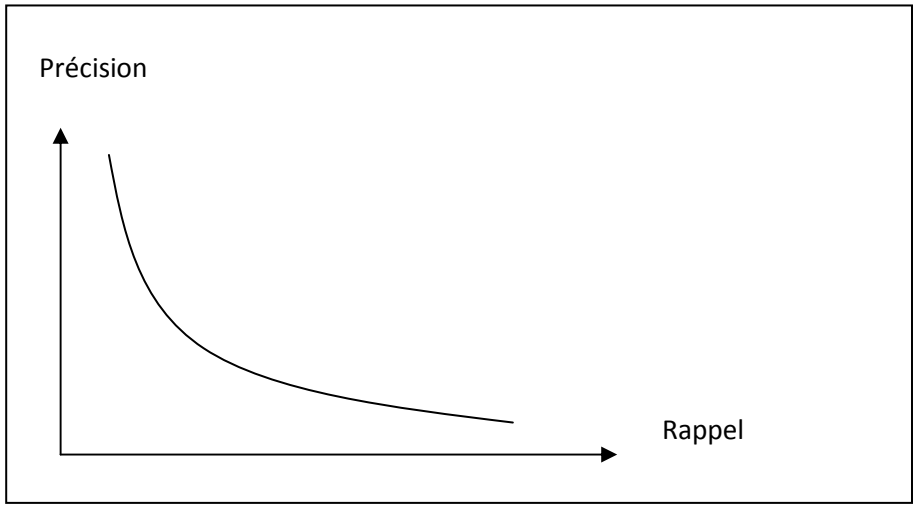

Figure I.2. Courbe générale de précision/rappel

6.2. compagnes d'évaluation et les collections de référence

Pour pouvoir réaliser de telle évaluation, on utilise des collections de testes fournies dans différentes compagnes d'évaluation. Une collection de tests est composée d'un ensemble de documents, de requêtes et de la liste des documents pertinents pour chaque requête.

Parmi les compagnes d'évaluation qui ont été entreprise au cours de l'histoire, on trouve la compagne de *cranfield* qui est considérée comme la première compagne d'évaluation, et les compagnes *TREC* qui constituent et que nous développons par la suite.

- **Les compagne TREC**

Les compagnes *TREC* (Text REtrieval Confirences) constituent l'approche la plus utilisée dans le domaine de l'évaluation des systèmes de recherche d'information. C'est un projet qui a été lancé en 1992 par le NIST (National Institue of Standards and Technology) avec le soutien financier de l'IARPA (Intelligence Advenced Research Project Activity). L'objectif de TREC et d'encourager les travaux dans le domaine de la recherche d'information, en fournissant l'infrastructure nécessaire (collection de tests) et en proposant des méthodologies d'évaluation des SRI.

La collection de testes qu'offre TREC comprend un ensemble de documents ainsi qu'un lot de thème (topics), chacun représentant un besoin en information. L'évaluation d'un SRI dans le cadre de la compagne d'évaluation TREC consiste à exprimer les 1000 premiers résultats qu'il retourne pour chaque thème (requête). Une précision moyenne est ensuite calculée et représente une mesure de performance globale du système.

7. Conclusion

Dans ce chapitre, nous avons présenté les principales notions et les principaux concepts de la recherche d'information. Nous avons développé les étapes d'un processus de recherche d'information, soit le processus d'indexation, le processus d'appariement et le processus de reformulation de la requête. Par la suite nous avons vu les principaux modèles de RI. Et enfin nous avons présenté les critères d'évaluation d'un SRI.

Jusqu'à présent nous nous sommes intéressés à la recherche d'information en générale. Dans le prochain chapitre nous allons nous intéresser à l'expansion des requête dans la RI.

Chapitre II

L’expansion de requêtes

1. Introduction

Les systèmes actuels de recherche d'information ont une boîte d'entrée unique qui accepte des mots-clés afin de rechercher des documents. Ces mots-clés soumis par l'utilisateur sont adaptés à l'indice de collection pour trouver les documents qui contiennent ces mots clés, qui sont ensuite triés par diverses méthodes.

Quand une requête de l'utilisateur contient plusieurs mots-clés sur des sujets précis qui décrivent avec précision son besoin d'information, le système est susceptible d'afficher de bons résultats. Cependant, étant donné que les requêtes des utilisateurs sont généralement courtes et que le langage naturel est ambiguë, ce modèle de récupération simple en général est source d'erreurs.

Nous allons voir dans ce chapitre que pour éliminer du moins minimiser les éventuelles erreurs que le système de recherche peut afficher nous utiliseront l'expansion de requête et ses différentes étapes, ainsi que les deux types d'approches qui traitent le problème d'expansion de requête.

2. Définition de l'expansion de requêtes

Plusieurs définitions d'expansion de requêtes, Query Expansion (QE) ont été données dans la littérature, nous en présentons dans cette section quelques unes.

Salton et McGill[38] définissent l'expansion de requêtes dans les systèmes de recherche d'information (SRI) comme un processus qui vise à rendre les résultats plus clairs et précis en permettant à l'utilisateur de modifier sa requête pour améliorer la pertinence de ses résultats. Selon Abberley [70], L'expansion de requêtes dans leurs systèmes permet de reformuler les requêtes et améliorer le processus de recherche d'information. Efthimiadis [71] qui a également proposé une classification des méthodes d'expansion de requêtes, donne la définition suivante: « L'expansion de requêtes est un processus qui vise à compléter la requête initiale en proposant des termes supplémentaires, elle est considéré comme une amélioration de la recherche d'information ». Il donne également les définitions suivantes : L'expansion de requêtes est une approche qui peut être appliquée quelle que soit la recherche ou les méthodes utilisées. La requête initiale telle qu'elle est saisie par l'utilisateur peut être une représentation inadéquate ou incomplète des besoins de l'utilisateur, soit de lui-même ou de la représentation des idées dans les documents, base de données, etc.

L'expansion de requêtes peut avoir lieu à la formulation de requête initiale, à la phase de reformulation de requête, ou bien les deux. Le concept plus général de modification de requête peut impliquer la suppression des termes de la requête. A l'ère du Web sémantique, Vechtomova et Wang [52] [69] présentent L'expansion des requêtes comme une amélioration de la formulation de la requête initiale dans la recherche de documents. Ces termes sont généralement choisis parmi les documents entiers, des paragraphes ou des parties du document où apparaissent les termes de la requête.

3. Les étapes de l'expansion de requête [5]

L'expansion de requête comprend quatre étapes principales que nous allons définir par la suite, l'entrée de ce processus c'est la requête initiale de l'utilisateur et en sortie on aura la requête reformulée avec les termes d'expansion. Nous schématisons ce processus avec la figure suivante :

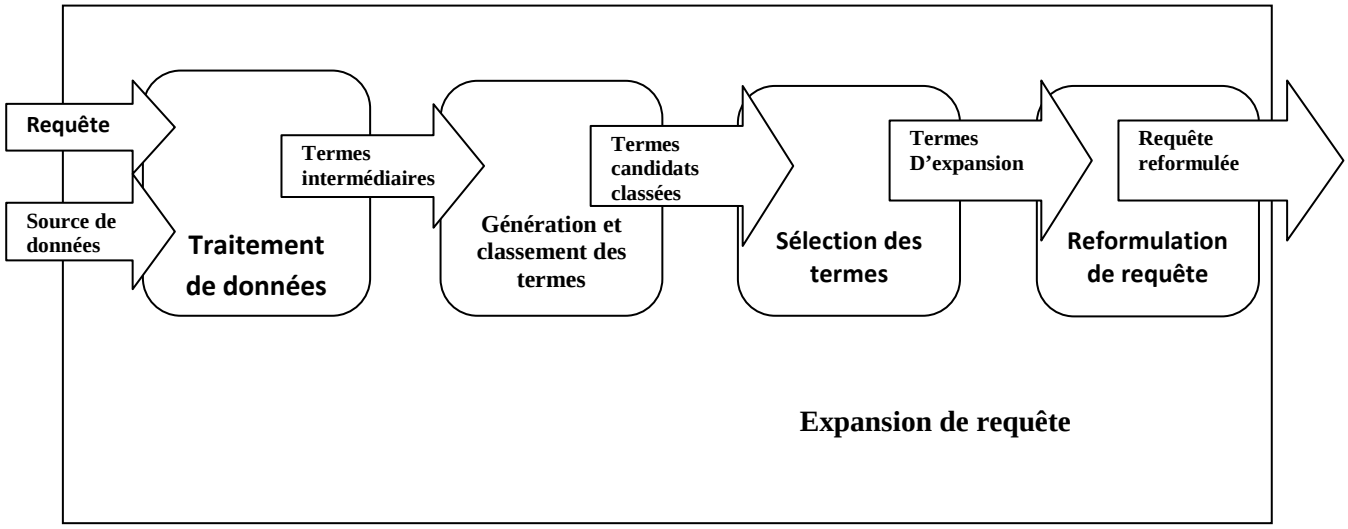

Figure II.1. Processus d'expansion de requête.

3.1.Traitement des données (requête et autres sources utilisées)

Cette étape transforme la première source de donnée utilisée pour étendre la requête de l'utilisateur dans un format qui sera traité plus efficacement par les étapes suivantes. Elle consiste habituellement d'une phase d'extraction de termes pour faciliter l'accès et la manipulation des fonctions de traitement.

Le prétraitement de la source de données (documents ou autre) est généralement indépendant de la requête de l'utilisateur qui doit être étendue, mais elle est spécifique au type de source de données.

De nombreuses techniques d'extension de requête sont basées sur les informations contenues dans les tops document, extraits en réponse à la requête de l'utilisateur initiale à partir d'une collection de documents. Dans l'étape de traitement de données, il est nécessaire d'indexer la collection et d'exécuter la requête.

L'indexation comprend habituellement :

(1) l'extraction de texte à partir de documents comme HTML, PDF, MS Word, etc.

(2) tokenization (c'est à dire, l'extraction de mots individuels, en ignorant la ponctuation).

(3) Elimination des mots vides (par exemple en enlevant des mots communs tels que les articles et les prépositions).

(4) Radicalisation (par exemple, dérivation de mots à leur forme racine),

(5) La pondération (l'attribution d'un score qui reflète l'importance du terme, généralement dans chaque document).

En conséquence, chaque document est représenté comme un ensemble de termes pondérés, avec un fichier complémentaire inversé de l'indice qui associe les termes du document aux termes de la requête. Le système d'indexation peut également stocker les positions des termes afin de fournir la recherche basée sur la proximité. Lorsque la collection utilisée pour l'expansion de requête est la même que celle en cours de recherche, le système de classement à le quelle la requête élargie sera soumise est généralement utilisé pour effectuer aussi un premier passage de classement. Si un corpus externe est utilisé (par exemple, des données Web pour les recherches sur l'intranet, ou données de bureau personnel pour recherches sur le Web), comme dans[7],[5], [72], et [2], un système de recherche d'information différent, en générale, est nécessaire; plusieurs options sont disponibles, telles que l'installation et le fonctionnement d'un moteur de recherche de bureau (commercial ou disponible gratuitement), en utilisant les récupérations API Web, ou encore le développement de son propre système pour l'indexation des documents et le classement.

D'autres techniques d'expansion de requêtes, basées sur l'analyse du corpus, nécessitent l'extraction des termes particuliers de la collection manuellement, qui sont généralement différents de celui utilisé à des fins d'indexation par un système de recherche d'information classique. Une approche bien connue est [Qiu et Frei 1993], où chaque terme est représenté comme un vecteur de documents pondéré en utilisant les statistiques de collection non-standard.
Un autre exemple est [9], qui construit un thésaurus de statistique en regroupant d'abord l'ensemble de la collection de documents via l'algorithme complet de clustering.

Certaines techniques d'expansion de requêtes nécessitent des procédures de prétraitement adaptés aux certaines sources de données. Par exemple, si l'expansion de requêtes utilise des textes d'ancrage, il faut analyser une collection de lien hypertexte pour extraire le contenu du texte de l'ancre balisé, et de continuer à traiter ces textes pour les normaliser et / ou de supprimer ceux qui contiennent trop peu ou trop de termes [9]. Les requêtes URL extraites des journaux d'archives des moteurs de recherche sont une autre source de données pour L'expansion de requêtes.

Dans les approches discutées jusqu'ici, le prétraitement est appliqué à une source de données précises. Telle est la situation prédominante, mais il ya des exceptions. La source de données peut être sélectionnée parmi plusieurs choix, comme dans [8] et [Il et Ounis 2007][10]. Une collection de FAQ (Frequently Asked Questions) est automatiquement construite en utilisant d'abord les requêtes Web tels que "inurl: faq" et appliquer ensuite des techniques d'apprentissage automatique pour extraire les FAQ réels de l'ensemble de pages récupérées.

3.2.Génération et Classement des termes candidats extension

Dans la deuxième étape de l'expansion de requêtes, le système génère et classe les termes candidats d'extension.

La raison pour laquelle le classement est important, c'est que la plus part des méthodes d'expansion de requête, ne pourront choisir qu'un petit nombre de termes candidats d'expansion à ajouter à la requête initiale.

L'entrée de cette phase est la requête d'origine et la source de données transformée; le résultat est un ensemble de termes d'expansion, généralement avec des scores associés. La requête initiale peut être prétraitée pour supprimer des mots communs et / ou extraire des termes importants pertinents.

Nous classons les techniques utilisées pour exécuter la génération et le classement des termes candidats selon le type de relation entre les termes d'expansion générés et les termes de la requête initiale (requête après prétraitement, le cas échéant).

3.2.1. Association un à un

La forme la plus simple de génération et classement des termes candidats est basée sur les associations un-à-un entre les termes d'expansion et les termes de la requête, dire, chaque terme d'expansion est lié à un terme unique de la requête. Dans la pratique, un ou plusieurs termes d'expansion sont générés et marqués pour chaque terme de la requête à l'aide d'une variété de techniques.

L'une de ces techniques consiste à s'appuyer sur les associations linguistiques, comme l'utilisation d'un algorithme de lemmatisation pour réduire des mots différents pour le même radical.

Une autre technique linguistique commune est de trouver des synonymes et mots connexes d'un mot de la requête à partir d'un thésaurus, le plus souvent utilisé est le WordNet.

Groupes de mots anglais dans des ensembles de synonymes appelés **synsets** sont enregistrés, il y'en a diverses relations sémantiques lexicales entre ces ensembles de synonymes. En particulier, il comprend Hyperonyme / relations hyponyme entre synsets nominaux qui peuvent être interprétées comme des relations de généralisation / spécialisation entre les ensembles correspondant à ces synsets.

La génération des termes d'expansion de WordNet implique de sélectionner un synset pour un terme de la requête, résolvant ainsi le problème de l'ambiguïté, puis traversant la hiérarchie en suivant ses liens typés. Afin de choisir un synset avec un sens similaire au terme de la requête, les termes de la requête adjacents peuvent être mieux adaptés aux ensembles présents dans chaque synset contenant le terme de la requête. Après avoir sélectionné le synset le plus pertinent, on pourrait envisager pour l'expansion de la requête tous les synonymes du terme de la requête dans le synset.

L'approche linguistique consiste à calculer automatiquement la similarité terme-à-terme dans une collection de documents.

L'idée générale est que les deux termes sont sémantiquement liés s'ils apparaissent dans les mêmes documents, tout comme deux documents sont considérés comme similaires s'ils contiennent les mêmes termes. Deux simples mesures de similarité sont le coefficient Dice et l'indice de Jaccard. Compte tenu des termes *u* et *v*, le coefficient Dice (D) est défini comme :

$$D = \frac{2.\mathrm{df}_{u \wedge v}}{\mathrm{df}_u + \mathrm{df}_v} \qquad \textbf{(II.1)}$$

Où $df_{u \wedge v}$ est le nombre de documents qui contiennent à la fois u et v, et df_u, df_v sont les nombres de documents contenants *u* et *v* respectivement. L'indice de Jaccard (J) est défini comme étant :

$$J = \frac{\mathrm{df}_{u \wedge v}}{\mathrm{df}_{u \vee v}} \qquad \textbf{(II.2)}$$

Où df_{uvv} est le nombre de documents contenants *u* ou *v*.

Une approche plus générale est la suivante. Considérons une matrice terme-document A, Où chaque cellule $A_{t,d}$ est un poids $W_{t,d}$ pour le terme *t* et le document *d*.
Si l'on calcule $C = AA^t$ alors C est une matrice de corrélation terme-terme, où chaque élément $C_{u,v}$ est une corrélation (similarité) score entre le terme *u* et *v* donnée par:

$$c_{u,v} = \sum_{\mathrm{dj}} \mathrm{w_{u,j}} . \mathrm{w_{v,j}} \qquad \textbf{(II.3)}$$

En utilisant la formule ci-dessus on peut calculer la corrélation entre chaque terme de la requête et chaque terme dans la collection de documents. Pour tenir compte de la fréquence des termes, il est préférable de produire des facteurs de corrélation normalisés, par exemple par la similitude de cosinus donnée comme suit : $\frac{c_{u,v}}{\sqrt{\sum_{dj} w_{u,u}^2 . \sum_{dj} w_{v,v}^2}}$ **(II.4)**

Selon la manière dont les documents et la fonction de pondération sont choisis, la formule (II.3) peut donner lieu à différentes méthodes de corrélation terme-à-terme.

Une technique bien connue, proposée dans [11], repose sur
L'ensemble des documents retournés en réponse à la requête initiale et utilise la fréquence des termes pondérés.

La cooccurrence des termes dans l'ensemble du document est simple, mais elle a l'inconvénient que la position n'est pas prise en compte, alors que deux termes qui apparaissent dans la même phrase semble plus corrélés que deux termes qui apparaissent loin l'un de l'autre dans un document. Cet aspect est généralement abordé en considérant la proximité des termes c'est à dire, en utilisant des documents textuels restreint tels que des documents de longueur fixe pour mesurer la cooccurrence des termes.

Cependant, la cooccurrence simple, que ce soit dans un contexte grand ou petit, ne signifie pas nécessairement que les termes sont corrélés. Une mesure plus complète pour l'association de mots qui incorpore la dépendance entre termes c'est l'information mutuelle [12], [13], défini comme suit:

$$\boldsymbol{I_{u,v} = \log_2 \left[\frac{P(u,v)}{P(u).P(v)} + 1 \right]} \qquad \textbf{(II.5)}$$

Où $P(u,v)$ est la probabilité conjointe que *u* et *v* apparaissent dans un certain contexte, habituellement un document, et P(*u*) et P(*v*) sont les probabilités d'occurrence des termes *u* et *v* respectivement.

L'information mutuelle est nulle en cas de cooccurrence zéro, égal à un si *u* et *v* sont indépendants, et égal à : $\log_2 \frac{1}{P(u)} + 1$ si *v est* parfaitement associé avec ***u***.

Un de ses inconvénients est qu'il a tendance à favoriser les termes rares sur les termes commun parce que $I(u,v)$ Augmenteront si $P(v|u)$ est fixé, mais P (u) diminue. Ce problème peut devenir plus aigu pour les données éparses. Sinon, nous pourrions envisager la définition

classique de la probabilité conditionnelle pour mesurer le degré de l'association de terme v au terme u:

$$P(v,u) = \frac{P(u,v)}{P(u)} \qquad \textbf{(II.6)}$$

La probabilité conditionnelle peut être calculée en divisant le nombre de contextes (par exemple phrases) dans laquelle les termes *u* et *v* coexistent par le nombre de contextes dans lesquels le terme *u* apparait. Cette approche (par exemple, [14], [15]) est similaire à la définition des règles d'association dans l'exploration des problèmes de données [16].

En fait, les règles d'association ont été explicitement utilisées pour trouver les termes d'expansion en corrélation avec les termes de la requête [17], [18].

Les termes d'expansion peuvent également être générés par l'utilisateur dans le but d'associer les termes de la requête originale avec des termes dans les dernières requêtes utilisées. Comme les textes extraits de ces données (éventuellement après prétraitement) sont généralement très courts, les techniques de corrélation standard basée sur la fréquence des termes ne peuvent pas être appliquées. En fait, plusieurs termes supplémentaires extraits des journaux de requêtes ont été utilisés pour aider à identifier les associations utiles, telles que l'examen des requêtes qui ont eu lieu dans la même session (par exemple, les requêtes successives émises par un seul utilisateur) ou requêtes qui ont donné des ensembles similaires de documents vraisemblablement pertinents (par exemple, en déployant le graphe biparti induit par les requêtes et les clics de l'utilisateur [19]).

Ces derniers types de preuve ne dépendent pas sur le contenu des requêtes et des documents et sont donc particulièrement utiles lorsque des approches basées sur le contenu ne sont pas applicables.

3.2.2. Associations un à plusieurs.

L'association un-à-un a tendance à ajouter un terme quand il est fortement lié à l'un des termes de la requête. Toutefois, cela peut ne pas refléter exactement les relations des termes d'expansion à la requête dans son ensemble.

Ce problème a été analysé dans [15]. Par exemple, alors que le mot «programme» peut ainsi être fortement associé au mot «ordinateur», une expansion automatique de toutes les requêtes contenant «programme» avec «ordinateur» pourrait bien fonctionner pour certaines requêtes (par exemple, «programme Java», «programme d'application»), mais pas pour d'autres (Par exemple, «programme de télévision», «programme de gouvernement», «programme spatial»). Ici encore, nous rencontrons la question de l'ambiguïté de la langue.

Le principe de l'approche associations un-à-plusieurs est d'étendre l'association un à un la technique décrite dans la section précédente pour les autres termes dans la requête. L'idée est que si un terme d'expansion est corrélé à plusieurs termes de la requête, donc il est corrélé à la requête dans son ensemble.

Dans [5], par exemple, il est nécessaire qu'un nouveau terme extrait du WordNet soit lié à au moins à deux termes de la requête originale avant qu'il soit inclut dans la requête étendue. Si nous utilisons des corrélations terme-à-terme, nous pourrions calculer les facteurs de corrélation d'un terme d'expansion candidat donné *v* pour chaque terme de la requête, puis combiner les scores trouvés pour trouver la corrélation de la requête *q* globale, par exemple, par la formule suivante :

$$c_{q,v} = \frac{1}{|q|}\sum_{u \in q} c_{u,v} \qquad \textbf{(II.7)}$$

Une approche similaire a été proposée dans [20] et [21], et suivie de plusieurs autres travaux de recherche [22], [15].

Les deux précédents articles sont intéressants non seulement parce qu'ils s'étendent le paradigme de la corrélation un à un à l'ensemble de la requête, mais aussi en raison de leurs fonctions de pondération particulières et les types des termes d'expansion.

Dans [20], la formule (II.3) est utilisée pour trouver des corrélations terme-terme dans toute la collection, considérée comme un espace concept-terme où les documents sont utilisés pour extraire les termes d'indexation. Le poids d'un terme dans un document est exprimé comme le produit de la fréquence du terme dans le document par la fréquence inverse du terme associée à ce document.

L'inverse de la fréquence du terme d'un document d_j est donné par $\mathbf{log}\frac{\boldsymbol{T}}{\boldsymbol{DT_j}}$ où T est le nombre de termes dans la collection et DT_j est le nombre de termes distincts dans le document d_j. Ce concept est similaire à l'inverse de la fréquence du document utilisé pour le classement des documents.

Un concept est un groupe de noms adjacents dans les documents les plus recherchés, et les concepts candidats sont analysés en utilisant des passages (c'est à dire, une portion de texte de taille fixe) à la place du document complet. La formule (II.3) est appliquée pour calculer la corrélation d'un terme-concept (au lieu d'une corrélation terme-terme), où $w_{u,j}$ est la fréquence du terme de la requête dans le j-ième passage et $w_{v,j}$ est la fréquence du concept *c* dans le j-ième passage.

La valeur de corrélation terme-concept exacte est déterminée en tenant compte de la fréquence inverse du terme et du concept dans les passages contenus dans l'ensemble de la collection. Les facteurs de corrélation de chaque terme de la requête unique à un concept donné sont ensuite combinés par une fonction de leur produit. Cette méthode est appelée l'analyse du contexte local.

L'approche étendue de l'association un-à-un peut être utile pour filtrer les termes d'expansion qui sont faiblement liées à certains termes de la requête, mais il n'est pas garantit que le terme d'expansion qui est fortement connecté à un seul terme sera écarté. Par exemple, si l'association de «ordinateur» avec «programme» est assez forte, «ordinateur» peut rester comme un terme d'expansion, même pour les requêtes «programme de télévision» ou «programme de gouvernement».

Ce problème peut être résolu en ajoutant des mots de contexte à une association terme à terme qui précisent les conditions dans lesquelles l'association est valide. Ces mots de contexte, par exemple, peuvent être dérivés comme conséquences logiques d'une base de connaissances, ou ils peuvent être extraits à partir d'un corpus en utilisant des cooccurrences de termes [15]. Considérant à nouveau notre exemple, si nous exigeons que «programme» apparaît avec «application» (ou «Java»), nous limitons l'applicabilité de l'association «programme» - «ordinateur» à des contextes appropriés.

Lorsque l'expansion de requête est basée sur WordNet, la nécessité est de relier les termes d'expansion à l'ensemble de la requête, et non à ses termes pris isolément. [5] ont montré que ces dernières techniques ne sont généralement pas efficaces car ils ne garantissent pas une bonne homonymie.

L'approche, associations un-à-plusieurs est basée aussi sur la combinaison de plusieurs relations entre les paires de termes dans un cadre de la chaîne de Markov [23]. Pour chaque requête, un réseau d'expression est construit, qui contient des paires de mots reliés par plusieurs types de relations, comme des synonymes, des radicaux, des cooccurrences, ainsi que les probabilités de transition. Ces relations peuvent être obtenues auprès de diverses sources

Ensuite, les mots ayant la plus forte probabilité de pertinence sont sélectionnés comme termes d'expansion, car ils reflètent le mieux les multiples aspects de la requête donnée.

Cette approche est plus robuste par rapport à la rareté des données et prend en charge des données complexes impliquant des chaînes de termes. Pour surmonter les limitations des relations entre les termes simples, on peut voir la requête comme une expression et de chercher des phrases qui lui sont liés. Les phrases fournissent généralement un contexte plus riche et une moindre ambigüité que leurs mots constitutifs, même si une évaluation de similarité au niveau de la phrase ne peut pas être simple.

3.3. Sélection des termes d'expansion

Après avoir classé les termes candidats, les éléments principaux sont sélectionnés pour l'expansion de la requête. La sélection se fait sur une base individuelle, sans tenir compte des interdépendances entre les termes d'expansion.

Ceci est bien sûr une hypothèse simplificatrice, bien qu'il existe quelques résultats expérimentaux qui semblent suggérer que l'hypothèse d'indépendance peut être justifiée [73]. Habituellement, seul un nombre limité de termes est sélectionné pour l'expansion, en partie parce que la requête qui en résulte peut être traitée plus rapidement, en partie parce que l'efficacité de récupération d'un petit ensemble de termes n'a pas nécessairement moins de succès qu'en ajoutant tous les termes candidat d'expansion, en raison de la réduction du bruit; par exemple, [24], [25].

Des recherches ont été effectuées sur le nombre optimal de termes à inclure et il ya des suggestions différentes, allant de cinq à dix termes [39] à quelques centaines [24], [26], [27]. D'autre part, la baisse de performance associée à des valeurs non optimales est généralement modeste [28], et la plupart des études expérimentales s'accordent à dire que le nombre de termes d'extension est de faible importance. Le choix typique est d'utiliser 10 à 30 termes.

Lorsque les scores des termes peuvent être interprétés comme des probabilités, on peut sélectionner uniquement les termes ayant une probabilité supérieure à un certain seuil; par exemple, p = 0,001, comme dans [29]. Plutôt que de se concentrer sur la recherche d'un nombre optimal de termes d'expansion, il peut-être plus pratique d'adopter des politiques de sélection plus éclairées**.** Il a été montré que les différentes requêtes ont un nombre varié de termes d'expansion optimal, et que de nombreux termes d'expansion [30], [31], [32]) - environ un tiers - sont nuisibles à la performance du repérage des documents. En fait, si l'on était en mesure de choisir exactement les meilleurs termes pour chaque requête, l'amélioration de la performance serait beaucoup plus élevée que d'habitude.

Pour aller au-delà d'une sélection simple fondée sur les poids attribués aux termes candidats, plusieurs méthodes qui emploient des informations supplémentaires ont été proposées. Une technique [33] consiste à utiliser plusieurs fonctions de classement de termes et en sélectionnant pour chaque requête les termes les plus courants.

Une idée similaire est exploitée dans [34], avec la différence que les multiples modèles feedback sont crées à partir de la même fonction de classement de termes et en générant des variantes de la requête initiale.

Une autre stratégie consiste à choisir une quantité variable de termes d'expansion en fonction de la difficulté de la requête. Dans [2], le nombre de termes d'expansion est en fonction de l'ambiguïté de la requête originale sur le web (ou dans le répertoire de données personnelles de l'utilisateur), telle que mesuré par le score de clarté [35].

Dans [36], les auteurs utilisent un classificateur pour faire la distinction entre la pertinence et la non pertinence du classement des termes d'expansion.

Pour apprendre les paramètres du classificateur (Support Vector Machine), un ensemble de mots simples sont étiquetés bon ou mauvais selon qu'ils améliorent ou nuisent au rendement, et chaque terme est décrit par un ensemble de fonctionnalités telles que la cooccurrence et la proximité avec les termes de la requête.

La sélection des meilleurs termes d'expansion pour une requête donnée est explicitement considérée comme un problème d'optimisation. En optimisant par rapport à un ensemble d'incertitude définis autour des données observées.

3.4.Reformulation de la requête

La dernière étape d'expansion de requête est la reformulation de la requête, à savoir comment décrire la requête étendue qui sera soumise au système de recherche d'information. Cela revient généralement à l'attribution d'un poids à chaque terme décrivant la requête élargi – appelé *pondération de requête.*

La technique de pondération des termes d'une requête la plus populaire est calquée sur la formule de Rocchio pour le retour de pertinence [37] et ses améliorations ultérieures [38], adaptée à la mise en AQE. La formule générale est la suivante, où q' est la requête étendue, q est la requête originale, λ est un paramètre de pondération, et le score t est un poids attribué au terme d'expansion t.

$$w'_{t,q'} = (1 - \lambda).w_{t,q} + \lambda.score_t \qquad \textbf{(II.8)}$$

Lorsque les termes d'expansion sont extraits des documents pseudo-pertinents et leur score est calculé en utilisant les documents, il est facile de montrer que le vecteur de requête étendue calculé par expression ci-dessus se dirige vers les documents pseudo-pertinents (selon les pondérations de document). Cependant, les avantages de la prise en compte de la différence de la répartition des termes entre les documents pseudo-pertinents et toute la collection pour-sélectionner les termes d'expansion- peut être réduite si nous pondérons ces termes.

La raison en est que les termes qui ont été correctement classés élevé seront pondérés à la baisse si leur valeur de pertinence par rapport à l'ensemble de la collection recherchée est faible.

Même un simple schéma de pondération sur la base d'une fonction inverse de classement de termes peut produire de bons résultats. Notez que les poids basés sur des documents utilisés pour la requête non expansée et les scores en fonction de différence de distribution utilisés pour les termes d'expansion ont différentes échelles, leurs valeurs doivent être normalisées avant de les additionner dans l'expression ci-dessus.

Plusieurs techniques de normalisation simples, discutées dans [40], ont été proposé; en général, elles produisent des résultats comparables, mais également accroître son uniformité pourrait être plus efficace [41].

La valeur de λ de l'expression ci-dessus peut être ajustée de façon à optimiser les performances, si les données sont disponibles. Un choix par défaut typique est de donner plus d'importance aux termes de la requête d'origine; par exemple deux fois plus que les termes d'expansion. autre possibilité est d'utiliser une formule de pondération de requête sans paramètres tels que proposé dans [42], compte tenu d'un paramètre de réinjection de pertinence, les auteurs

utilisent une approche d'apprentissage pour prédire la valeur optimale de λ pour chaque requête et chaque ensemble de documents de feedback, explorer un certain nombre de paramètre liées à la requête et aux documents (tels que la longueur, et la clarté) et à la divergence entre la requête et les documents feedback.

L'expression ci-dessus peut également être utilisée lorsque les termes d'expansion ont été extraits d'un thésaurus ou WordNet. Les pondérations peuvent être fondées sur des critères tels que le nombre de termes, le nombre de cooccurrences, la longueur du document, et le type de relation. Dans [5], par exemple, le vecteur de requête expansée est composé de sous-secteurs de onze types de concepts différents avec un poids important associé: un pour les termes de la requête d'origine, un pour les synonymes, et un pour chacun des autres types de relations contenues dans la partie nom de WordNet.

Si le classement des documents est effectué par le biais d'une approche de modélisation du langage, l'étape de pondération de requête est naturellement prise en charge.

Dans le cadre de base de la modélisation du langage, les documents les plus pertinents sont ceux qui minimisent la divergence de Kullback-Leibler entre le modèle de langue de requête et le modèle de langue du document:

$$sim(q,d) \propto \sum_{t\in v} P(t|\theta_q) \log \frac{P(t|\theta_q)}{P(t|\theta_d)} \qquad \textbf{(II.9)}$$

Dans la formule (II.9), le modèle de requête est généralement estimé en ne considérant que les mots de la requête d'origine, tandis que le modèle de document est estimé en tenant également compte des mots invisibles :

$$P(t|\theta'_d) = (1-\lambda)\cdot P(t|\theta_d) + \lambda \cdot P(t|\theta_c)$$

Ainsi, la question se pose de savoir s'il est possible de créer un meilleur modèle de requête pour trouver les mots liés avec leurs probabilités associées et ensuite utiliser le modèle d'expansion de requête correspondant (QEM) pour lisser le modèle de requête d'origine, de la même manière que le modèle document est lissé avec le modèle de la collection.

Diverses méthodes pour créer un modèle d'expansion de requête ont été explorées, fondée non seulement sur les documents de feedback [43], [29], mais aussi sur les relations des termes [44]. Indépendamment de la méthode de génération spécifique, le modèle final étendu est donné par la formule suivante :

$$P(t|\theta'_q) = (1-\lambda)\cdot P(t|\theta_q) + \lambda \cdot (t|\theta_{QEM})$$

Qui peut être vu comme une généralisation de l'expression (II.8).

La pondération de requête est commune dans AQE mais elle n'est pas toujours assurée.

Une approche alternative simple est d'augmenter le nombre de terme décrivant la requête sans effectuer la pondération requête.

Autre approche consiste à augmenter le nombre de termes de la requête, puis l'application d'une version modifiée de la fonction de pondération utilisée par le système de classement (de manière explicite traiter avec les termes d'extension), à la différence du classement des documents en utilisant la fonction de pondération de base du système en conjonction avec une requête élargie pondérée. Un exemple bien connu est [47], utilisé pour étendre l'Okapi BM25 fonction de classement [Robertson et al. 1998]. Dans d'autres cas, il est produit une requête booléenne [48], [45], ou, plus généralement, une requête structurée [Collins-Thompson et Callan 2005][23]. Il a été montré que la probabilité de l'opérateur ET, en combinaison avec des termes de la requête expansée au maximum, était très efficace pour l'expansion de la requête.

4. Approches pour l'expansion de requêtes

4.1.Approches Globales

Les techniques de ce type est d'analyser le contenu d'une base documentaire complète pour identifier les termes utilisés de façon similaire. La plupart des premières approches statistiques d'AQE étaient les corrélations entre paires de termes générés par l'exploitation des termes de cooccurrences, soit au niveau du document ou dans des contextes plus restreints tels que des paragraphes, des phrases. Les termes de concept [20] et le regroupement des termes [46], [49].

D'autres approches pour la construction d'un thésaurus sont décrites dans [52], [51] et [50], ce qui nécessite l'utilisation d'un vecteurs de contexte, l'information mutuelle, l'indexation sémantique. Ce paradigme d'AQE a également été récemment étendu avec de bons résultats à des documents multimédias [53]. Notez que depuis que les techniques globales sont guidées par les données, ils ne peuvent pas toujours avoir une interprétation linguistique simple.

4.2.Approches locales (basées sur les résultats de recherche)

Les Techniques spécifiques à la requête bénéficient du contexte local fourni par la requête. Elles peuvent être plus efficaces que les techniques spécifiques corpus car celle-ci pourraient être basées sur les termes qui sont fréquents dans la collection, mais pas pertinentes pour la requête

Les techniques spécifiques de requêtes utilisent généralement des documents de premier rang.

La plupart des méthodes couramment utilisées sont l'analyse de la différence de distribution de termes et le modèle d'expansion de requête de base.

Un autre type de recherche sur les techniques spécifiques de la requête est basée sur le Prétraitement des documents récupérés pour filtrer les termes pertinents avant l'utilisation du classement des termes. Plusieurs méthodes pour trouver des documents de représentations plus compactes et d'informations ont été proposées, comme l'extraction des passages [21] et le texte résumé [54].

Dans [Chang et al. 2006], les résumés de documents passent par un autre processus de regroupement et de classification dans le but de trouver un ensemble encore plus réduit de termes orthogonaux décrivant chaque document. Dans ce cas, le regroupement est utilisé pour extraire intra-document.

5. Expansion de requête dans les modèles de RI.

La requête initiale de l'utilisateur, est souvent trop courte, elle ne satisfait pas son besoin en information, ce qui fait que les résultats obtenus avec cette requêtes ne sont pas pertinents.

La reformulation de requête est utilisée afin d'améliorer lés résultats de la première recherche basée sur la requête soumise par l'utilisateur, pour trouver des documents plus pertinents en trouvant des termes d'extension et les pondérer à la nouvelle requête.

5.1.Expansion dans le modèle vectoriel.

La méthode de Rocchio [67] est largement utilisée en recherche d'information, Rocchio considère que pour restituer les documents pertinents, il faut maximiser la différence entre le vecteur des documents pertinents et celui des documents non pertinents.

Le but de la reformulation est donc de rapprocher le vecteur de la requête initiale du vecteur moyen des documents pertinents, ceci est mis en œuvre en ajoutant de nouveaux termes à la requête initiale. La forme standard de l'algorithme de Rocchio est donnée comme suit :

$$Q_{nlle} = \alpha Q_{init} + \frac{\beta}{|D_r|}\sum_{D_j \in D_r} D_j - \frac{\gamma}{|D_n|}\sum_{D_j \in D_n} D_j$$

Où :

$|.|$: Désigne le cardinal d'un ensemble.

Q_{nlle} : Le vecteur de la nouvelle requête.

Q_{init} : Le vecteur de la requête initiale.

D_r : L'ensemble des documents restitués et jugés pertinents par l'utilisateur.

D_n : L'ensemble des documents restitués et jugés non pertinents par l'utilisateur.

D_j : Le $j^{ème}$ document d'un ensemble.

$\alpha, \beta \; et \; \gamma$: sont des documents.

Le nouveau vecteur de la requête est le vecteur de la requête initiale plus les termes qui différencient au mieux les documents pertinents des documents non pertinents. Une requête reformulée contient les termes originaux et les nouveaux termes extraits des documents jugés pertinents.

5.2.Expansion dans le modèle de Probabiliste :

Dans le modèle probabiliste, nous trouvons que Robertson et al [47] ont pris en compte le jugement de l'utilisateur sur la pertinence des documents restitués par le système pour développer les formules de pondération. La similitude initiale documents-requête est calculée avec la formule suivante :

$$Sim(Q_k, D_j) = \sum_{i=1}^{r} q_{ki} * d_{ji} * \log\frac{p_i(1-U_i)}{U_i(1-p_i)} + C$$

Où :

$p_i = p\left(d_{ji} = \frac{1}{D}\ est\ pertinent\right)$

$U_i = p\left(d_{ji} = \frac{1}{D}\ est\ non\ pertinent\right)$

d_{ji} : Poids du terme t_i dans le document d_j avec $d_{ji} = 1$ si le terme i occurre dans le document d_j ,0 $sinon$.

q_{ki} : Poids du terme t_i dans la requête q_{ki}.

$p_{init} = 0$

$U_{init} = \frac{n_i}{N}$

C : Une constance.

n_i : Le nombre de documents de la collection contenant le terme t_i .

N : Le nombre total de documents jugés non pertinents.

Les recherches ultérieures exploitent l'occurrence des termes dans les documents jugés pertinents et document jugés non pertinents. Une liste de termes candidats à l'expansion de requête est triée selon une valeur de sélection donnée par la formule :

$$VS(t_i) = log\frac{p_i * r_i(1-q_i)}{q_i * R(1-p_i)}$$

Où :

p_i : Poids du terme t_i dans le document jugé pertinent.

q_i Poids du terme t_i dans la requête q.

R : Nombre de document jugés pertinents

Avec :

$p_i = \frac{r_i}{R}$

$U_i = \frac{n_i - r_i}{NR - R}$

r_i : Nombre de documents jugés pertinents, contenants le terme candidat t_i.

NR : Nombre de documents non pertinents retrouvés.

La fonction de similitude utilisée lors des itérations feedback devient alors la suivante :

$$sim(Q_k, D_j) = \sum_{i=1}^{r} q_{ki} * \log\left(\left(\frac{r_i}{R - r_i}\right) + \frac{(n_i - r_i)}{(N - NR - n_i + r_i)}\right)$$

5.3.Expansion de requête dans le modèle de langue.

La reformulation dans le modèle de langue part d'un principe différent des approches traditionnelles de RI ; on ne tente pas de modéliser directement la notion de pertinence, mais on considère que la pertinence d'un document face à une requête est en rapport avec la probabilité que la requête puisse être générée par le modèle de langue d'un document. On suppose qu'on effet, qu'a chaque document vis-à-vis d'une requête q est déterminé par la probabilité de génération de la requête sachant le modèle de ce document, soit $p(q|M_d)$.

La plus part des modèle de langue développés pour la RI se base sur ce principe de génération de la requête par un modèle de document, néanmoins, il existe d'autre variantes qui sont discutées dans ce qui suit.

5.3.1. Approche d'exploitation des modèles de langue en RI

En se basant sur la représentation des documents et la fonction de « Ranking », les approche de modélisation de langue pour la RI peuvent être classés en trois catégories :

1) Génération de la requête par le modèle de document (query Likelihood models) : cette catégorie correspond à ce que nous avons expliqué ci-dessus.
   Un modèle de langue est associé à chaque document. Les documents sont alors classés selon leurs probabilités de génération de la requête, soit $p(q|M_d)$.
2) Génération de document par le modèle de la requête (Document Likelihood models) : cette approche procède dans le sens inverse. Ainsi, un modèle de langue est associé à la requête, les documents sont alors classés selon leurs probabilités que leur contenu soit généré par le modèle de la requête, soit $p(d|M_q)$.
3) Similarité document-requête : dans cette catégorie, on considère qu'a chaque document et chaque requête est associée à un modèle de langue. Les documents sont alors ordonnés selon la similarité de leurs modèles avec celui de la requête.

5.3.1.1.générations de la requête par le modèle de document (Query Likelihood Models)

Ponte et Croft [62] furent les premiers à proposer un modèle de langue pour RI. L'intuition derrière leur modèle est que l'utilisateur est supposé avoir une idée des termes qui sont présents dans le document pertinent. L'utilisateur génère la requête en utilisant ces termes. Ainsi la requête q est censée fournir des indice (des termes), associés au modèle du document(les documents recherchés).

On cherche alors à estimer la probabilité que la requête provienne du modèle ayant généré ce document, soit :

$$score\ (q, d) = \ (q|M_d)$$

Ce modèle utilisant une distribution de Bernoulli, c'est-à-dire que non seulement les mots présents dans la requête, mais aussi ceux absent de la requête sont pris en compte. Le score du document vis-à-vis de la requête est alors exprimé ainsi :

$$score(q,d) = \prod_{t \in q} p(t|M_d) * \prod_{t \in q} \left(1 - p(t|M_d)\right)$$

Ce score est composé de deux parties : la probabilité d'observer les termes de la requête dans le document et la probabilité de ne pas observer les termes absents de la requête dans le document.

La probabilité $p(t|M_d)$ est calculée par une méthode non paramétrique qui utilise la probabilité moyenne d'apparition du terme t dans les documents qui le contiennent p_{avg}(t) et un facteur de risque pour un terme observé dans le document $R(t,d)$. par contre la probabilité d'un terme dans la collection est utilisée pour les termes qui n'apparaissent pas dans le document. Le calcule de cette probabilité est exprimé ainsi :

$$p(t|M_d) = \begin{cases} P_{ML}(t|d)^{(1-R(t,d))} * P_{Avg}(t)^{R(t,d)} & si\ \ tf(t,d) > 0 \\ \dfrac{tf(t,C)}{|C|} & sinon \end{cases}$$

Hiemstra [68] a proposé un modèle où la requête est considérée comme une séquence de termes $q = (t_1, t_2, \dots, t_n)$. Dans ce modèle, on ne considère que les mots présents dans la requete, une distribution multi nominale est utilisée. Le score d'un document vis-à-vis d'une requête (la probabilité de générer une requête q sachant le modèle de document M_d est donné par la formule suivante :

$$score(q,d) = p(d) \prod_{i=1}^{n} p(t_i|M_d)$$

Où : $p(d)$ est la probabilité a priori d'un document.

Pour l'estimation de la probabilité $p(t_i|M_d)$, Hiemstra utilise l'approche par interpolation qui combine le modèle de document M_d avec le modèle de langue de la collection. Le calcule de cette probabilité est exprimé ainsi :

$$p(t_i|M_d) = \lambda P_{ML}(t_i|M_d) + (1-\lambda) P_{ML}(t_i|C)$$

Où :

λ : est le poids d'interpolation qui varie entre 0 et 1. Ce paramètre peut être considéré constant ou il peut être estimé d'une manière sophistiquée en utilisant un processus d'optimisation automatique tel que l'algorithme de maximisation d'espérance(EM).

Les modèles $P_{ML}(t_i|M_d)$ et $P_{ML}(t_i|C)$ sont estimés en se basant sur la vraisemblance maximale, exprimés ainsi.

$$P_{ML}(t_i|C) = \frac{df(t_i)}{\sum_{t_j \in v} df(t_j)}$$

$$P_{ML}(t_i|M_d) = \frac{tf(t_i, d)}{|d|}$$

Tels que :

$tf(t_i, d)$: Est la fréquence du terme t_i dans le document d.

df (t_i) : Est le nombre de document contenant le terme t_i.

$|d|$: Est la taille du document.

v : Est le vocabulaire d'index

5.3.1.2.Génération de document à partir du modèle de la requête (Document Likelihood Models)

Au lieu de modéliser la RI comme processus de génération de la requête, Lavrenko et Croft[62] on propose de modéliser explicitement le modèle de la requête sans utiliser les données d'entrainement, en faisant le parallèle avec la modélisation de la pertinence proposée dans le modèle probabiliste classique. Ils considèrent en effet, que pour chaque requête, il existe un modèle permettant de générer le sujet (thème) abordé par la requête, c'est ce que les auteurs appellent le modèle de pertinence (θ_R).

Le but est alors d'estimer la probabilité $p(t|\theta_R)$, de générer un terme à partir du modèle de pertinence. Comme le modèle de pertinence n'est pas connu, les auteurs ont suggéré d'exploiter les documents retournés les mieux classés (feedback documents) en assumant qu'ils sont générés du modèle de pertinence.

Ce modèle est formalisé comme suit :

$$P(t|\theta_R) = \sum_{d \epsilon R} p(d)p(t|d) \prod_{i=1}^{k} p(q_i|d)$$

Avec :

R : L'ensemble des tops documents pertinents.

$p(d)$: La probabilité de choisir un document d des tops documents pertinents retournés.

Ainsi, le modèle de pertinence obtenu est une combinaison pondérée du modèle individuel de chaque document feedback $p(t|d)$ avec le score de ce document vis-à-vis de la Requête $p(q_i|d)$.

Les résultats des expérimentations ont montré que cette approche améliore sensiblement les performances de la recherche d'information, de +10% à +29% d'amélioration de précision moyenne par rapport au modèle de langue de base[1]

6. conclusion :

Ce chapitre a porté essentiellement sur l'expansion de requête, nous avons présenté les différentes étapes de ce processus, ainsi que les modèles de recherche d'information.

L'expansion de requête est une phase importante du processus de recherche d'information. Elle consiste de manière générale à enrichir la requête initiale de l'utilisateur en ajoutant des termes permettant de mieux exprimer son besoin. Cette technique peut être appliquée automatiquement ou d'une façon interactive, c'est-à-dire avec l'intervention de l'utilisateur.

Nous décrivons dans le chapitre suivant notre approche de sélection de documents pour l'expansion de requête ainsi que les résultats obtenus de l'évaluation.

Chapitre III

Evaluation & Expérimentations

1. Introduction

Nous avons abordé dans le chapitre précédent les différentes approches pour l'expansion de requêtes, dans ce chapitre nous allons présenter une nouvelle approche pour améliorer encore la pertinence des termes à ajouter à la requête initiale afin de bien exprimer le besoin en information de l'utilisateur.

Le chapitre suivant est structuré comme suit :

En premier lieu nous allons présenter notre approche en introduisant un exemple illustratif, puis nous parlerons des outils utilisés pour l'implémentation de notre approche sous la plate forme Terrier et enfin quelques résultats expérimentaux obtenus sur la collection de teste TREC AP88.

2. L'approche utilisée

2.1.Architecture générale de notre approche

La figure suivante illustre l'architecture de notre approche.

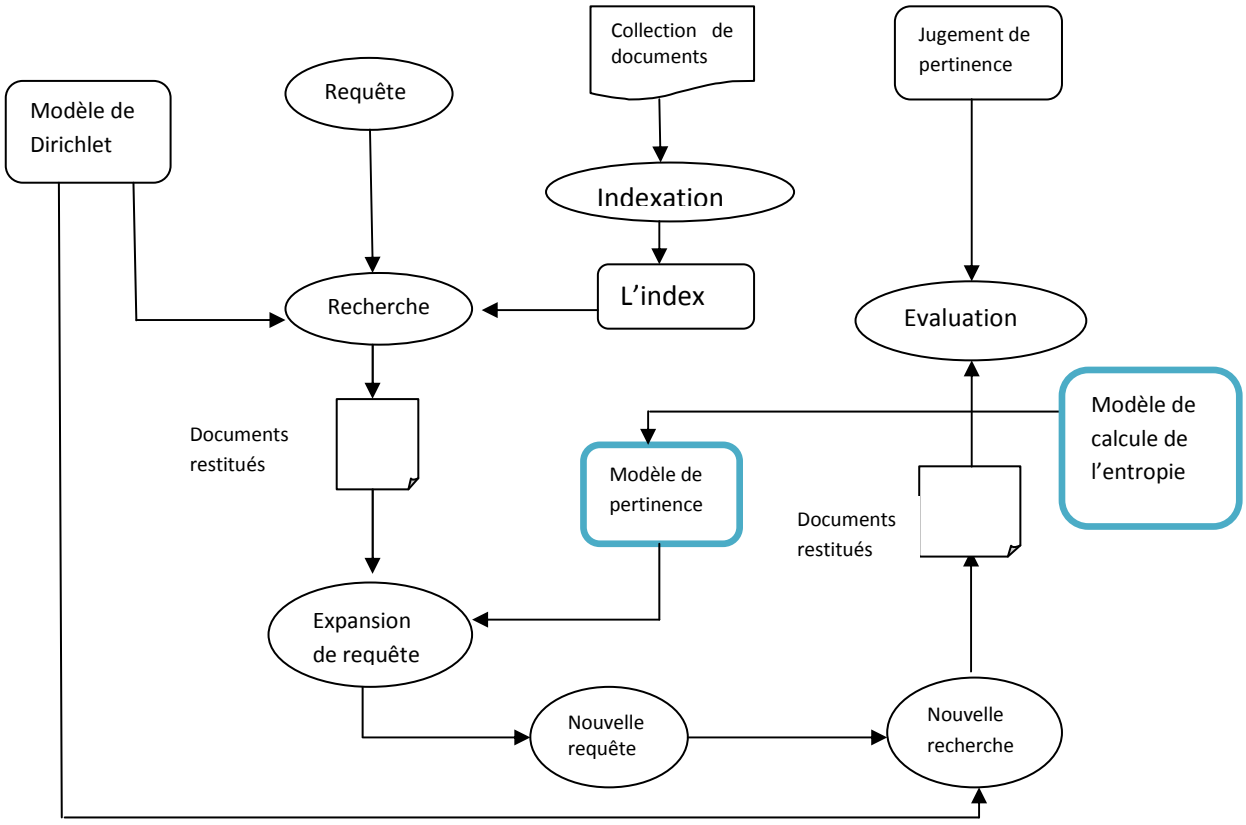

Figure III.1. Architecture générale de notre approche

2.2.Présentation de notre approche

L'utilisateur exprime son besoin en information par une requête qui sera exécutée par le système de recherche d'information mais cette requête là n'est pas toujours complète et claire pour retourner de meilleurs résultats, c'est pour cela que nous utilisons l'expansion de requête pour améliorer la requête initiale en ajoutant des termes générés, classés et choisis pour enfin reformuler cette requête.

Dans cette section nous allons présenter une nouvelle approche qui consiste à étendre le modèle de pertinence avec des formules en variant des paramètres précis afin de donner la meilleure pertinence possible.

2.2.1. La recherche simple : la recherche simple est la première étape du processus de notre approche qui consiste à assigner un score à chaque terme de la requête dans le document puis assigne des scores aux documents de la collection en se basant sur le modèle de **Dirichlet** à l'issu de cette étape nous aurons un résultat d'une première recherche qui est une liste de documents ordonnés suivant leur scores.

- **Modèle de Dirichlet :** Le modèle de Dirichlet est une méthode de lissage utilisée dans notre cas pour faire une première recherche de document. Le modèle de Dirichlet et présenté sous la formule suivante :

$$\boldsymbol{P(m_i|d) = \frac{tf(m_i,d) + \mu P_{ML}(m_i|C)}{|d| + \mu}}$$

Avec :

$$\boldsymbol{P_{ML}(m_i|d) = \frac{tf(m_i,d)}{|d|}}$$

Où :

$\boldsymbol{|d|}$ est la taille du document, représente le nombre d'occurrence de mots dans le document ;

$\boldsymbol{tf(m_i,d)}$ est la fréquence du mot $\boldsymbol{m_i}$ dans le document **d ;**

$\boldsymbol{\mu}$ est un paramètre appelé pseudo fréquence.

2.2.2. L'Expansion de requête

Notre travail dans cette étape consiste à étendre le modèle de pertinence avec la caractéristique de la taille et la caractéristique de l'entropie.

2.2.2.1. Le modèle de pertinence :

Le modèle de pertinence retourne une liste des tops documents à partir d'une collection en se basant sur l'estimation de la probabilité $P(t|\theta_{REL})$, une description détaillée présentée dans le chapitre précédent (section 5.3.1.2).

Il est formalisé comme suit :

$$P(t|\theta_{REL}) = \sum_{d \in R} p(d)\, p(t|d) \prod_{i=1}^{k} p(q_i|d)$$

Avec :

k : Le nombre de termes q_i de la requête.

$p(d)$: La probabilité a priori de choisir un document d des tops documents pertinents.

$p(t|d)$: La probabilité d'observer le terme t dans le document d.

$p(q_i|d)$: La probabilité d'observer les termes q_i de la requête dans le document d.

R : Le nombre de tops documents pertinents $\theta_{REL.}$

2.2.2.2.Utilisation de l'entropie : [66]

a) Définition :

L'entropie d'une manière générale permet de quantifier le contenu moyen en information d'un ensemble de message ou de document. L'entropie se calcule comme suit : si on considère *k* évènements disjoints de probabilités respectives $p_1, p_2, \dots, p_k \; avec \; p_1 + p_2 +, \dots, p_k = 1$, alors la quantité d'information correspondant à cette distribution de probabilité est $p_1 \log\left(\frac{1}{p_1}\right) +, \dots, + p_k \log\left(\frac{1}{p_k}\right)$. Cette quantité s'appelle l'entropie de la distribution de probabilité. L'entropie permet donc de mesurer la quantité d'information moyenne d'un ensemble d'évènements (en particulier de messages) et de mesurer son incertitude. On la note H

$$H(n) = -\sum_{i=1}^{k} p_i * \log_2 p_i$$

Dans notre cas nous utilisons l'entropie dans le domaine de la recherche d'information, particulièrement pour calculer la pertinence a priori des documents. Nous nous sommes basé sur l'intuition suivante :

b) L'intuition :

Un document entropique est un document qui comprend beaucoup d'informations, moins de redondance, donc il est a priori pertinent.

Nous formalisons cette caractéristique comme suit :

Entropie $H(n) = -\sum_{i=1}^{k} P(t_i|d) * \log P(t_i|d)$

c) Formalisation :

Nous allons étendre le modèle de pertinence de base avec la formule suivante :

$$p(d) = Entropie(d) = -\sum_{i=1}^{n} p(t_i|d) * \log_2\big(p(t_i|d)\big)$$

$$= -\sum_{i=1}^{n} \frac{tf_i}{|d|} * \log_2\left(\frac{tf_i}{|d|}\right) \qquad \text{F (3)}$$

Puis avec la deuxième formule suivante en introduisant le log pour diminuer l'écart entre les valeurs de l'entropie :

$$p(d) = \log\left(-\sum_{i=1}^{n} \frac{tf_i}{|d|} * \log_2\left(\frac{tf_i}{|d|}\right)\right) \qquad \text{F (4)}$$

2.2.2.3.Utilisation de la taille de document :

a) L'intuition :

L'intuition par rapport à la taille d'un document consiste à dire qu'un document plus long (c'est-à-dire plus de mots dans le document) comprend plus d'information et donc plus pertinent.

b) Formalisation :

La première formule proposée est formalisée ainsi :

$$P(d) = \frac{taille(d)}{taille(R)} \qquad \text{F (1)}$$

La deuxième formule fait intervenir le log de la taille d'un document pour diminuer l'écart entre les valeurs de la taille de document, elle est formalisée ainsi :

$$P(d) = log\left(\frac{taille(d)}{taille(R)}\right) \qquad \text{F (2)}$$

2.3.Exemple illustratif :

Dans ce qui suit nous allons montrer comment que les documents change de rands avec le modèle de pertinence et le modèle de pertinence étendu avec la formule F3.

Id_doc	Rang_doc		
	Recherche simple	Expansion avec le modèle de pertinence	Expansion avec l'approche implémentée
AP880412-0268	18	1	1
AP880627-0045	15	12	9
AP881219-0271	129	42	21
AP880721-0290	155	30	12

III.1. Classement des documents avant et après expansion

Le document **AP880412-0268** a un rand égal à **18** dans la recherche simple puis il a passé au rang **1** dans le modèle de pertinence et il a gardé ce rand même dans le modèle de pertinence étendu avec la formule 3.

Le document **AP880627-0045** a un rand de **15** dans la recherche simple et il est passé au 12eme dans le modèle de pertinence et il a gagné encore **3** rands dans la dernière recherche.

Le document **AP881219-0271** est dans le rand numéro **129** avec la recherche simple mais il est passé au rand numéro **42** dans le modèle de pertinence et au numéro **21** avec la dernière recherche.

Le document **AP880721-0290** est dans le rand numéro de **155** avec la recherche simple est il occupe le rand **30** avec le modèle de pertinence et enfin il sera dans le rand **12** avec notre approche.

3. L'environnement technique

Dans ce qui suit nous allons présenter notre environnement technique et connaitre les différent outils utilisés commençons par présenter la plate forme Terrier sous laquelle nous allons implémenter notre approche, puis la JAVA comme langage de programmation et Netbeans.

3.1.Présentation de la plate forme terrier

TERRIER**, TERabye ReteIEveR :** est un systeme de recherche robuste et efficace, développé par le département informatique de l'université Glasgow de Scotland. Utilisé avec succès dans :

- La recherche ad-hoc
- La recherche sur le web
- La recherche multilingue

Terrier offre une plate forme idéale destinée à l'indexation de volumes importants de documents : jusqu'à 25 million de documents. C'est un logiciel open Source écrit en java

Comme tout système de recherche d'information, terrier permet :

- L'indexation classique : extraction des mots clés des documents appartenant à une collection et les stocker dans un index
- Recherche des documents pertinents pour répondre aux requêtes formulées par l'utilisateur
- Evaluation des résultats de la recherche

- **Architecture de terrier**

La figure suivante illustre l'architecture de Terrier.

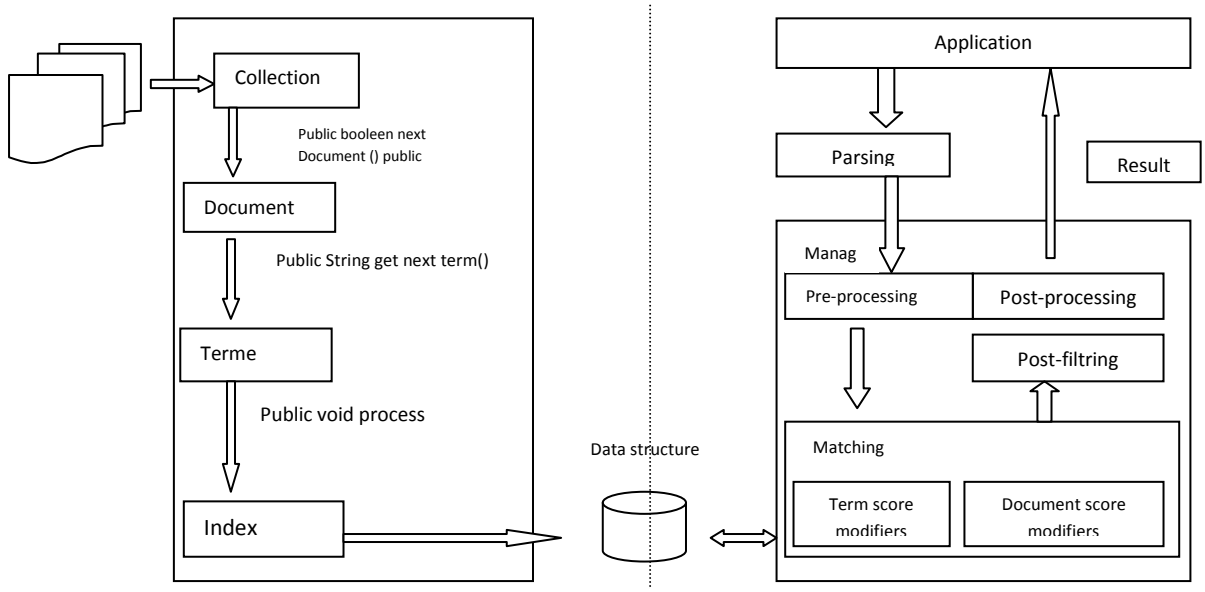

Figure III.2. Architecture de Terrier

A. API d'indexation :

La figure ci-dessous donne une vue d'ensemble d'interaction des composants principaux impliqués dans le processus d'indexation.

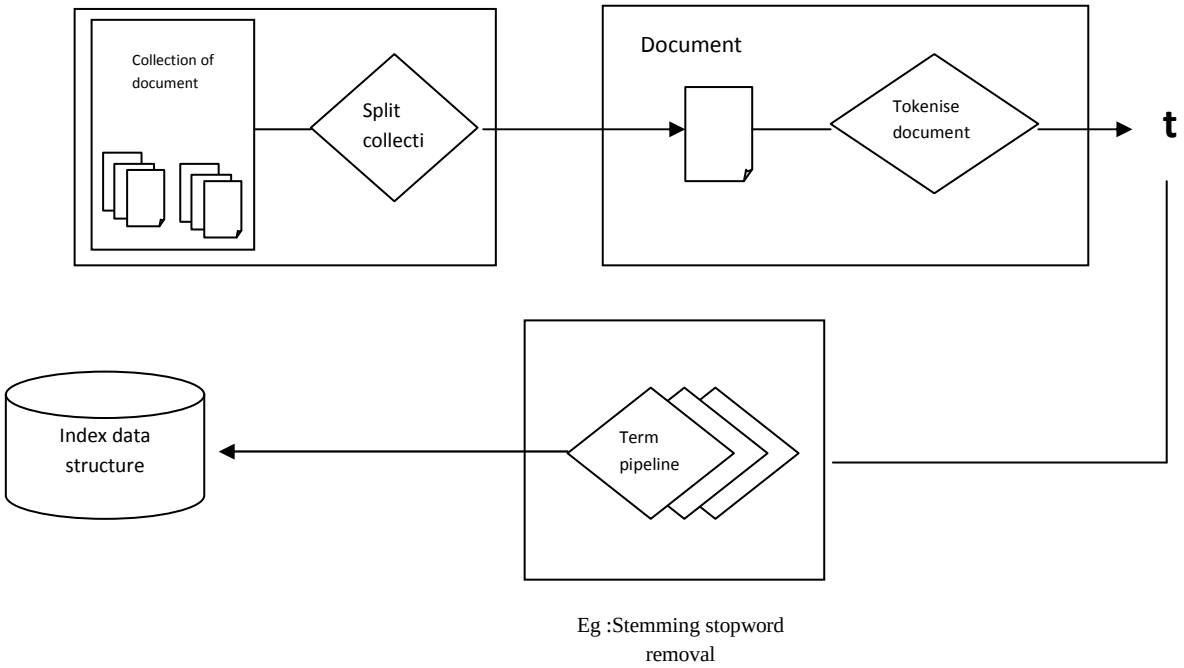

Figure III.3. Le processus d'indexation dans Terrier

- **Collection** : Interface dans le package …\src\uk\ac\gla\terrier\indexing, permet de splitter une collection(ou corpus)
- **Document** : Interface dans le package …\src\uk\ac\gla\terrier\indexing, permet de parcourir les documents et extraire les termes(en utilisant Tokeniser).
- **Term Pipeline** (interface) : effectue les traitement des termes extraits :
  - Eliminer les mots vides(Stopwords)
  - Lemmatisation des termes selon la langue.

B. Les structure de l'indexe.

Après l'indexation les termes sont stockés en structure de données suivantes :

- **Lexicon :** contient les informations sur chaque terme de la collection (Terme, Id terme, nombre docs qui contiennent le terme, fréquence de terme dans la collection, Offset dans le fichier inverse).
- **Inverted Index** : Fichier inverse (Id Terme, Id document, Fréquence terme dans le document, #filds).
- **Direct index :** Index (Id Terme, Id document, Fréquence terme dans le document, #filds).
- **Document Index :** (Id Terme, Fréquence terme, #filds).

C. API de recherche :

L'API de recherche contient les classes suivantes :

- **Query** : Classe abstraite qui représente la requête.

Terrier supporte trois modèles de requête :

Single Terme Query : Désigne la requête qui contient un seul terme
Multi Terme Query : Désigne la requête qui contient plusieurs termes
Field Query : Terme qualifié par un champ (EXP : dans le titre du document).

- **Manager :** chargé de la gestion de la recherche, il contient les étapes suivantes :

Pre-processing : Appliquer l'élimination des mots vides et troncature.
Matching : Déterminer les documents qui répondent à la requête en initialisant :

- *WeightingModels* : Assigner un score pour chaque terme de la requête dans le document(Pondération), Plusieurs modèles de pondération sont implémentés : TF_IDF, BM25,…etc.
- *DocumentScoreModifier* : Il permet de modifier le score d'un document en fonction du langage de la requête.

Post-filtring : Filtrer les documents pertinents selon un critère bien défini (exemple : type de document).

Post-processing : Reclasser les documents pertinent s'il y'a une expansion de la recherche.

- **Set-Results** : Contient la liste des documents retournés classés selon leur degré de pertinence.

3.2. Le langage java

Java est un langage de programmation moderne développé par Sun Microsystems (aujourd'hui racheté par Oracle). Il ne faut surtout pas le confondre avec javaScript (Langage de Script utilisé principalement sur les sites web), car java n'à rien avoir. Une de ses plus grande force est son excellente portabilité : une fois votre programme crée, il fonctionnera automatiquement sous Windows, Mac, Linux etc. On peut faire de nombreuse sorte de programmes avec java :

- Des applications, sous forme de fenêtres ou de console ;
- Des applets, qui sont des programmes java incorporés à des pages web ;
- Des applications pour appareils mobiles, ave J2ME ;
- et bien d'autres J2EE, JMF, J3D pour la 3D…etc.

Java se voit attribuer plusieurs qualités dont voici quelques unes :

- **Java est orienté objet** : java se veut un pur langage de P.O.O, c'est-à-dire qu'un programme s'y trouvera formé d'une classe ou de la réunion de plusieurs classes et il instanciera des objets.
- **Java est simple :** La syntaxe de java est, en grande partie, tirée de celle de C++, avec l'avantage de l'élimination des mécanismes complexes, comme la gestion des pointeurs et de la mémoire, rendant ainsi les programmes java plus faciles à écrire et à compiler avec le moins d'erreurs possible.

- **Java est distribué** : java implémente les protocoles réseau standards, ce qui permet de développer des applications client/serveur en architecture distribuée, afin d'invoquer des traitements et/ou de récupérer des données sur des machine distantes.
- **Java est interprétée :** Un programme java n'est pas exécuté, il est interprété par la machine virtuelle ou JVM (Java Virtual Machine), ce qui le rend un plus lent, mais cela apporte des avantages, notamment celui de ne pas être obligé de recompiler un programme java d'un système à une autre car il suffit, pour chacun des systèmes, de posséder sa propre machine virtuelle java.
- **Java est robuste :** Java est un langage fortement typé et très strict. Par exemple la déclaration des variables doit obligatoirement être explicite en java. Le code est vérifié (Syntaxe, type) à la compilation et également au moment de l'exécution, ce qui permet de réduire les bugs et les problèmes d'incompatibilité de version.
- **Java est sécurisé :** Au moment de l'exécution d'un programme java, le JRE utilise un processus nommé le classe loader qui s'occupe du chargement du *byte code* (ou langage binaire intermédiaire) contenu dans les classes java, le byte code est ensuite analysé afin de contrôler qu'il n'a pas fait de création ou de manipulation de pointeurs en mémoire et également qu'il n'y a pas de violation d'accès.
- **Java est portable :** cette caractéristique est l'une de celles qui ont contribué à sa grande réputation parmi les communautés d'internet et ceci grâce à son indépendance de toute plate forme d'exécution car un programme java peut tourner sur n'importe quelle machine possédant une JVM.

3.3.Présentation de NetBeans

NetBeans est à l'origine un EDI (Environnement de Développement Intégré) Java. NetBeans fut développé à l'origine par une équipe d'étudiants à Prague, racheté ensuite par Sun Microsystems. En 2002, Sun a décidé de rendre NetBeans open-source. Mais NetBeans n'est pas uniquement un EDI java, c'est également une plateforme. Il vous est possible de créer votre propre application awt ou swing, basé sur la plateforme NetBeans. Pour celles et ceux d'entre vous qui vienne du monde Eclipse, cela correspond à Eclipse RCP.

Sa conception est complètement modulaire : tout est module, même la plateforme. Ce qui fait de NetBeans une boite à outils facilement améliorable ou modifiable. La licence de NetBeans permet de l'utiliser gratuitement à des fin commerciales ou non. Les modules que vous pourriez écrire peuvent être open-sources comme ils peuvent être closed-source, ils peuvent être gratuits, comme ils peuvent être payants.

Il présente une interface conviviale **GUI** (Graohical User Interface) qui nous permet d'éditer, compiler et exécuter un programme écrit en langage java.

4. Résultats et expérimentations :

4.1.Collection de teste utilisée :

Nous avons utilisé dans notre approche (pour tester le modèle de pertinence étendu) une collection de document TREC AP88, et nous avons utilisé aussi 50 requêtes.

4.2.Evaluation et résultats :

Pour évaluer notre approche, nous allons faire une comparaison entre les résultats obtenus avec la recherche simple (modèle Dirichlet) et les résultats de notre approche, et cela en appliquant les mêmes paramètres de recherche (nombre de termes et de documents d'expansion). Pour évaluer nos résultats nous allons utiliser La MAP (moyenne average precision).

4.2.1. Résultats obtenus avec la recherche simple :

La recherche simple est basée sur le modèle de Dirichlet et pour obtenir les meilleurs résultats possibles pour les utiliser dans l'expansion de requête, nous avons varié le Paramètre μ de 100 à 34000 avec un pas de 100 pour le fixer par la suite dans l'étape suivante (l'expansion)

Ce tableau résume les résultats obtenus.

μ	MAP	μ	MAP
100	0.0165	13000	0.0714
200	0.0251	14000	0.0721
300	0.0304	15000	0.0723
400	0.0355	16000	0.0725
500	0.0399	17000	0.0728
600	0.0432	18000	0.0731
700	0.0457	19000	0.0734
800	0.0474	20000	0.0738
900	0.0485	20100	0.0738
1000	0.0497	20200	0.0738
2000	0.0586	**20300**	**0.0738**
3000	0.0619	20400	0.0737
4000	0.0648	20500	0.0736
5000	0.0666	20600	0.0735
6000	0.0680	20700	0.0734
7000	0.0685	20800	0.0733
8000	0.0691	20900	0.0733
9000	0.0682	30000	0.0720
10000	0.0686	30100	0.0720
11000	0.0686	30200	0.0720
12000	0.0701	30300	0.0720

III.2. Tableau des résultats avec la recherche simple

Le tableau ci-dessus montre que la meilleure précision obtenue dans la recherche simple en variant le paramètre μ du modèle Dirichlet est **0.0738** avec $\boldsymbol{\mu = 20300}$

4.2.2. Résultats obtenus avec le modèle de pertinence (de base) :

Dans cette étape nous allons présenter les résultats obtenus avec le modèle de pertinence, nous avons testé en faisant varier les deux paramètres essentiels de ce modèle qui sont le nombre de documents allant de **1** à **30** avec un pas de **5** et en variant aussi le nombre de termes allant de **1** à **30** également avec un pas de **5** .
Sachant que nous avons prit le paramètre $\boldsymbol{\mu = 20300}$ du modèle de Dirichlet.

Les résultats obtenus sont résumés dans le tableau suivant :

Nombre de documents	Nombre de termes	MAP	Nombre de document	Nombre de termes	MAP
1	1	0.1007	15	1	0,0952
	5	0.1007		5	0,0942
	10	0.1007		10	0,0942
	15	0.1007		15	0,0942
	20	0.1007		20	0,1121
	25	0.1007		25	0,1119
	30	0.1007		30	0,113
5	1	0,1084	20	1	0,1026
	5	0,1084		5	0,1033
	10	0,1084		10	0,1033
	15	0,1084		15	0,1121
	20	0,1162		20	0,1033
	25	0,1136		25	0,1226
	30	0,1137		30	0,1244
10	1	0,1137	25	1	0,1048
	5	0,1137		5	0,1042
	10	0,1137		10	0,1138
	15	0,1079		15	0,1199
	20	0,1103		20	0,1221
	25	0,1114		25	0,1222
	30	0,1114		30	0,1223
			30	1	0,0972
				5	0,0974
				10	0,0974
				15	0,0974
				20	0,0974
				25	0,0974
				30	0,0974

III.3. Résultats obtenus avec le modèle de pertinence.

Le tableau ci-dessus résume les résultats obtenus dans l'expansion de requête avec le modèle de pertinence de base, et nous constatons que la meilleure précision est **0.1244,** elle est

obtenue avec le nombre de document égal à **20** et le nombre de termes est égal à **30** et donc maintenant nous allons tester notre approche avec ces deux paramètres et le paramètre $\boldsymbol{\mu = 20300}$ obtenu avec la première recherche.

4.2.3. Résultats obtenus avec notre approche

Dans le tableau suivant nous allons présenter la précision et le taux d'amélioration de chaque formule ajoutée au modèle de pertinence de base afin de montrer la formule qui apporte des améliorations pour l'expansion de requête.

Formule	MAP	Taux d'amélioration
F1	0.1331	6,99 %
F2	0.1241	-0,24 %
F3	**0.1363**	**9,57 %**
F4	0,0737	-40,76 %

Tableau III.4. Résultats obtenus avec notre approche

D'après le tableau ci-dessus nous constatons que le modèle de pertinence étendu avec la formule 3 a apporté une importante amélioration par rapport au modèle de pertinence de base avec une MAP de **0.1363** et un taux d'amélioration de **9,57 %.**

Nous résumons les différentes précisions du modèle de pertinence étendu avec les 4 formules et le modèle de pertinence de base dans la figure suivante :

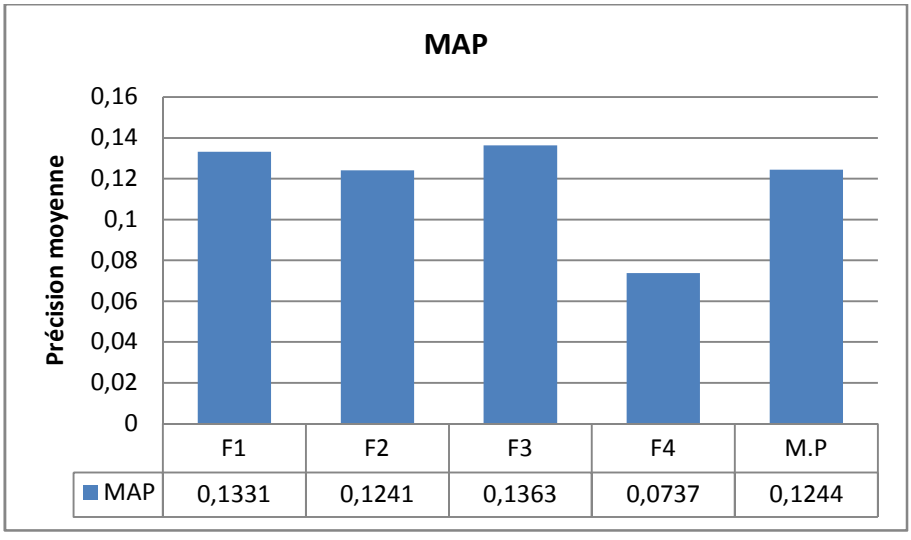

Figure III.4. Comparaison du modèle de pertinence étendu avec les formules proposées avec le modèle de pertinence

Cette figue montre que le modèle de pertinence étendu avec la formule F3 a apporté la meilleure précision.

4.2.4. Evaluation requête par requête :

Dans les tableaux qui suivent nous allons montrer les résultats précédents mais cette fois requête par requête.

Requête	MAP (modèle de Dirichlet)	MAP (modèle de pertinence)	Taux d'amélioration %	Requête	MAP (modèle de Dirichlet)	MAP (modèle de pertinence)	Taux d'amélioration %
51	0,3836	0,7007	82,66	76	0,0008	0,0001	-87,50
52	0,0976	0,1452	48,77	77	0,1439	0,4462	210,08
53	0,2077	0,5711	174,96	78	0,3069	0,5268	71,65
54	0,1028	0,1507	46,60	79	0	0	0
55	0,0044	0,0691	1470,45	80	0,0106	0,0113	6,60
56	0,0557	0,0124	-77,74	81	0,067	0,1658	147,46
57	0,3224	0,5528	71,46	82	0,1058	0,1635	54,54
58	0,0141	0,0053	-62,41	83	0,0024	0,0006	-75,00
59	0	0	0	84	0,0038	0,0022	-42,11
60	0,0009	0,0025	177,78	85	0,0185	0,0065	-64,86
61	0,0334	0,0096	-71,26	86	0,0029	0,0014	-51,72
62	0,0318	0,1874	489,31	87	0,0009	0,0005	-44,44
63	0,0486	0,0267	-45,06	88	0,0274	0,0241	-12,04
64	0,0038	0,0003	-92,11	89	0,0062	0,0086	38,71
65	0	0	0	90	0,1456	0,0744	-48,90
66	0,0006	0,0006	0	91	0	0	0
67	0	0	0	92	0	0	0
68	0,0168	0,0282	67,86	93	0,0008	0,0005	-37,50
69	0,0035	0,0011	-68,57	94	0,001	0,0003	-70,00
70	0,2496	0,8035	221,92	95	0	0	0
71	0,044	0,0212	-51,82	96	0,0132	0,0181	37,12
72	0,0072	0,0095	31,94	97	0,566	0,8238	45,55
73	0,0009	0,0011	22,22	98	0,5107	0,4663	-8,69
74	0,0002	0,0002	0	99	0,0216	0,0077	-64,35
75	0,0277	0,0487	75,81	100	0,001	0,0005	-50,00

Tableau III.5.Résultats obtenus avec l'analyse requête par requête avant et après l'expansion avec le modèle de pertinence

Le tableau ci-dessus montre le taux d'amélioration du modèle de pertinence par rapport au modèle de Dirichlet où nous avons obtenu les résultats suivants:

- 21 requêtes améliorées
- 20 requêtes dégradées
- 9 requêtes nulles

Requête	MAP (modèle de Dirichlet)	MAP (F3)	Taux d'amélioration %	Requête	MAP (modèle de Dirichlet)	MAP (F3)	Taux d'amélioration %
51	0,3836	0.7341	91,37	76	0,0008	0.0005	-37,50
52	0,0976	0.2030	107,99	77	0,1439	0.5123	256,01
53	0,2077	0.5866	182,43	78	0,3069	0.7486	143,92
54	0,1028	0.1744	69,65	79	0	0.0000	0,00
55	0,0044	0.0471	970,45	80	0,0106	0.0000	-100,00
56	0,0557	0.0164	-70,56	81	0,067	0.1839	174,48
57	0,3224	0.6244	93,67	82	0,1058	0.2255	113,14
58	0,0141	0.0055	-60,99	83	0,0024	0.0004	-83,33
59	0	0.0000	0,00	84	0,0038	0.0015	-60,53
60	0,0009	0.0037	311,11	85	0,0185	0.0080	-56,76
61	0,0334	0.0119	-64,37	86	0,0029	0.0017	-41,38
62	0,0318	0.1716	439,62	87	0,0009	0.0004	-55,56
63	0,0486	0.0320	-34,16	88	0,0274	0.1126	310,95
64	0,0038	0.0004	-89,47	89	0,0062	0.0069	11,29
65	0	0.0000	0	90	0,1456	0.0896	-38,46
66	0,0006	0.0000	-100,00	91	0	0.0000	0
67	0	0.0000	0	92	0	0.0000	0
68	0,0168	0.0323	92,26	93	0,0008	0.0005	-37,50
69	0,0035	0.0019	-45,71	94	0,001	0.0004	-60,00
70	0,2496	0.8051	222,56	95	0	0.0000	0
71	0,044	0.0290	-34,09	96	0,0132	0.0000	-100,00
72	0,0072	0.0107	48,61	97	0,566	0.7229	27,72
73	0,0009	0.0011	22,22	98	0,5107	0.4661	-8,73
74	0,0002	0.0001	-50,00	99	0,0216	0.0087	-59,72
75	0,0277	0.0530	91,34	100	0,001	0.0006	-40,00

Tableau III.6.Résultats obtenus avec l'analyse requête par requête avant et après l'expansion avec le modèle de pertinence étendu avec (F3)

Le tableau ci-dessus montre le taux d'amélioration du modèle de pertinence étendu avec la formule 3 par rapport au modèle de Dirichlet où nous avons obtenu les résultats suivants:

- 20 requêtes améliorées
- 23 requêtes dégradées
- 07 requêtes nulles

Requête	MAP (modèle de pertinence)	MAP (F3)	Taux d'amélioration %	Requête	MAP (modèle de Pertinence)	MAP (F3)	Taux d'amélioration %
51	0,7007	0.7341	4,77	76	0,0008	0.0005	-37,50
52	0,1452	0.2030	39,81	77	0,1439	0.5123	256,01
53	0,5711	0.5866	2,71	78	0,3069	0.7486	143,92
54	0,1507	0.1744	15,73	79	0,00	0.0000	0,00
55	0,0691	0.0471	-31,84	80	0,0106	0.0000	-100,00
56	0,0124	0.0164	32,26	81	0,067	0.1839	174,48
57	0,5528	0.6244	12,95	82	0,1058	0.2255	113,14
58	0,0053	0.0055	3,77	83	0,0024	0.0004	-83,33
59	0,00	0.0000	0,00	84	0,0038	0.0015	-60,53
60	0,0025	0.0037	48,00	85	0,0185	0.0080	-56,76
61	0,0096	0.0119	23,96	86	0,0029	0.0017	-41,38
62	0,1874	0.1716	-8,43	87	0,0009	0.0004	-55,56
63	0,0267	0.0320	19,85	88	0,0274	0.1126	310,95
64	0,0003	0.0004	33,33	89	0,0062	0.0069	11,29
65	0,00	0.0000	0,00	90	0,1456	0.0896	-38,46
66	0,0006	0.0000	-100,00	91	0 ,00	0.0000	0,00
67	0 ,00	0.0000	0,00	92	0,00	0.0000	0,00
68	0,0282	0.0323	14,54	93	0,0008	0.0005	-37,50
69	0,0011	0.0019	72,73	94	0,001	0.0004	-60,00
70	0,8035	0.8051	0,20	95	0,00	0.0000	0,00
71	0,0212	0.0290	36,79	96	0,0132	0.0000	-100,00
72	0,0095	0.0107	12,63	97	0,566	0.7229	27,72
73	0,0011	0.0011	0,00	98	0,5107	0.4661	-8,73
74	0,0002	0.0001	-50,00	99	0,0216	0.0087	-59,72
75	0,0487	0.0530	8,83	100	0,001	0.0006	-40,00

Tableau III.7.Comparaison de l'analyse requête par requête obtenue dans l'expansion avec le modèle de pertinence étendu avec F3 et c'elle obtenue avec le modèle de pertinence

Le tableau ci-dessus montre le taux d'amélioration du modèle de pertinence étendu avec la formule 3 par rapport au modèle de pertinence de base où nous avons obtenu les résultats suivants:

- 25 requêtes améliorées
- 18 requêtes dégradées
- 07 requêtes nulles

4.2.5. Analyse des résultats obtenus précédemment en se basant sur le type de requête :

Nous proposons dans ce qui suit l'amélioration des résultats selon le type de requête, nous avons classé les requêtes en trois types :

- Ambigües si la valeur de MAP<0.05 ;
- Moyennes si 0.05<MAP<0.2 ;
- Claires si MAP<0.2.

Requêtes	Recherche simple	Modèle de pertinence	Taux d'amélioration	Modèle étendu avec F3	Taux F3 par rapport à la recherche simple	Taux F3 par rapport au modèle de pertinence
51	Claire	+	**85.71%**	+,++	**85.71%**	**85.71%**
53	Claire	+		+,++		
57	Claire	+		+,++		
70	Claire	+		+,++		
78	Claire	+		+,++		
97	Claire	+		+,++		
98	Claire					
52	Moyenne	+	**71.43%**	+,++	**71.43%**	**85.71%**
54	Moyenne	+		+,++		
56	Moyenne			++		
77	Moyenne	+		+,++		
81	Moyenne	+		+,++		
82	Moyenne	+		+,++		
90	Moyenne					
55	Ambigüe	+	**27.78%**	+	**25%**	**33.33%**
58	Ambigüe			++		
59	Ambigüe					
60	Ambigüe	+		+,++		
61	Ambigüe			++		
62	Ambigüe	+		+		
63	Ambigüe			++		
64	Ambigüe			++		
65	Ambigüe					
66	Ambigüe					
67	Ambigüe					
68	Ambigüe	+		+,++		
69	Ambigüe			++		
71	Ambigüe			++		
72	Ambigüe	+		+,++		
73	Ambigüe	+		+		
74	Ambigüe					
75	Ambigüe	+		+,++		
76	Ambigüe					
79	Ambigüe					

80	Ambigüe	+				
83	Ambigüe					
84	Ambigüe					
85	Ambigüe					
86	Ambigüe					
87	Ambigüe					
88	Ambigüe			+,++		
89	Ambigüe	+		+,++		
91	Ambigüe					
92	Ambigüe					
93	Ambigüe					
94	Ambigüe					
95	Ambigüe					
96	Ambigüe	+				
99	Ambigüe					
100	Ambigüe					

Tableau III.8. Les résultats obtenus en utilisant l'expansion avec le modèle de pertinence étendu avec F3 et ceux obtenus en utilisant l'expansion avec le modèle de pertinence en se basant sur le type des requêtes

Dans la recherche simple, les résultats on révélé :

- 7 requêtes claires
- 7 requêtes moyennes
- 36 requêtes ambigües

Dans l'expansion avec le modèle de pertinence, les résultats on révélé :

- 6 requêtes améliorées sur les 7 requêtes claires de la recherche simple avec un taux d'amélioration de **85.71%.**
- 5 requêtes améliorées sur les 7 requêtes moyennes de la recherche simple avec un taux d'amélioration de **71.43%.**
- 10 requêtes améliorées sur les 36 requêtes ambigües de la recherche simple avec un taux d'amélioration de **27.78%.**

Dans l'expansion avec le modèle de pertinence avec F3, les résultats on révélé :

- 6 requêtes améliorées sur les 7 requêtes claires de la recherche simple avec un taux d'amélioration de **85.71%**et 6 requêtes améliorées sur les 7 requêtes claires du modèle de pertinence avec un taux d'amélioration de **85.71%.**
- 5 requêtes améliorées sur les 7 requêtes moyennes de la recherche simple avec un taux d'amélioration de **71.43%** et 6 requêtes améliorées sur les 7 requêtes moyennes du modèle de pertinence avec un taux d'amélioration de **85.71%.**
- 9 requêtes améliorées sur les 36 requêtes ambigües de la recherche simple avec un taux d'amélioration de **25%** et 12 requêtes améliorées sur les 36 requêtes ambigües du modèle de pertinence avec un taux d'amélioration de **33.33%.**

5. Conclusion :

Dans ce chapitre nous avons exposé notre approche qui consiste à étendre le modèle de pertinence de base avec les deux caractéristiques : la taille de document et l'entropie, puis nous avons testé cette approche et obtenus des résultats que nous avons présentés sous forme de tableaux et d'histogrammes pour les comparer aux résultats obtenus avec le modèle de pertinence de base.

Le teste de cette approche en introduisant l'entropie nous a apporté de meilleurs résultats en terme de précision, ainsi que la pertinence des documents retournés à l'utilisateur.

Conclusion générale

Notre projet consistait à retourner à l’utilisateur les bons documents contenants l’information dont il a besoin comme réponse à une requête.

En effet, l’expansion de requête permet que cela arrive en faisant la sélection, le classement, et le choix des termes d’expansion d’une requête initiale soumise par l’utilisateur jugée courte ou mal exprimée.

Dans notre approche implémentée, nous avons proposé d’améliorer le processus de sélection des termes d’expansion en se basant sur le modèle de pertinence. Nous avons proposé des formules que nous avons évaluées et ajoutées au modèle de pertinence pour justement arriver à un processus efficace pour la recherche d’information.

D’après les résultats obtenus nous sommes dans la bonne voix avec la collection de teste TREC APP88, et parmi les perspectives envisagées pour notre travail est l’expérimentation de notre approche sur d’autre collections plus volumineuses afin de valider les résultats obtenus.

Références bibliographiques

Bibliographie

[1] Thèse de Monsieur Hammache arezki sur la recherche d'information

[2]Chirita, P.-A., Firan, C. S., and Nejdl, W. 2007. Personalized query expansion for the web. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Amsterdam, The Netherlands, 7–14.

(3) Diaz, F. and Metzler, D. 2006. Improving the estimation of relevance models using large external corpora. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Seattle, Washington, USA, 154–161.

[5] E; Voorhees, (using wordnet to disambiguate word senses for text retrieval), processing of the annual conference on research and development in information retrieval, SIGIR 93, Pittsburgh, PA, 1993.

[6]Diaz, F. and Metzler, D. 2006. Improving the estimation of relevance models using large external corpora. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Seattle, Washington, USA, 154–161.

[7] Krovetz, R. and Croft, W. B. 1992. Lexical ambiguity and information retrieval. ACM Transactions on Information Systems (TOIS) 10, 2, 115–141.

[8] Crouch, C. and Yang, B. 1992. Experiments in automatic statistical thesaurus construction. In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Copenhagen, Denmark, 77–88.

[9] Kraft, R. and Zien, J. 2004. Mining anchor text for query refinement. In Proceedings of the 13th international conference on World Wide Web. ACM Press, New York, NY, USA, 666–674.

[10] Macdonald, C. and Ounis, I. 2007. Expertise drift and query expansion in expert search. In Proceedings of the 16th Conference on Information and Knowledge Management (CIKM 2007). ACM Press, Lisboa, Portugal.

[11] Attar, R. and Fraenkel, A. S. 1977. Local Feedback in Full-Text Retrieval Systems. Journal of ACM 24, 3, 397–417.

[12] Harper, G. W. and van Rijsbergen, C. J. 1978. An Evaluation of Feedback in Document Retrieval Using Co-Occurrence Data. Journal of Documentation 34, 3, 189–216.

[13] Church, K. and Hanks, P. 1990. Word association norms, mutual information and lexicography. Computational Linguistics 16, 1, 22–29.
[14] Sch¨utze, H. and Pedersen, O. 1997. A Co-occurrence based Thesaurus and Two Applications to Information Retrieval. Information Processing and Management 33, 3, 307–318.

[15] Bai, J., Song, D., Bruza, P., Nie, J.-Y., and Cao, G. 2005. Query expansion using term relationships in language models for information retrieval. In Proceedings of the 14th ACM international conference on Information and knowledge management. ACM Press, Bremen, Germany, 688–695.

[16] Agrawal, R., Imielinski, T., and Swami, A. 1993. Mining association rules between sets of itemsin large databases. In ACM SIGMOD International Conference on Management of Data. ACMPress, Washington, D.C., USA, 207–216.

[17] Latiri, C. C., Yahia, S. B., Chevallet, J. P., and Jaoua, A. 2004. Query expansion using fuzzy association rules between terms. In Proceedings of JIM'2003. Metz , France.

[18] Song, M., Song, I.-Y., Hu, X., and Allen, R. B. 2007. Integration of association rules and ontologies for semantic query expansion. Data & Knowledge Engineering 63, 1, 63–75.

[19] Beeferman, D. and Berger, A. 2000. Agglomerative clustering of a search engine query log. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM Press, Boston, MA, USA, 407–416.

[20] Qiu, Y. and Frei , H.-P. 1993. Concept based query expansion. In Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Pittsburgh, Pennsylvania, USA, 160–169.

[21] Xu, J. and Croft, W. B. 1996. Query expansion using local and global document analysis. In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Zurich, Switzerland, 4–11.

[22] Cui, H., Wen, J.-R., Nie, J.-Y., and Ma, W.-Y. 2003. Query expansion by mining user logs. IEEE Transactions on Knowledge and Data Engineering 15, 4, 829–839.

[23] Collins-Thompson, K. and Callan, J. 2005. Query expansion using random walk models. In Proceedings of the 14th Conference on Information and Knowledge Management (CIKM 2005). ACM Press, Bremen, Germany, 704–711.

[24] Salton, G. and Buckley, C. 1990. Improving Retrieval Performance by Relevance Feedback. Journal of the American Society for Information Sciences 41, 4, 288–297.

[25] Harman, D. K. 1992. Relevance feedback and other query modification techniques. In Information Retrieval – Data Structures and Algorithms, W. B. Frakes and R. Baeza-Yates, Eds. pubprenticehall, Englewood Cliffs, N. J., USA, 241–263.
[26] Wong, W. S., Luk, R. W. P., Leong, H. V., Ho, K. S., and Lee, D. L. 2008. Re-examining the effects of adding relevance information in a relevance feedback environment. Information Processing and Management 44, 3, 1086–1116.

[27] Bernardini, A. and Carpineto, C. 2008. Fub at trec 2008 relevance feedback track: extending rocchio with distributional term analysis. In Proceedings of TREC-2008. National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA.

[28] Amati, G., Carpineto, C., and Romano, G. 2001. FUB at TREC-10 Web Track: A Probabilistic Framework for Topic Relevance Term Weighting. In Proceedings of the 10th Text Retrieval

[29] Zhai, C. and Lafferty, J. 2001a. Model-based feedback in the language modeling approach to information retrieval. In Proceedings of the tenth international conference on Information and knowledge management. ACM Press, Atlanta, Georgia, USA, 403–410.

[30] Buckley, C. and Harman, D. K. 2003. Reliable information access final workshop report. In Reliable Information Access Workshop (RIA), NRRC. NRRC, Bedford, Massachusetts, USA, 1–30.

[31] Billerbeck, B. and Zobel, J. 2004a. Questioning query expansion: an examination of behavior and parameters. In Proceedings of the 15th Australasian database conference - Volume 27. Australian Computer Society, Dunedin, New Zealand, 69–76.

[32] Cao, G., Gao, J., Nie, J.-Y., and Robertson, S. 2008. Selecting good expansion terms for pseudo-relevance feedback. In Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Singapore, 243–250.

[33] Carpineto, C., Romano, G., and Giannini, V. 2002. Improving retrieval feedback with multipleterm-ranking function combination. ACM Transactions on Information Systems (TOIS) 20, 3, 259–290.

[34] Collins-Thompson, K. and Callan, J. 2007. Estimation and use of uncertainty in pseudorelevance feedback. In Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Amsterdam, The Netherlands, 303–310.

[35] Cronen-Townsend, S. and Croft, W. B. 2002. Quantifying query ambiguity. In Proceedings of the 2nd international conference on Human Language Technology Research. ACM Press, San Diego, California, 104–109.

[36] Cao, G., Gao, J., Nie, J.-Y., and Robertson, S. 2008. Selecting good expansion terms for pseudo-relevance feedback. In Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Singapore, 243– 250.

[37] Rocchio, J. J. 1971. Relevance feedback in information retrieval. In The SMART Retrieval System, G. Salton, Ed. Prentice-Hall, Englewood Cliffs, N. J., USA, 313–323.

[38] Salton, G. and Buckley, C. 1990. Improving Retrieval Performance by Relevance Feedback. Journal of the American Society for Information Sciences 41, 4, 288–297.

[39] Amati, G. 2003. Probabilistic models for information retrieval based on divergence from randomness. Ph.D. thesis, Department of Computing Science, University of Glasgow, UK.

[40] Wong, W. S., Luk, R. W. P., Leong, H. V., Ho, K. S., and Lee, D. L. 2008. Re-examining the effects of adding relevance information in a relevance feedback environment. Information Processing and Management 44, 3, 1086–1116.

[41] Montague, M. and Aslam, J. 2001. Relevance score normalization for metasearch. In Proceedings of the 10th international conference on Information and knowledge management. ACM Press, Atlanta, Georgia, USA, 427–433.

[42] Lv, Y. and Zhai, C. 2009. Adaptive relevance feedback in information retrieval. In Proceedings of the 18th Conference on Information and Knowledge Management (CIKM 2009). ACM Press, Hong Kong, China, 255–264.

[43] Lavrenko, V. and Croft, W. B. 2001. Relevance based language models. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, New Orleans, Louisiana, USA, 120–127.

[44] Cao, G., Gao, J., Nie, J.-Y., and Bai, J. 2007. Extending query translation to cross-language query expansion with markov chain models. In Proceedings of the 16th Conference on Information and Knowledge Management (CIKM 2007). ACM Press, Lisboa, Portugal.

[45] Graupmann, J., Cai, J., and Schenkel, R. 2005. Automatic query refinement using mined semantic relations. In Proceedings of the International Workshop on Challenges in Web Information Retrieval and Integration (WIRI). IEEE Computer Society, Tokyo, Japan, 205213.

[46] Crouch, C. and Yang, B. 1992. Experiments in automatic statistical thesaurus construction. In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Copenhagen, Denmark, 77–88.

[47] Robertson, S. E. and Walker, S. 2000. Microsoft cambridge at trec-9: Filtering track. In Proceedings of the 9th Text REtrieval Conference (TREC-9), NIST Special Publication 500-249. National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, 361–368.

[48] Liu, S., Liu, F., Yu, C., and Meng, W. 2004. An effective approach to document retrieval via utilizing wordnet and recognizing phrases. In Proceedings of the 27th annual international.

[49] Sch¨utze, H. and Pedersen, O. 1997. A Co-occurrence based Thesaurus and Two Applications to Information Retrieval. Information Processing and Management 33, 3, 307 318.

[50] Milne, D. N., Witten, I. H., and Nichols, D. M. 2007. A knowledge-based search engine powered by wikipedia. In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management. ACM Press, Lisbon, Portugal, 445–454.

[51] Hu, J., Deng, W., and Guo, J. 2006. Improving retrieval performance by global analysis. In Proceedings of the 18th International Conference on Pattern Recognition. IEEE Computer Society, Hong Kong, China, 703–706.

[52] Gauch, S., Wang, J., and Rachakonda, S. M. 1999. A corpus analysis approach for automatic query expansion and its extension to multiple databases. ACM Transactions on Information Systems (TOIS) 17, 3, 250–269.

[53] Natsev, A., Haubold, A., Teˇsi´c, J., Xie, L., and Yan, R. 2007. Semantic concept-based query expansion and re-ranking for multimedia retrieval. In Proceedings of the 15th international conference on Multimedia. ACM Press, Augsburg, Germany, 991–1000.

[54] Lam-Adesina, A. M. and Jones, G. J. F. 2001. Applying summarization techniques for term selection in relevance feedback. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. New Orleans, Louisiana, USA, 1–9.

[55] Agichtein, E., Lawrence, S., and Gravano, L. 2004. Learning to find answers to questions on the Web. ACM Transactions on Internet Technology (TOIT) 4, 2, 1299–162.

[56] Harabagiu, S. and Lacatusu, F. 2004. Strategies for advanced question answering. In Proceedings of the HLT- NAACL'04 Workshop on Pragmatics of Question Answering. Boston, MA, USA, 1–9.

[57]Riezler, S., Vasserman, A., Tsochantaridis, I., Mittal, V., and Liu, Y. 2007. Statistical machine translation for query expansion in answer retrieval. In Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07). Association for Computational Linguistics, Prague, Czech Republic, 464–471.

[58]Harabagiu, S., Moldovan, D., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., Grju, R., Rus, V., and Morarescu, P. 2001. The role of lexico-semantic feedback in open-domain textual question-answering. In Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-01). Association for Computational Linguistics, Toulouse, France, 282–289.

[59] Singhal, A. and Pereira, F. 1999. Document expansion for speech retrieval. In Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval. ACM Press, Berkeley, California, USA, 34–41.

[60] Billerbeck, B. and Zobel, J. 2005. Document expansion versus query expansion for ad-hoc retrieval. In Proceedings of the 10th Australasian Document Computing Symposium. Australian Computer Society, Sydney, Australia, 34–41.

[61] Kherfi, M. L., Ziou, D., and Bernardi., A. 2004. Image Retrieval from the World Wide Web: Issues, Techniques, and Systems. ACM Comput. Surv. 36, 1, 35–67.

[62] Belkin, N. J. and Croft, W. B. 1992. Information filtering and information retrieval: two sides of the same coin? Communications of the ACM 35, 12, 29–38.

[63] Palleti, P., Karnick, H., and Mitra, P. 2007. Personalized web search using probabilistic query expansion. In Proceedings of the 2007 IEEE/WIC/ACM International Conference on Web Intelligence. IEEE Computer Society, Silicon Valley, CA, USA, 83–86.

[64] Zimmer, C., Tryfonopoulos, C., and Weikum, G. 2008. Exploiting correlated keywords to improve approximate information filtering. In Proceedings of the 31th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Singapore, Singapore, 323–330.

[65] Thèse de doctorat de Guillermo Valente G_ OMEZ CARPIO.

[66] l'entropie de shannon(wikipedia).

[67] Hamid TEBRI formalisation et spécialisation d'un système de filtrage incrémental d'information.

[68] Hauff, C., Hiemstra, D., and de Jong, F. 2008. A survey of pre-retrieval query performance predictors. In Proceedings of the 17th Conference on Information and Knowledge Management (CIKM 2008). ACM Press, Napa Valley, California, USA, 1419–1420.

[69] Vechtomova, O. and Karamuftuoglu, M. 2004. Elicitation and use of relevance feedback Information. Information Processing and Management 42, 1, 191–206.

[70] D. Abberley, D. Kirby, S. Renals, and T. Robinson. The THISL broadcast news retrieval system. In Proc. ESCA Workshop on Accessing Information In Spoken Audio, pages 19{24, Cambridge, 1999.

[71] Efthimis N. Efthimiadis. Query Expansion. Annual Review of Information Science and Technology, ARIST. 31 :121{187, 1996.

[72] Diaz, F. and Metzler, D. 2006. Improving the estimation of relevance models using large external corpora. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM Press, Seattle, Washington, USA, 154–161.

[73] Lin, J. and Murray, G. C. 2005. Assessing the term independence assumption in blind relevance feedback. In Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval. ACM Press, Salvador, Brazil, 635–636.