République Algérienne Démocratique Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université d'Ibn Khaldoun – Tiaret

Faculté des Mathématiques et de l'Informatique

Département d' Informatique

Thème

Optimisation de la recherche d'information

Sous Le modèle Vectoriel

Pour l'obtention du diplôme de Master II

Spécialité : Réseaux et télécommunications

Réalisé par : Melle.DADOUNE Soumia

M elle. TOUAK Amina

Dirigé par : KHAROUBI Sahraoui

Année universitaire 2014-2015

Remerciements

Je remercie, tout d'abord, Dieu tout puissant de m'avoir donné le courage, et la patience d'achever ce modeste travail.

Je tiens tout d'abord à exprimer ma profonde gratitude à Monsieur Sahraoui Kherroubi mon encadreur.

Au département d'Informatique université de Tiaret, pour sa présence scientifique et humaine, et l'honneur qu'il m'on fait en acceptant de nos encadrer.

Avec un intérêt constant et une grande compétence ainsi pour l'intérêt qu'ils ont bien voulu porter à nos travail.

Mes respects et ma gratitude vont également aux membres du jury qui m'ont fait l'honneur de juger ce travail et qui par leur disponibilité, leurs observations et leurs rapports m'ont permis d'enrichir notre travail.

Je remercie toutes les personnes qui ont participé de manière directe ou indirecte à la concrétisation de ce travail.

Dédicace

A l'homme de ma vie, celui qui s'est toujours sacrifié pour Me voir réussir, que dieu te garde dans son vaste paradis, à toi Mon père ALI.

A la flamme de mon cœur, ma vie et mon bonheur ; maman que j'adore AICHA.

Aux personnes dont j'ai bien aimé la présence dans ce jour, à tous mes frères et mes sœurs,

mes neveux Mostapha et Mehdi, et toute la famille DADOUNE et HENINI , je dédie ce travail dont le grand plaisir leurs revient en premier lieu pour leurs conseils, aides, et encouragements.

A mon binôme TOUAK AMINA et toute la famille TOUAK.

Aux personnes qui m'ont toujours aidé et encouragé, qui étaient toujours à mes côtés, et qui m'ont accompagnaient durant mon chemin d'études supérieures, mes aimables amis, collègues d'étude,et frères de cœur, A toi halouma, chahrazed ,asmaa et houda ,hanane, malika et khalida,Amel.

Dadoune soumia

Dédicace

Je dédie ce modeste travail à :

Ma très chère mère Fatima pour sa grande affection, tendresse, son dévouement et ses prières pour mon bonheur.

Mon père Tayeb pour son amour inestimable, son confiance, son soutien, son sacrifice et toutes les valeurs qu'il a su m'enseigner.

Mes Sœurs : Asmaa, Achoiak, Ikhlass et Malak.

Mon frère: Mohamed Mustapha.

Toute la famille : 'Touak' et 'Triki'.

Mon binôme : Dadoune Soumia et toute sa famille

A Mes meilleures amies et à toute ma promotion Master 2 RI.

Ils vont trouver ici le témoignage d'une fidélité et d'une amitié infinie

ET

Que Dieu vous protège tous et vous bénisse à tous ceux que j'aime.

Touak Amina

Résume

La représentation de l'information numérique à considérablement évolué, elle est passée d'une représentation traditionnelle simple à une représentation bien plus riche.

Notre mémoire s'inscrit dans le domaine de la recherche d'information et l'amélioration du processus de recherche. En effet, le grand nombre de documents numériques disponible a soulevé l'attractivité des outils de recherche d'information. Ceci à conduit les chercheurs à s'intéresser au développement d'outils automatique permettant d'exploiter aux mieux l'ensemble des informations disponibles pour ce type de documents. La représentation du besoin d'information est constitué d'un petit ensemble de mots-clés plus souvent connu sous la dénomination de « requête ». Or, quelques mots peuvent ne pas être suffisants pour représenter précisément et efficacement l'état cognitif complet d'un humain par rapport à son besoin d'information initial. Un système de recherche d'information SRI peut ne pas renvoyer certains documents pertinents exprimant des concepts n'étant pas explicitement évoqués dans la requête. Dans ce travail, Nous présentons une extension du modèle vectoriel plus particulièrement à adapter une requête active à un profil utilisateur plus au moins stable.

Mot clés :

Recherche d'information, , modèle vectoriel, Indexation, la pertinence.

Sommaire

Chapitre IV

Sommaire

Liste des figures

Introduction générale

Face à l'évolution rapide des moyens d'information et de communication, une masse importante d'information est produite tous les jours à travers des milliers de livres et d'articles de journaux. Cette forte augmentation quantitative d'information à gérer et à consulter ne peut plus se contenter des méthodes et des techniques classiques de stockage et de consultations utilisées. Dans ce contexte, les Systèmes de Recherche d'Information (SRI) sont développés en Vue d'automatiser la gestion de ces informations et restituer à un utilisateur l'information qu'il recherche, le plus facilement et le plus rapidement possible. A cet effet, Ils sont dotés de fonctionnalités de stockage et d'organisation des informations, ainsi que des fonctionnalités de recherche selon des modèles différents et de restitution des documents susceptibles de répondre à une demande donnée.

L'avènement du web a popularisé ces systèmes de recherche d'information et a permis à beaucoup d'utilisateurs, avisés ou occasionnels de produire et d'accéder d'une façon rapide et sans aucune contrainte à des milliers d'informations. Dés lors une simple recherche d'un utilisateur. La capacité du système de recherche d'information à séparer les informations pertinentes de celles qui ne le sont pas.

Le (SRI) a pour but de mettre en correspondance une représentation du besoin de l'utilisateur avec une représentation du contenu des documents au moyen d'une fonction de correspondance. L'évaluation d'un SRI consiste à mesurer ses performances vis-à-vis du besoin de l'utilisateur, bien que les méthodes d'évaluation largement adoptées en RI sont basées sur des modèles déférents, Pour notre part, nous nous intéressons à étendre le modèle vectoriel par une stratégie passive dans un corpus hétérogène. Des métriques usuelles sont discutées pour le modèle adopté. Ainsi, la satisfaction de l'utilisateur est basée essentiellement sur un score de pertinence calculé par notre application. Notre travail est organisé comme suit le : nous passons en revue les éléments de base des SRI en chapitre 01 ensuite, le chapitre 02 est consacré à la présentation de la pluparts des modèles reconnus en RI. En chapitre 03 nous avons détaillé le modèle vectoriel suivi par une implémentation de ce modèle doté d'une fonctionnalité passive en perspective d'aider l'utilisateur d'avoir des sdocuments pertinents.

Chapitre I

Les Concepts de base de RI

1 Contexte et problématique

Le monde assiste depuis ces dernières décennies, a une production massive d'informations dans tous les domaines d'intérêt. De multiples directions de recherche ont tenté de mettre en œuvre des processus automatiques d'accès à l'information. L'objectif est d'exploiter au mieux les bases volumineuses de ces informations.

Un Système de Recherche d'Information (SRI), nécessite la combinaison de modèles et algorithmes. Ces derniers permettent la représentation, le stockage, la recherche et la visualisation des informations. L'objectif principal de ce système est de mettre en œuvre un processus de comparaison entre besoin utilisateur et documents d'une collection dans le but de retrouver ceux qui sont pertinents. L'élaboration d'un mécanisme de recherche d'information pose alors des problèmes lies tant à la représentation qu'a la localisation de l'information pertinente. En effet, la recherche d'information induit un processus d'inférence véhicule par l'objet de la requête, en se basant sur une description structurelle des unîtes d'information.

Tout au long de premier chapitre, notre intérêt se porte ainsi sur les principes de la recherche d'information. En décrivent ses concepts de base ainsi que les étapes de la recherche d'information.

En effet, le résultat d'une recherche ne peut-être pertinent si la requête ne décrit pas explicitement et clairement les besoins de l'utilisateur ainsi que la bonne représentation des informations apparus dans la base de collection des documents (Corpus). Cela, issu au mal compréhension du domaine recherché ou à une limitation des connaissances d'utilisateur.

Nous terminons ce chapitre par les différentes types d'indexation plus particulièrement dans le contexte automatique

2 Généralités sur les systèmes de recherche d'information (SRI)

2.1 Définition

Le but d'un système de recherche d'information est de retrouver des documents en réponse à une requête des usagers, de manière à ce que les contenus des documents soient pertinents au besoin initial d'information de l'usager.

Un système de recherche d'information est défini par un langage de représentation des documents (qui peut s'appliquer à différents corpus de documents) et des requêtes qui expriment un besoin de l'utilisateur (sous forme de mots-clés par exemple), et une fonction de mise en correspondance du besoin de l'utilisateur et du corpus de documents en vue de fournir comme résultats des documents pertinents pour l'utilisateur, c'est-à-dire répondant à son besoin d'information. La figure I.1, présente un système de recherche d'information.

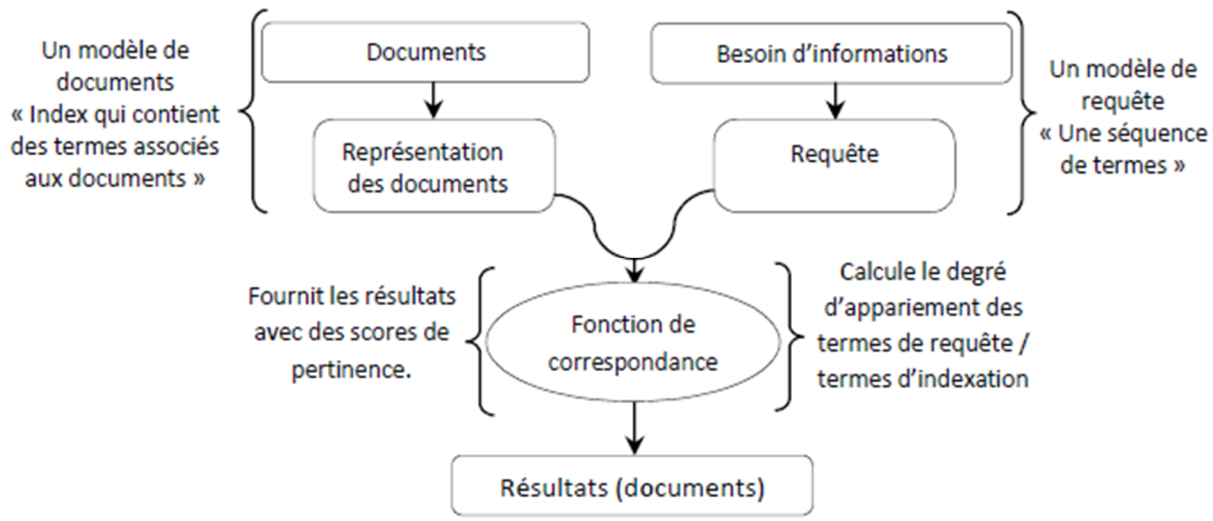

Figure I.1 : Système de Recherche d'Information.

2.2 Concept de base de la RI

La recherche d'information est l'ensemble des techniques permettant de gérer des textes ou des documents implique stocker, rechercher et explorer des documents pertinents.

Un système de recherche d'information intègre un ensemble de techniques et de processus permettant de sélectionner dans une collection de documents ceux qui sont susceptibles de répondre au besoin d'un utilisateur, ces processus permettent :

- la représentation des informations et des besoins.
- l'interrogation, la recherche et la sélection des informations pertinentes répondant aux besoins d'un utilisateur. [18] [20]

La problématique majeure émanant de tout système de recherche d'information est de retrouver les quelques dizaines ou milliers de documents pertinents parmi des Millions de documents cet écart de cardinalité rend cette tache encore plus difficile. [25]

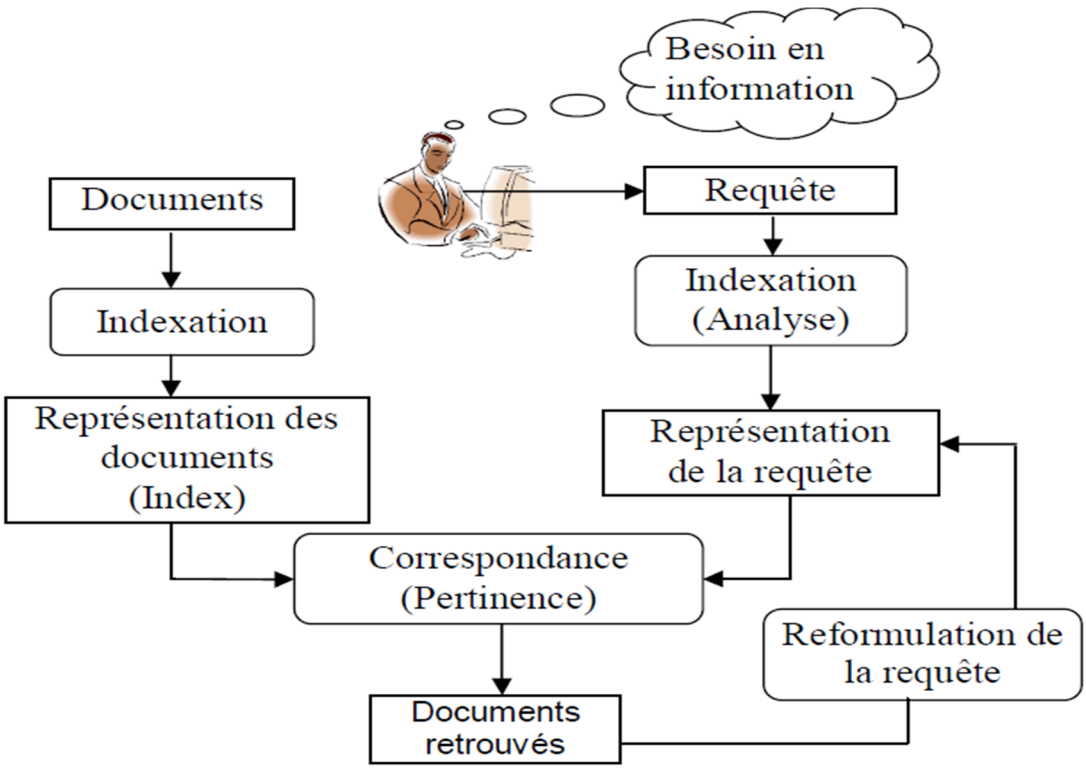

Figure I.2 : Architecture générale d'un système de recherche d'information**.**

2.3 Etapes de la recherche d'information

Un système de recherche d'information manipule un corpus de documents qu'il transpose à l'aide d'une fonction d'indexation en un corpus indexé. Ce corpus lui permet de résoudre des requêtes traduites à partir de besoins utilisateur. Un tel système repose sur la définition d'un modèle de recherche d'information qui effectue ces deux transpositions et qui fait correspondre les documents aux requêtes. La transposition d'un document en un document indexé repose sur un modèle de document. De même, la transformation du besoin utilisateur en requête repose sur un modèle de requête. Enfin, la correspondance entre une requête et des documents s'établit par une relation de pertinence.

Dans la section qui suit, nous allons définir les éléments de la recherche d'information séparément.

2.3.1 Besoin d'information

La notion de besoin en information en recherche d'informations est souvent assimilée au besoin de l'utilisateur. Trois types de besoin utilisateur ont été définis par [22]

- Besoin vérificatif

  L'utilisateur cherche à vérifier le texte avec les données connues qu'il possède déjà. Il recherche donc une donnée particulière, et sait même souvent comment y accéder. La recherche d'un article sur Internet à partir d'une adresse connue serait un exemple d'un tel besoin. Un autre exemple serait de chercher la date de publication d'un ouvrage dont la référence est connue. Un besoin de type vérificatif est dit stable, c'est-à-dire qu'il ne change pas au cours de la recherche.

- Besoin thématique connu

  L'utilisateur cherche à clarifier, à revoir ou à trouver de nouvelles informations dans un sujet et domaine connus. Un besoin de ce type peut être stable ou variable ; il est très possible en effet que le besoin de l'utilisateur s'affine au cours de la recherche. Le besoin peut aussi s'exprimer de façon incomplète, c'est-à-dire que l'utilisateur n'énonce pas nécessairement tout ce qu'il sait dans sa requête mais seulement un sous-ensemble. C'est ce qu'on appelle dans la littérature le label.

- Besoin thématique inconnu

  Cette fois, l'utilisateur cherche de nouveaux concepts ou de nouvelles relations hors des sujets ou domaines qui lui sont familiers. Le besoin est intrinsèquement variable et est toujours exprimé de façon incomplète.

2.3.2 Document

Le document constitue l'information élémentaire d'une collection de documents, l'information élémentaire appelée aussi granule de document, peut représenter tout ou une partie d'un document. Nous utilisons dans la suite de ce mémoire le terme document pour représenter le document ou le granule documentaire.

2.3.3 Collection de documents

La collection de documents (ou fond documentaire, corpus) constitue l'ensemble des informations exploitables et accessibles. Elle est constituée d'un ensemble de documents. Dans le cas général et pour un souci d'optimalité, la base constitue des représentations simplifiées mais suffisantes pour ces documents. Ces représentations sont étudiées de telle sorte que la gestion (ajout suppression d'un document) ou l'interrogation (recherche) de la base se font dans les meilleures conditions de coût.

2.3.4 Requête

La requête constitue l'expression du besoin en information de l'utilisateur. Elle représente l'interface entre le SRI et l'utilisateur. Divers types de langages d'interrogation sont proposés dans la littérature. Une requête est un ensemble de mots clés, mais elle peut être exprimée en langage naturel, booléen.

2.3.5 Analyse

La représentation des documents et des requêtes est supportée par un ensemble de règles et notations permettant la traduction d'une requête ou d'un document d'une description brute vers une description structurée. Ce processus de conversion est appelé Indexation.

2.3.6 Correspondance

Le processus d'appariement requête-document est le noyau d'un système de recherche D'information. Il permet d'associer à chaque document une valeur de pertinence vis à vis d'une requête. Les documents ayant une pertinence positive sont sélectionnes.

La mesure de pertinence est calculée à partir d'une fonction de similarité, notée *RSV* (*Q*) (Retrieval Sattus Value), *Q* étant une requête et *d* un document. Elle tient compte des poids des termes détermines en fonction d'analyses statiques et probabilistes. Notons que ce processus est étroitement lié aux représentations des documents et des requêtes.

En effet, si l'opération d'indexation est la même dans la plupart des modèles de recherche d'information, ces derniers différents souvent par rapport aux fonctions utilisées pour la mesure des poids et pour l'appariement requête-document.

La notion de pertinence est souvent difficile a appréhender, on parle souvent de deux types de pertinence :

- la pertinence utilisateur le document est jugé pertinente par l'utilisateur en fonction de son besoin en information.

- la pertinence système le document est jugé pertinente par le SRI pour une requête sur la base de la fonction de pertinence.

2.3.7 Reformulation de la requête

Le but essentiel d'un système de recherche d'information est de permettre à l'utilisateur d'avoir un résultat satisfaisant (des documents pertinents) par apport à son besoin exprimé par une requête. La possibilité de reformuler la requête initiale s'avère intéressante dans le processus de la RI. Cela fera en sorte que le résultat retourné soit plus pertinent. Il existe trois méthodes de reformulation de requêtes :

- La reformulation manuelle

Elle consiste à présenter à l'utilisateur une liste de documents jugés pertinents en réponse à la requête initiale. C'est à l'utilisateur de sélectionner à partir des documents pertinents ceux dont lesquels le système va extraire les termes à rajouter à la requête initiale dans le but d'effectuer une nouvelle recherche.

- La reformulation semi-automatique

Cette technique nécessite l'intervention de t'utilisateur qui doit identifier et sélectionner les documents pertinents et les documents non pertinents. [6][22]

- La reformulation automatique

La reformulation automatique de requêtes permet de générer une requête plus adéquate à la recherche d'information dans l'environnement du SRI, que celle initialement formulée par l'utilisateur. Son principe est de modifier la requête de l'utilisateur par ajout de termes significatifs et/ou par estimation de leur poids.

Cette reformulation intervient dans un processus plus général d'optimisation de la fonction de pertinence. Celle-ci a pour but de rapprocher la pertinence système de la pertinence utilisateur. Elle se présente comme une opération primordiale dans un SRI. En effet, compte tenu des volumes croissants des bases d'information, retrouver des informations pertinentes en utilisant seulement la requête initiale est une tâche quasi-impossible (les taux de précision obtenus dans la tache ad hoc de TREC2 ne dépassent pas 30%). [19]

La dimension de l'espace de recherche est élevée, c'est ainsi que la difficulté fondamentale de la reformulation de requêtes est la définition de l'approche à adopter en vue de réduire l'espace de recherche, cette approche [7] passe par la détermination de :

- critères de choix des termes d'extension.
- règles de calcul des poids des nouveaux termes.
- hypothèse de base quant aux liens entre termes et documents.

La reformulation de requête par injection de pertinence est plus connue sous le nom de Relevance feedback [26] [2] est le processus le plus utilisé (aussi utilise pour l'indexation [9]). Cette méthode permet une modification de la requête initiale, sur la base des jugements de pertinence de l'utilisateur sur les documents restituent par le système.

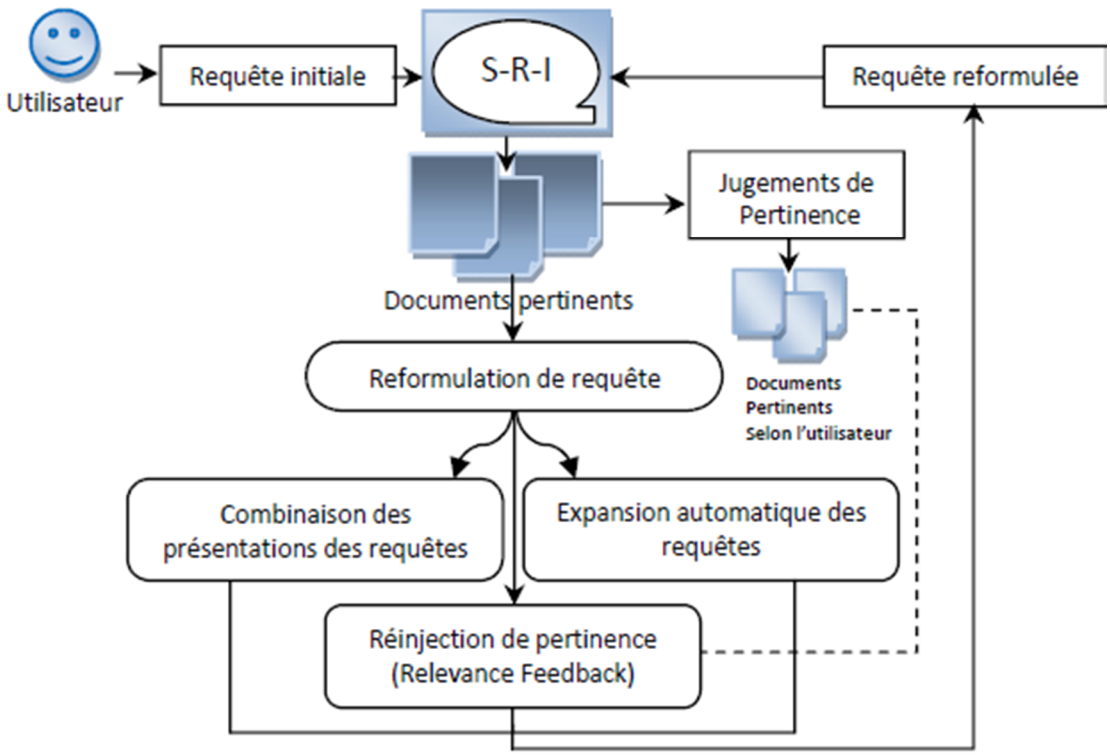

Figure I.3 : Reformulation des requêtes.

3. Indexation

L'indexation consiste à extraire des documents les mots les plus discriminants encore appelés index. Cette première tâche est généralement effectuée en marge du processus de recherche car, la construction des index peut être assez longue en fonction du nombre de documents de la collection ainsi que de la taille des documents. Les index ont un caractère réducteur car tous les termes d'un document ne sont pas importants à prendre en compte pour la recherche.

L'indexation peut se faire de trois manières différentes : manuellement (faite par un humain), de manière semi-automatique (par exemple créée par un humain assisté d'un programme proposant des termes), ou de manière automatique (crée par un programme informatique).

3.1 Types d'indexation

- Indexation manuelle

C'est le documentaliste ou un spécialiste du domaine qui effectue l'analyse du document, pour identifier son contenu et construire une représentation de ce contenu (choix des mots effectué par des indexeurs).Elle est basée sur un vocabulaire contrôlé (lexique, liste hiérarchiques, thésaurus, ontologie). [4]

- Indexation semi-manuelle

Le choix final revient au spécialiste, qui intervient souvent pour choisir d'autres termes significatifs, l'indexation semi manuelle se divise en deux parties, une partie automatique permettant d'extraire une liste de descripteur, et une deuxième partie qui est manuelle réalisée par un spécialiste du domaine dont la tâche est de sélectionner des termes significatifs parmi les descripteurs retournés auparavant.

- Indexation automatique

C'est le SRI qui génère les indexes des documents. L'indexation automatique a été crée afin de remédier aux problèmes liés aux approches précédentes, elle présente l'avantage d'une régularité du processus, car l'indexation automatique fournit toujours le même index pour le même document, ce qui constitue une qualité du système. En effet, l'indexation automatique pêche par son incapacité à interpréter un texte et son manque d'adaptation à de nouveaux vocabulaires. Il est impossible de trouver dans les documents autre chose que ce que le système peut détecter. [11]

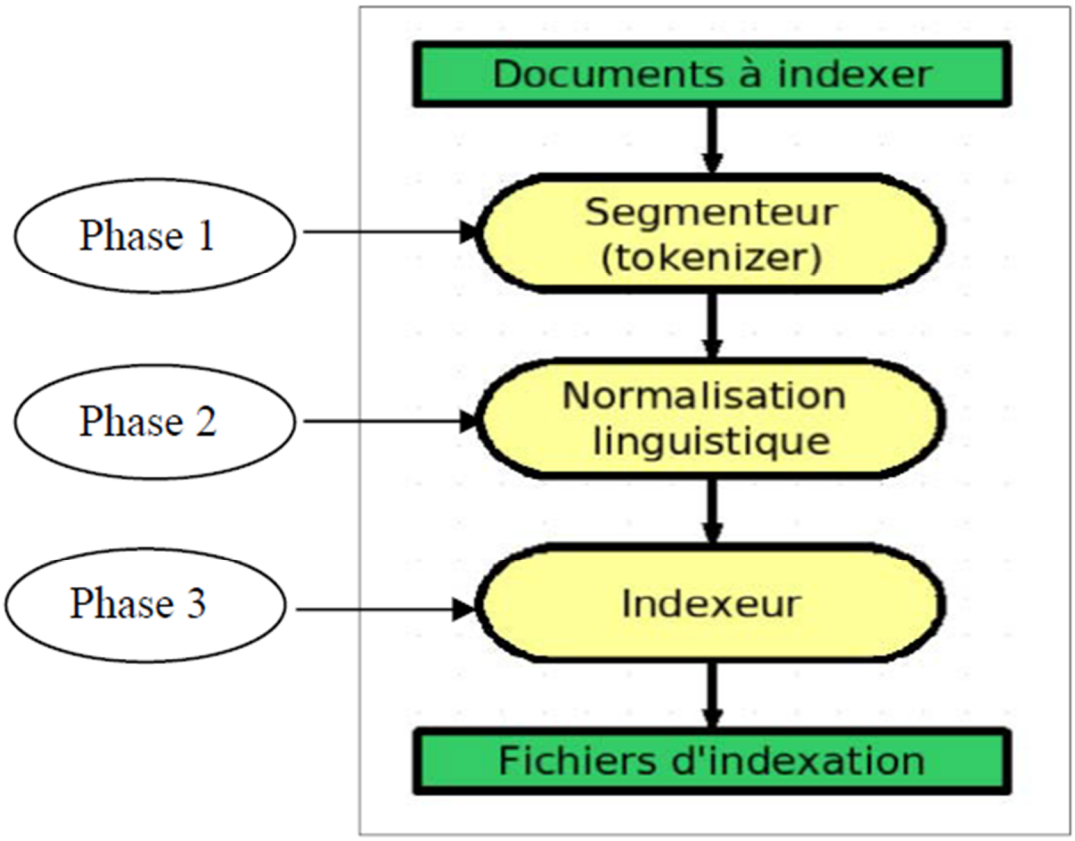

Figure I. 4 : Etapes de l'indexation automatique**.**

Le processus d'indexation automatique (figure I. 4) passe obligatoirement par 3 phases, chaque phase pouvant contenir une ou plusieurs étapes selon les usages des utilisateurs, c'est au programmeur de sélectionner les étapes qu'il souhaite intégrer au processus d'indexation automatique du corpus.

3.2 Phases d'indexation automatique

Ce processus est divise en plusieurs phase a savoir

3.2.1 Phase 1 : Segmentation

Cette phase représente la segmentation des documents en unités, cette segmentation est basée en générale sur la ponctuation et sur une liste de séparateur, le résultat de cette étape est un ensemble de mots.

3.2.2 Phase 2 : Normalisation

Cette phase peut contenir plusieurs étapes, dans ce qui suit nous allons expliquer une étape parmi les étapes les plus importantes et les plus utilisées

- Elimination des mots vides

Les mots vides sont des mots qui permettent de lier entre eux les mots d'une phrase pour la structurer (les articles, les conjonctions de coordination, les verbes auxiliaires, etc.). Ces mots ne portent pas de sens, ils ne peuvent pas constituer des index il faut donc les éliminer, cependant il ne faut pas oublier de tenir compte de certain mots vides qui auraient pour homographes des mots significatifs.

a	ces	est	n	s
à	cet	et	ni	se
ainsi	cette	été	on	soit
au	ci	il	ont	sont
aussi	comme	ils	or	sur
aux	d	l	ou	tous
avec	dans	la	p	tout
c	de	le	par	toute
car	des	les	plus	toutes
ce	donc	leur	pour	un
ceci	du	leurs	qu	une
cela	elle	mais	que	y
celle	en	même	qui	

Figure I.5 : Liste des mots outils.

3.2.3 Phase 3 : Pondération

Dans cette phase on utilise une approche permettant de sélectionner les index et de leur associer une pondération, cette dernière permet d'assigner aux termes leur degrés d'importance dans les documents, il existe trois approches pour le choix des index.

- Approche basée sur la fréquence d'occurrences

Cette approche consiste à choisir les mots représentants selon leur fréquence d'occurrence. La façon la plus simple consiste à définir un seuil sur la fréquence: si la fréquence d'occurrence d'un mot dépasse ce seuil, alors il est considéré important pour le document.

Cependant, en calculant ces fréquences, on s'aperçoit que les mots les plus fréquents sont des mots fonctionnels (mots vides) .

- Approche basée sur la valeur de discrimination

Par "discrimination", on réfère au fait qu'un terme distingue bien un document des autres documents, c'est-à-dire, un terme qui a une valeur de discrimination élevée doit apparaître seulement pour un petit nombre de documents. L'idée est de garder seulement les termes discriminants, et éliminer ceux qui ne le sont pas. Le calcul de la valeur de discrimination a été développé dans le modèle vectoriel.

Pour calculer la valeur de discrimination d'un terme, on doit comparer une sorte d'uniformité au sein du corpus avec celle du corpus transformé dans lequel le terme en question a été uniformisé (mis au même poids). L'idée est que, si on uniformisant le poids d'un terme dans tous les documents, on obtient une grande amélioration dans l'uniformité du corpus, ce terme était donc très différent (non uniformément distribué) dans différents documents. Il a donc une grande valeur de discrimination. En revanche, si en uniformisant le poids du terme, on n'obtient pas beaucoup d'amélioration sur l'uniformité, ce terme était donc déjà distribué de façon uniforme, donc peu discriminant.

- Approche basée sur La pondération des termes *:* ayant pour but de mesurer l'importance d'un terme dans un document.

Ce dernier traitement est basé généralement sur les notions de :

- Fréquence des termes dans un document (tf) *:* qui représente l'importance locale d'un terme.
- Fréquence inverse de document (idf) *:* est une mesure de l'importance globale d'un terme dans l'ensemble du corpus.

Le produit de ces deux mesures (tf $*$ idf), utilisé avec des formules diverses, sert donc à indiquer l'importance d'un terme par rapport aux autres.

En général, cette valeur est déterminée par la fréquence du terme dans le document. Par idf, on mesure si le terme est discriminant (ou non-uniformément distribué). Ici, on donne quelques formules de tf et d'idf souvent utilisées.

Une formule tf*idf combine les deux critères qu'on a vus: l'importance du terme pour un document (par tf), et le pouvoir de discrimination de ce terme (par idf). Ainsi, un terme qui a une valeur de tf*idf élevée doit être à la fois important dans ce document, et aussi il doit apparaître peu dans les autres documents.

tf = fréquence d'occurrence du terme dans un document f(t, d).

tf = f(t, d) / Max [f(t, d)] où Max [f(t, d)]

tf = log (f(t, d))

tf = log (f(t, d) + 1)

idf = log (N/n) où N est le nombre de documents dans le corpus, et n ceux qui contient le terme.

3.3 Résultat d'indexation

Le résultat d'une indexation est donc un ensemble de termes qui peuvent être soit un mot, soit une racine de mot (soit un terme composé si on possède un mécanisme pour reconnaître des termes composés).L'indexation des documents se fera dynamiquement relativement aux termes extraits d'une requête en utilisant la fonctionnalité d'indexation. Les termes (termes exacts et leurs variantes) extraits seront utilisés pour représenter les documents (on pourra si on le souhaite ignorer le contenu des documents ne correspondant pas à des termes reconnus). Un score de pertinence pour chaque document relativement à la requête devra être calculé, afin de pouvoir ordonner les différents résultats. Un seuil minimal de pertinence pourra par exemple être utilisé pour déterminer les documents qui devront faire partie des résultats retournés.

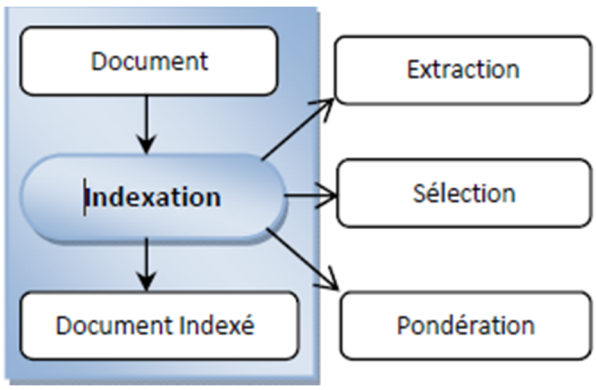

Figure I. 6 : Résultat d'indexation d'un document.

4 Interrogation

Il s'agit de l'expression du besoin d'information de l'utilisateur dans la forme imposée par le système, la recherche dans le corpus, et la présentation des résultats. Cette phase nécessite un modèle de représentation du besoin de l'utilisateur, appelé modèle de requêtes, ainsi qu'une fonction de correspondance qui doit évaluer la pertinence des documents par rapport à la requête. La réponse du système est un ensemble de références à des documents qui obtiennent une valeur de correspondance élevée. Cet ensemble est généralement présenté sous la forme d'une liste ordonnée suivant la valeur de correspondance.

5 Conclusion

Tout système de recherche d'information développe une méthodologie formelle ou opérationnelle pour affirmer si les termes de chaque document correspondent à ceux de la requête. La plupart de ces systèmes s'appuie sur l'hypothèse que les termes extraits des documents ont été parfaitement reconnus ou identifiés, et de fait leur fonction de correspondance repose sur une capacité à disposer d'une relation d'égalité entre terme du document et terme de la requête.

Dans ce premier chapitre nous avons présenté les principales notions et concepts de base de la recherche d'information. On y trouve les notions de besoin d'information, de requête, de document de pertinence et le processus d'indexation dans le chapitre qui suit, nous décrivons les différent modèles de la recherche d'information à savoir le modèle booléen, le modèle langages et le modèle probabiliste …etc.

chapitre II

Modèles de recherche d’information

(RI)

1 Introduction

Plusieurs axes de recherche ont tenté de mettre en œuvre des processus automatiques d'accès à l'information. Le but est d'exploiter au mieux les bases gigantesques de ces informations. Un Système de Recherche d'Information (SRI), nécessite l'hybridation de modèles et algorithmes. Ces derniers permettent la représentation, le stockage, la recherche et la visualisation des informations. L'objectif principal de ce système est de mettre en œuvre un processus de comparaison entre besoin utilisateur et documents d'une collection (corpus) afin de retrouver ceux qui sont pertinents. La réalisation d'un prototype de recherche d'information RI pose alors des problèmes liés tant à la représentation qu'a la localisation de l'information pertinente. En effet, la RI induit un processus d'inférence mené par l'objet de la requête, en se basant sur une description structurelle des unités d'information.

2 Prototypes de recherche d'information

Un modèle de RI a pour rôle de fournir une formalisation du processus de RI et un cadre théorique pour la modélisation de la mesure de pertinence. Il existe un grand nombre de modèles de RI textuelle développé dans la littérature. Ces modèles ont en commun le vocabulaire d'indexation basé sur le formalisme mots clés et diffèrent principalement par le modèle d'appariement requête-document.

Différents modèles de recherche d'information ont été proposés, les principaux sont :

– le modèle booléen ou ensembliste.

– le modèle vectoriel.

– le modèle probabiliste.

– le modèle connexionniste.

– les modèles de langage …etc.

Dans ce qui suit, nous détaillons chacun de ces modèles.

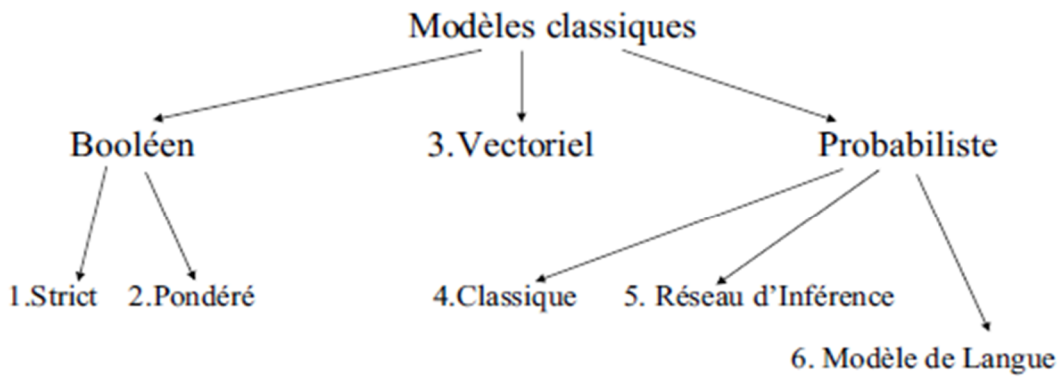

Figure II.1: les modèles classiques du RI**.**

2.1 Modèle booléen

Le modèle booléen est basé sur la théorie des ensembles. Dans ce modèle, les documents et les requêtes sont représentés par des ensembles de mots clés. Chaque document est représenté par une conjonction logique des termes non pondérés qui constitue l'index du document. Un exemple de représentation d'un document est comme suit :

$$d = t1 \wedge t2 \wedge t\,3 ... \wedge tn \qquad (1)$$

Une requête est une expression booléenne dont les termes sont reliés par des opérateurs logiques (OR, AND, NOT) permettant d'effectuer des opérations d'union, d'intersection et de différence entre les ensembles de résultats associés à chaque terme. Un exemple de représentation d'une requête est comme suit :

$$q = (t1 \wedge t2) \vee (t3 \wedge t4) \qquad (2)$$

La fonction de correspondance est basée sur l'hypothèse de présence/absence des termes de la requête dans le document et vérifie si l'index de chaque document *dj* implique l'expression logique de la requête *q*. Le résultat de cette fonction est donc binaire est décrit comme suit :

$$RSV(q, d) = \{1,0\} \qquad (3)$$

2.1.1 Avantages

Le modèle de recherche booléen est reconnu pour sa force pour faire une recherche très restrictive et obtenir, pour un utilisateur expérimenté, une information exacte et spécifique.

- La simplicité du modèle le rend aisément compréhensible pour un utilisateur.

- L'efficacité du modèle est due aux spécialistes qui ont explorés le corpus avec une bonne connaissance du vocabulaire.

2.1.2 Inconvénients

L'inconvénient majeur de ce modèle comme est que les documents pertinents dont la représentation ne correspond qu'approximativement à la requête ne sont pas sélectionnés.

- Tous les termes ont la même importance et il est incapable de trier les documents pertinents.
- L'impossibilité de rendre compte d'une correspondance partielle d'un document à une requête.
- la pondération binaire des termes du vocabulaire limite la pertinence des résultats et ne permet pas de les ordonner. [14]
- Un terme indexe totalement un document ou ne l'indexe pas du tout pour le modèle booléen strict.
- Un document est totalement similaire à une requête ou ne l'est pas du tout dans le cas pour le modèle booléen strict.

2.2 Modèle basé sur les ensembles flous

Une autre extension du modèle booléen est basée sur la théorie des ensembles flous proposée par [27]. Dans la théorie des ensembles flous, quand un élément a un degré d'appartenance à un ensemble, cet ensemble est dit ensemble flou. Cette théorie a influencé les chercheurs en RI pour modéliser les notions de vague et d'imprécision qui existent à différents niveaux du processus de RI [12], et surtout pour réduire l'incomplétude et traiter l'imprécision dans les processus d'indexation et de recherche. Dans ce modèle, un document est représenté comme un ensemble de termes pondérés comme suit :

$$D_j = \{(t_1, a_1), (t_i, a_i)\} \qquad (4)$$

Où : a_i est le degré d'appartenance du terme t_i au document D_j.

2.2.1 Avantages

- de réduire l'imperfection et de traiter l'imprécision qui caractérise le processus d'indexation.
- de contrôler l'imprécision de l'utilisateur dans sa requête

- de traiter les réponses reflétant la pertinence partielle des documents par rapport aux requêtes.

2.2.2 Inconvénients

- Ne prennent pas nécessairement en compte toutes les valeurs de L'inconvénient majeur de ces modèles est qu'ils ne sont pas adaptés au classement (ranking) des documents pertinents, étant donné que les scores de pertinence qu'ils attribuent aux documents sont calculés par des fonctions *min* ou *max* qui ne pertinences des termes de la requête.
- Des extensions ont été proposées à ces modèles [3][15] notamment pour améliorer l'ordonnancement des documents sélectionnés.[5]

2.3 Modèle probabiliste

Le modèle de recherche probabiliste utilise un modèle mathématique fondé sur la théorie de la probabilité. Le processus de recherche se traduit par calcul de proche en proche, du degré ou probabilité de pertinence d'un document relativement à une requête. Pour ce faire, le processus de décision complète le procédé d'indexation probabiliste en utilisant deux probabilités conditionnelles :

Probabilité que le terme *ti* de poids *dji* occure dans le document *Dj* sachant que ce dernier n'est pas pertinent pour la requête.

Le calcul d'occurrence des termes d'indexation dans les documents est basé sur l'application d'une loi de distribution (type loi de poisson).

En posant les hypothèses que :

La distribution des termes dans les documents pertinents est la même que leur distribution par rapport à la totalité des documents

Les variables « documents pertinents », « document non pertinent » sont indépendantes, la fonction de recherche est obtenue en calculant la probabilité de pertinence d'un document *D*.L'ordre des documents est basé sur l'une des deux méthodes :

Considérer seulement les termes présents dans les documents et requêtes.

Considérer les termes présents et termes absents dans les documents et requêtes.

La similitude entre requête et document . De manière générale, le modèle probabiliste présente l'intérêt d'unifier les représentations des documents et concepts. [4]

2.3.1 Avantages

- Apprentissage du besoin d'information.
- La fonction d'appariement permet de trier les documents.

2.3.2 Inconvénients

- Le modèle considère que tous les termes sont indépendants (inconvénient théorique)
- Pas de langage de requête pour ce modèle.
- Problème des probabilités initiales.
- Il est impossible d'ordonner les résultats de la recherche.
- Tous les documents retournés sont sur le même plan.
- L'utilisateur préfère un classement lorsque la liste est grande.
- Résultats comparables à ceux du modèle vectoriel [35]

2.4 Modèle connexionniste

Les systèmes de recherche d'informations basent sur le modèle connexionniste [8][16] utilisent les fondements des réseaux de nuerons tant pour la modélisation des unîtes textuelles que pour la mise en œuvre du processus de recherche d'information. Ce modèle peut être vu comme un modèle vectoriel récurrent non linéaire, les neurones formels représentent des objets de la recherché d'information.

L'idée de base est que la recherche d'information est un processus associatif bien représente par les mécanismes de propagation et d'activation des réseaux de neurones. Par ailleurs, les capacités d'apprentissage de ces modèles peuvent permettre d'obtenir des systèmes de recherche d'informations adaptatifs.

Sur la base de l'architecture du réseau qu'ils utilisent, les systèmes de recherché d'informations connexionnistes peuvent être repartis en deux catégories :

- les modèles à auto-organisation.
- les modèles à couches.

Dans ce qui suit nous allons définir chaque catégorie.

- Les modèles à auto-organisation :

Les systèmes de cette catégorie sont bases sur le modèle des cartes topologiques de Kohonen [13].L'apprentissage du réseau permet d'effectuer la classification des documents a partir de leurs descriptions initiales.

- Les modèles à couches :

Ces modèles sont largement utilisés en recherché d'information. Certains travaux sont basés sur le modèle probabiliste. Le réseau est généralement construit a partir des représentations initiales de documents et informations descriptives associées. La pertinence des documents est alors mesurée grâce a leur niveau d'activation. Actuellement les réseaux de neurones sont largement utilisés en recherche d'information et la notion de réseau généralement est très intéressante.

2.4.1 Avantages

L'avantage de l'approche connexionniste est que les représentations associées ont une mesure de similarité intégrée qui traite facilement le problème de « similaire mais pas identique ». ACME [13] propose une architecture où l'analogie résulte de l'activation des nœuds du réseau. Contraintes doivent être satisfaites simultanément : la similarité structurelle, la similarité sémantique, et l'importance pragmatique.

2.4.2 Inconvénients

- Robuste au bruit.
- moins typique : problèmes symbolique.

2.5 Modèle de langages

Dans les modèles classiques de recherche, on cherche à mesurer la similarité entre un document d_i et une requête q ou à estimer la probabilité que le document réponde à la requête L'hypothèse de base dans ces modèles consiste à dire qu'un document n'est pertinent que s'il "ressemble" à la requête. Les modèles de langage, comme décrits par leurs initiateurs [17], sont eux basés sur une hypothèse différente. Fournit une requête en pensant à un ou plusieurs documents qu'il souhaite retrouver. Par conséquent, un document n'est pertinent que si la requête de l'utilisateur ressemble à celle inférée, On cherche alors à estimer la probabilité que la requête soit inférée par le document (). Pour cela, on fait les hypothèses suivantes :

- une requête est exprimée comme une suite $Q_k = q_{1k}, \ldots, q_{nk}$, de termes.
- Les termes de la requête sont indépendants (comme c'est le cas pour les modèles probabiliste et vectoriel).

- Tous les $P(Q_j / D_k)$ termes d'indexation (ceux présents dans la requête et ceux absents) sont considérés.

Etant donnée une requête composée de l'ensemble des termes et ne contenant pas les autres termes du document. [2]

2.5.1 Avantages

- la pertinence d'un document ne doit pas influencer la pertinence d'un autre document.
- Ce modèle a acquis une grande popularité, en raison de sa simplicité, efficacité, performance et son fondement théorique solide.
- Il offre la possibilité de combiner différent information de combiner différentes informations sur un document pouvant être liée au contenu textuel du document ou non liées telles que la structure du document.

2.5.2 Inconvénients

- la pertinence des documents doit être une variable aléatoire binaire prenant l'une des valeurs vrai ou faux.

3. Conclusion

Les systèmes de recherche d'information (SRI) sont conçus à l'origine pour retrouver les documents traitant d'un sujet donné. Or, la majorité des systèmes actuels, présentés ici, se contentent de chercher les documents qui contiennent les mêmes mots que ceux de la requête. Ceci est évidemment insuffisant dans la mesure où l'utilisateur du système ne connaît pas à priori le vocabulaire que l'auteur a utilisé dans le document. Actuellement, ces systèmes décrivant l'information par une liste de mots simples (bag of words), même s'ils sont relativement performants (45% de précision dans le cadre de TREC), sont connus comme générant beaucoup de bruit.

Des approches basées sur le concept comme unité de représentation, sont présentées par des chercheurs comme solution au problème de la représentation de l'information en RI**.**

Dans ce deuxième chapitre, nous sommes essentiellement intéressés aux modèles de recherche et de représentation d'information d'une façon particulière. Chacun de ces modèles contribue à la résolution des problèmes inhérents à la recherche d'information. Certains modèles traitent de la représentation des documents et des requêtes, d'autres traitent de la modification automatique de la requête. La finalité de chaque système de recherche d'information est de satisfaire les besoins des utilisateurs, ces derniers sont préoccupés par un seul problème : celui de pouvoir récupérer tous les documents dont il a besoin d'une façon rapide et efficace.

Afin de réduire la complexité des documents et de faciliter leur manipulation, il faut transformer chaque document, sa version textuelle intégrale, en un vecteur qui décrit le contenu du document. Dans ce qui suit ce modèle est détaille dans le chapitre III.

Chapitre III

Le modèle vectoriel

1 Principe du modèle vectoriel

Le principe fondamental est qu'il doit y avoir correspondance optimale entre les termes de la requête et ceux contenus dans les documents indexés. Le moteur compare l'ensemble des termes présents dans la requête à l'ensemble des radicaux des termes présents dans la base de données, effectue une mesure de leur proximité et délivre une liste de réponses triées suivant ce critère de proximité. Ainsi, les documents correspondant le mieux à la requête sortent au début de la liste. Il s'agit donc d'un calcul de proximité entre la requête et les descriptions, que l'on appelle pertinence (*Relevance* en anglais).Une des façons les plus utilisées pour calculer la pertinence s'appuie sur le modèle vectoriel. Chaque document et chaque requête sont caractérisés par la liste de ces valeurs numériques qui forment un vecteur.

Ils peuvent donc être traités mathématiquement comme des vecteurs dans l'espace . [22]

L'idée au cœur de notre approche est de se servir d'un nombre *m* de documents dits *documents-références*. Pour chaque document de la collection, on calcule alors sa similarité, qui est un score, à chacune de ces références selon le modèle de similarité imposé par le SRI utilisé (chaque document de la collection joue le rôle de la requête).

Les *m* valeurs obtenues sont rassemblés dans un vecteur qui caractérise désormais chaque document. Ce processus, éventuellement coûteux, se fait bien sûr hors-ligne.

Les requêtes subissent le même traitement, en ligne, toutefois.

Les requêtes et les documents se trouvent donc représentés par des vecteurs de dimensions *m*. La distance entre une requête et les documents de la collection se calcule ensuite de manière similaire à ce qui se fait dans les modèles vectoriels, et ce indépendamment du modèle utilisé pour construire ces vecteurs.

La recherche de documents pertinents pour une requête se fait donc de manière indirecte, par le bais des proximités avec les documents-références. La proximité finale peut donc être dite de second ordre. Cela implique aussi qu'un document puisse être jugé pertinent pour une requête même s'il ne contient aucun mot de cette requête. Bien entendu, pour que cette technique soit intéressante d'un point de vue calculatoire, il faut que $m < D$, où D est le nombre de documents de la collection.

2. Interprétation du Modèle vectoriel

Le modèle vectoriel est une représentation mathématique du contenu d'un document, selon une approche algébrique. Ce modèle se base sur une formalisation géométrique. En effet les documents et les requêtes sont représentés dans le même espace, défini par un ensemble de dimensions, chaque dimension représente un terme d'indexation.

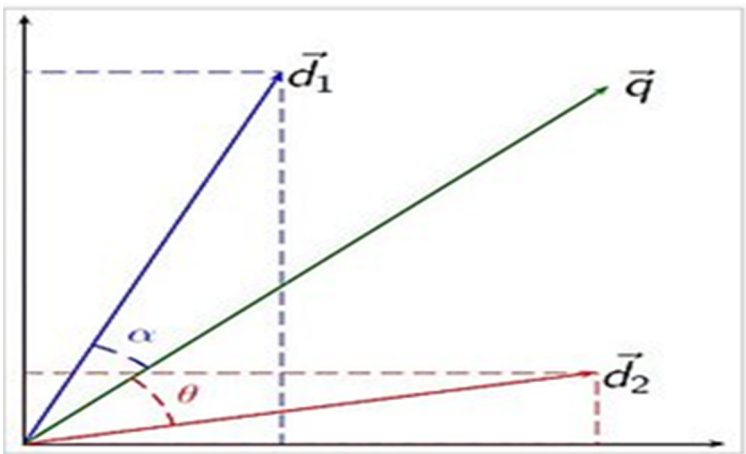

Figure III.1 : représentation de deux documents (d1et d2) et d'une requête (q).

Dans un espace vectoriel.la proximité de la requête aux documents est représentée par les angles α et θ entre les vecteurs.

Dans un espace vectoriel, il est possible de calculer la distance entre le vecteur qui correspond à la requête et ceux qui correspondent à des descriptions. Il s'agit d'une mesure de similarités, qui s'appuie souvent sur la valeur du cosinus entre deux vecteurs. Une valeur 1 coïncide avec une correspondance exacte. Ensuite, la valeur diminue à mesure que les vecteurs s'éloignent : 0 correspond à un angle de 90° et -1 à un angle plat (vecteurs diamétralement opposés).

L'ensemble de représentation des documents est un vocabulaire comprenant des termes d'indexation. Ceux-ci sont typiquement les mots les plus significatifs du corpus considéré : noms communs, noms propres, adjectifs... Ils peuvent éventuellement être des constructions plus élaborées comme des expressions ou des entités sémantiques. À chaque élément du vocabulaire est associé un index unique arbitraire. On parle aussi de 'sacs de mots' où les mots sont considérés comme indépendants et où l'ordre est sans importance. La valeur de chaque caractéristique est appelé le poids du terme et est en général une fonction de

fréquence de termes dans le document. Par conséquent, en utilisant la fréquence de chaque terme comme un poids, les termes qui apparaissent le plus fréquemment sont plus importants et donc descriptifs du document, Documents et requête sont représentés par un vecteur. La représentation d'un document sous forme vectorielle se déroule en deux étapes :

- extraire les termes pertinents du document.
- calculer les poids des termes restants.

2.1 Extraction des termes pertinents

Il s'agit de prétraiter le texte des documents textuels en supprimant les mots-vides, la ponctuation et les éventuels 'retours-chariots', de lemmatiser le texte et de le segmenter.
Exemple : A partir des documents d1 et d2 donnésdans le figure III .1, après élimination de la ponctuation, élimination des mots-vides et lemmatisation , l'ensemble de termes pertinents pour d1 est : {yeux, si, profond, pench , boir, v, tou, soleil, ven, mir, jet, mour, desesp, perd, memog}; pour d2 : { admir, jour, vaste, parc, pam, sous, œil, brul, soleil, comme, jeune, domin, amour}.figure III.2.

2.2 Calcul des poids

Le poids de chaque terme dans un document peut être obtenu de différentes manières : booléenne, fréquence des termes, TF-idf (Term frequency - Inverse Document Frequency).

Formellement un modèle vectoriel est défini par un ensemble de vecteurs de base linéairement indépendants

Chaque contenu est ainsi représenté par un vecteur *v*, dont la dimension correspond à la taille du vocabulaire. Chaque élément v_i du vecteur *v* consiste en un poids associé au terme d'indice *i* et à l'échantillon de texte. Un exemple simple est d'identifier v_i au nombre d'occurrences du terme *i* dans l'échantillon de texte. La composante du vecteur représente donc le poids du mot i dans le document. L'un des schémas de pondération les plus usités est le TF-IDF.

2.2.1 Fréquence des termes

Pour la fréquence des termes, le poids d'un terme est obtenu en comptant les occurrences du terme dans le document : tfi;j représente donc la fréquence du terme i dans le document j

Exemple : Les représentations vectorielles de d1 et d2 avec la fréquence des termes sont :

Figure III.2 :Exemple de fréquence des termes

- La lemmatisation du contenu d'un texte permet de regrouper les mots d'une même famille. Chacun des mots d'un contenu se trouve ainsi réduit en une entité appelée lemme (forme canonique). La lemmatisation regroupe les différentes formes que peut revêtir un mot, soit : le nom, le pluriel, le verbe à l'infinitif, ...etc.
- Les mots considérés vides, ici, sont : pour, le, la, les, l',..., un, une, des, ..., y, à, se, s', les dérivés du verbe être, les dérivés du verbe avoir, mon, ton, son, mes, tes, ..., qu', que, qui,...
- Quelques exemples de lemmatisation pour cet exemple : yeux est Yeux, profonds est Profond, jeter est jet, désespères est desesp, mémoire est memo, domination est domin, ... L'algorithme de Porter [24] est le plus utilisé.

2.2.2 Contribution d'un terme isolé

- S'il est présent dans le document et la requête, il augmente le score
- S'il est présent dans un des deux seulement, il diminue le score

2.2.3 Tf-Idf

Le TF-idf permet, quant à lui, d'évaluer l'importance d'un terme contenu dans un document, relativement à une collection. Le poids augmente proportionnellement avec le nombre d'occurrences du mot dans le document. Il varie également en fonction de la fréquence du mot dans la collection. Ainsi, la fréquence inverse du document (idf) est une mesure de

l'importance du terme dans l'ensemble des documents. Dans le cas du TF-idf , elle vise à donner un poids plus important aux termes les moins fréquents, considérés comme les plus discriminants. Il s'agit de calculer le logarithme de l'inverse de la proportion de documents qui contiennent le terme :

$$idf_i = log\frac{|D|}{|\{d_j : t_i \in d_j\}|} \quad (1)$$

Où |D| est le nombre total de documents et $|\{d_j : t_i \in d_j\}|$ est le nombre de documents où le terme t_i apparait. Finalement, le poids s'obtient en multipliant les deux mesures

$$tfidf_{i,j} = tf_{i,j}.idf_i \quad (2)$$

Exemple : Notre exemple contient deux documents, donc |D| = 2. Pour le terme "yeux", seul le document d_1 le contient. Par conséquence $idf_{yeux} = log(\frac{2}{1}) \approx 0,301$

De même, le terme "soleil" apparait dans les deux documents, donc

$idf_{soleil} = log(\frac{2}{2}) = 0.$

Le calcul du TF-idf du terme "yeux" est donc :

$$tfidf_{yeux,d_1} = tf_{yeux,d_1}.idf_{yeux} = \frac{2}{27}.log\frac{2}{1} \approx 0,0222$$

Pour le document d1 et $tfidf_{yeux,d_2} = tf_{yeux,d_2}.idf_{yeux} = 0.log\frac{2}{1} = 0$

Pour le document d2. Celui du terme "soleil" est donc :

$tfidf_{soleil,d_1} = tf_{soleil,d_1}.idf_{soleil} = \frac{1}{27}.log\frac{2}{2} = 0$ pour Le document d1

$tfidf_{soleil,d_2} = tf_{soleil,d_2}.idf_{soleil} = \frac{1}{27}.log\frac{2}{2} = 0$ pour le document d_2.

Les représentations vectorielles de d_1 et d_2 avec le TF-idf son

	yeux	si	profond	pench	boir	v	tou	soleil	ven	mir	jet	mour	desesp	perd	memo	admir
d_1	0,02	0,02	0,02	0,01	0,01	0,01	0,01	0	0,01	0,01	0,01	0,01	0,01	0,01	0,01	0
d_2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,01

jour	vaste	parc	pam	sous	oeil	brul	comme	jeune	domin	amour
0	0	0	0	0	0	0	0	0	0	0
0,01	0,01	0,01	0,01	0,02	0,01	0,01	0,01	0,01	0,01	0,01

Figure III.3 : Les représentations vectorielles de d_1 et d_2.

3. Proximité entre documents

Étant donnée une représentation vectorielle d'un corpus de documents, on peut introduire une notion d'espace vectoriel sur l'espace des documents en langage naturel. On en arrive à la notion mathématique de proximité entre documents.

En introduisant des mesures de similarité adaptées, on peut quantifier la proximité sémantique entre différents documents. Les mesures de similarité sont choisies en fonction de l'application.

3.1 Mesures de similarité existantes

Une mesure de similarité est en général une fonction qui quantile le rapport entre deux objets, comparés en fonction de leurs points de ressemblance et de dissemblance. Les deux objets comparés sont bien entendu de même type.

Toutes les mesures de similarité ne sont pas des métriques.

3.2 Similarité syntaxique

En mathématiques et en informatique, une mesure permettant de comparer des documents textuels, consiste à comparer des chaines de caractères. C'est une métrique qui mesure la similarité ou la dis similarité entre deux chaînes de caractères. Par exemple, les chaînes de caractères "Sam" et "Samuel" peuvent être considérées comme similaires. Une telle mesure sur les chaines de caractères fournit une valeur obtenue algorithmiquement.

Parmi de telles mesures de similarité, citons par exemple, Similarité Cosinus, la distance euclidienne le coefficient de Jaccard, Indice de Dice…etc.

3.2.1 Similarité Cosinus

La similarité cosinus est fréquemment utilisée [1] en tant que mesure de ressemblance entre deux documents d1 et d2. Il s'agit de calculer le cosinus de l'angle entre les représentations vectorielles des documents à comparer. La similarité obtenue $sim_{cosinus}$(d1; d2) ϵ [0; 1].

$$sim_{cosinus}(d_1, d_2) = \frac{\vec{d_1}.\vec{d_2}}{\|\vec{d_1}\|\|\vec{d_2}\|} \quad (3)$$

Exemple : Par souci de lisibilité et de facilité des calculs, nous nous limitons ici à des vecteurs de taille 6 contenant les termes : {yeux, profond, soleil, memo, sous, œil}.

Les représentations vectorielles de d_1, d_2 et d_3 avec le TF sont :

	yeux	profond	soleil	memo	sous	oeil
d_1	$\frac{1}{3}$	$\frac{1}{3}$	$\frac{1}{6}$	$\frac{1}{6}$	0	0
d_2	0	0	$\frac{1}{6}$	0	$\frac{1}{3}$	$\frac{1}{6}$
d_3	$\frac{1}{4}$	0	$\frac{1}{4}$	0	0	0

Nous avons donc $sim_{cosinus}$ ϵ (d_1; d_2)

$$= \frac{\vec{d_1}.\vec{d_2}}{\|\vec{d_1}\|\|\vec{d_2}\|} = \frac{\frac{1}{36}}{\sqrt{\frac{10}{36}}\cdot\sqrt{\frac{1}{6}}} \approx 0,129.$$

De la même manière, $sim_{cosinus}$ (d_1; d_3) $0,447$ et $sim_{cosinus}(d_2, d_3) \approx 0,288.$

Par conséquent, selon la similarité cosinus, d_1 et d_3 sont les plus similaires.

Cette mesure n'est pas sensible à la norme des vecteurs, donc ne tient pas compte de la longueur des documents.

Un avantage de la similarité cosinus est qu'elle peut efficacement profiter d'une implémentation par index inversé à condition d'indexer également la norme des documents. Chaque élément non nul de la requête q permet de retrouver des documents potentiellement pertinents et le produit scalaire (numérateur de la similarité cosinus) est simultanément calculé par accumulation.

Une alternative tout aussi efficace est de calculer le carré de la norme L2 entre q et d_1 exprimée par:

$$\|q - d_1\|_2^2 = \|q\|_2^2 + \|d_1\|_2^2 - 2d_1 \cdot q \quad (4)$$

3.2.2 Distance euclidienne

La distance euclidienne calcule la similarité entre deux documents d_1 et d_2 comme la distance entre leurs représentations vectorielles ramenées à un seul point.

$$sim_{euclidienne}(d_1, d_2) = \| \vec{d_1} - \vec{d_2} \| = \sqrt{\sum_{i=1}^{n}(d_{1_i} - d_{2_i})^2} \quad (5)$$

Où n est le nombre total de termes représentés. la taille des vecteurs.

Exemple : En gardant des vecteurs de taille 6, nous avons

$$(d_1, d_2) = \| \vec{d_1} - \vec{d_2} \| = \sqrt{\sum_{i=1}^{n}(d_{1_i} - d_{2_i})^2} = \sqrt{\frac{10}{27}} \approx 0,608.$$

De la même manière, distance euclidienne (d_1; d_3)

$$\approx 0,408 \text{ et } sim_{euclidienne}(d_2, d_3) \approx 0,408.$$

Par conséquent, selon la distance euclidienne, d_3 est à la même distance de d_1 que de d_2.

3.2.3 Coefficient de Jaccard

L'indice de Jaccard ou coefficient de Jaccard [10] est le rapport entre la cardinalité (la taille) de l'intersection des ensembles considérés et la cardinalité de l'union des ensembles. Il

permet d'évaluer la similarité entre les ensembles. Les documents d_1 et d_2 sont donc représentés, non pas comme des vecteurs,

Mais comme des ensembles de termes. La similarité obtenue $sim_{jaccard}$ (d_1; d_2) appartient au [0; 1].

$$sim_{jaccard}(d_1, d_2) = \frac{\|d_1 \cap d_2\|}{\|d_1 \cup d_2\|} \quad (5)$$

Il est aussi possible d'utiliser la représentation vectorielle.

$$sim_{jaccard}(d_1, d_2) = \frac{\vec{d_1}.\vec{d_2}}{\|\vec{d_1}\|\|\vec{d_2}\| - \vec{d_1}.\vec{d_2}} \quad (6)$$

Exemple : En gardant des vecteurs de taille 6, nous avons

$$sim_{jaccard}(d_1, d_2) = \frac{\vec{d_1}.\vec{d_2}}{\|\vec{d_1}\|\|\vec{d_2}\| - \vec{d_1}.\vec{d_2}} = \frac{\frac{1}{36}}{\sqrt{\frac{10}{36}} \cdot \sqrt{\frac{1}{6}} - \frac{1}{36}} \approx 0,148.$$

De la même manière

$$sim_{jaccard}(d_1, d_3) \approx 0,809 \text{ et } sim_{jaccard}(d_2, d_3) \approx 0,405.$$

Par conséquent, selon le coefficient de Jaccard, d_1 et d_3 sont les plus similaires.

3.2.4 Indice de Dice

L'indice de Dice mesure la similarité entre deux documents d_1 et d_2 en se basant sur le nombre de termes communs à d_1 et d_2

$$sim_{dice}(d_1, d_2) = \frac{2N_c}{N_1 + N_2}$$

Où N_c est le nombre de termes communs à d_1 et d_2, et N_1 (resp. N_2) est1 le nombre de termes de d_1 (resp. d_2).

Exemple : Les documents d_1 et d_2 ont deux termes en commun, à savoir "se" et "la". Le document d1 contient 36 mots. Le document d_2 contient 21 mots. Ainsi, l'indice de Dice entre les documents d_1 et d_2 vaut : .

$sim_{dice}(d_1, d_2) = \frac{2N_c}{N_1 + N_2} = \frac{2*2}{36+21} \approx 0,07$. De la même manière, $sim_{dice}(d_1, d_3) \approx 0,204$ et $sim_{dice}(d_2, d_3) \approx$ 0.117 De la même manière, Par conséquent, selon l'indice de Dice , d_1 et d_3 sont les plus similaires.[8] [29]

4. Pertinence

Dans un ensemble des documents se trouvant dans un corpus documentaires ou encore sur le web. Le but est de permettre aux utilisateurs de retrouver les documents dont le contenu répond à leur besoin en information, il s'agit donc de retourner l'ensemble de documents pertinents. Cependant, nous constatons que la notion de pertinence dépend de la satisfaction de l'utilisateur d'une part, et des différents sens portés par les termes de la requête d'une autre part. Cette constatation constitue le point faible de la recherche d'information classique, elle représente également le point de départ pour de nouveaux paradigmes de recherche

La notion de pertinence est souvent difficile à appréhender, on parle souvent de deux

Types de pertinence :

4.1 Pertinence utilisateur

Le langage de requête est plus simple les performances sont meilleures grâce à la pondération des termes. Un appariement partiel qui permet de retrouver les documents qui répondent en partie à la requête. Le document est jugé pertinent par l'utilisateur en fonction de son besoin en information

4.2 Pertinence système

Le langage de requête est plus simple les performances sont meilleures grâce à la pondération des termes. Un appariement partiel qui permet de retrouver les documents qui répondent en partie à la requête. [16]

5. Applications

Parmi les applications existantes, on peut citer :

- la catégorisation : regrouper automatiquement des documents dans des catégories prédéfinies.
- la classification : étant donné un ensemble de documents, déterminer automatiquement les catégories qui permettront de séparer les documents de la meilleure façon possible (catégorisation non supervisée).

- la recherche documentaire : trouver les documents qui répondent le mieux à une requête (ce que fait un moteur de recherche) ; la requête de l'utilisateur est considérée comme un document, traduite en vecteur, et comparée aux vecteurs contenus dans le corpus des documents indexés.
- Le filtrage : classer à la volée des documents dans des catégories prédéfinies (par exemple, identifier un spam sur la base d'un nombre suspect d'occurrence du mot « pénis » dans un mail et l'envoyer automatiquement à la corbeille).

5.1 Avantages

- Quelque soit la technique utilisée, basée sur le modèle vectoriel a de fait le même format initial, à savoir la représentation vectorielle.
- Les techniques basées sur le modèle vectoriel sont faciles à développer, il s'agit uniquement de calcul vectoriel.
- Le langage de requête est plus simple (liste de mot-clé).
- Les performances sont meilleures grâce à la pondération des termes.
- Le renvoi de documents à pertinence partielle est possible.
- La fonction d'appariement permet de trier les documents.
- Le modèle vectoriel est relativement simple à appréhender (algèbre linéaire) et est facile à implémenter. Il permet de retrouver assez efficacement des documents dans un corpus non structuré (recherche d'information), son efficacité dépendant pour une grande part de la qualité de la représentation (vocabulaire et schéma de pondération). La représentation vectorielle permet aussi une mise en correspondance des documents avec une requête imparfaite.
- Il comporte également plusieurs limitations qui furent, pour certaines, corrigées par des affinements du modèle. En particulier, ce modèle suppose que les termes représentatifs sont indépendants. Ainsi, dans un texte, l'ordre des mots n'est pas pris en compte. Dans sa version la plus simple, il ne prend pas non plus en compte les synonymes ou la morphologie des contenus.
- Simple et efficace a utiliser dans system de recherche d'information.
  [21] [23]

5.2 Inconvénients

- Des mots identiques considérés comme peu pertinents peuvent parfois trop influer sur la valeur de la similarité. Par exemple, pour les phrases "Tout est bien qui finit bien" et "C'est notre seul bien", le terme "est" n'est pas vraiment pertinent et pourtant, il va avoir un poids certain.

Notons cependant que la lemmatisation, l'élimination des mots-vides et le TF-idf permettent de pallier cet inconvénient. [29]

6. Conclusion

Dans ce troisième chapitre, nous sommes essentiellement intéressés à l'étude d'un système de recherche d'information, d'une façon générale et au modèle vectoriel d'une façon particulière. On a commencé par le principe du modèle vectoriel dans la RI dent a parlé de l'interprétation de ce modèle**,** de TF idf et de similarité entre les documents et la pertinence.

Alors il contribue à la résolution des problèmes inhérents à la recherche d'information et il récupère tous les documents dont il a besoin d'une façon rapide et efficace. Par la suite on implémente tout les détaille de notre l'application.

Chapitre IV

Métrique et implémentation d’application

1 Introduction

Une théorie sans pratique peut être qualifiée d'aveugle. Ce chapitre se rapporte donc à la mise en pratique des théories que nous avons étudiées précédemment. Il a pour vocation de présenter des illustrations pratiques de l'application réalisée à l'issue de notre mémoire.
L'évaluation des systèmes de recherche d'information est donc un thème de recherche important en sciences de l'information. Elle peut porter sur plusieurs critères : le temps de réponse, la pertinence des résultats, la qualité de la présentation, etc. Le critère le plus important est sans doute celui qui mesure la capacité du système à satisfaire le besoin de l'utilisateur en information, cette satisfaction se traduit par une forte adéquation entre la requête émise et les documents retournés. Dans ce contexte des campagnes d'évaluation ont été mises en place depuis les années soixante pour juger l'efficacité de ces systèmes et ainsi faire évoluer leurs performances sur le plan technique mais également par rapport aux attentes des utilisateurs.la section ci-dessous présente les métriques les plus utilisés dans le domaine RI , nous décrivons par la suite l'interface de notre implémentation qui consiste à réaliser le modèle vectoriel doté d'une extension passive aidant l'utilisateur à trouver de plus en plus les documents pertinents.

2 Métrique d'évaluation des SRI

L'évaluation des systèmes de recherche d'information constitue une étape importante dans l'élaboration d'un modèle de recherche d'information. En effet, elle permet de caractériser le modèle et de fournir des éléments de comparaison entre modèles. D'une façon générale, tout système de recherche d'information présente deux objectifs

- retrouver tous les documents pertinents,
- rejeter tous les documents non pertinents.

Ces deux objectifs sont évalues par les mesures de précision et de rappel définis ci-dessous.

2.1 Corpus de Test

Pour arriver à une telle évaluation, on doit connaître d'abord les réponses idéales de l'utilisateur. Ainsi, l'évaluation d'un système s'est faite souvent avec certains corpus de test. Dans un corpus de test, il y a:

- un ensemble de documents.
- un ensemble de requêtes.
- la liste de documents pertinents pour chaque requête.

Pour qu'un corpus de test soit significatif, il faut qu'il possède un nombre de documents assez élevé. Les premiers corpus de test développés dans les années 1970 renferment quelques milliers de documents. Les corpus de test plus récents (par exemple, ceux de TREC) contiennent en général plus 100 000 documents (considérés maintenant comme un corpus de taille moyenne), voir des millions de documents (corpus de grande taille).

L'évaluation d'un système ne doit pas se reposer seulement sur une requête. Pour avoir une évaluation assez objective, un ensemble de quelques dizaines de requêtes, traitant des sujets variés, est nécessaire. L'évaluation du système doit tenir compte des réponses du système pour toutes ces requêtes.

Finalement, il faut avoir les réponses idéales pour l'utilisateur pour chaque requête. Le dernier élément d'un corpus de test fournit cette information. Pour établir ces listes de documents pour toutes les requêtes, les utilisateurs (ou des testeurs simulant des utilisateurs) doit examiner chaque document de la base de document, et juger s'il est pertinent. Pour la construction d'un corpus de test, les jugements de pertinence constituent la tâche la plus difficile.

2.2 Mesures d'évaluation

Le but du processus de recherche d'information est de minimiser la distance entre la pertinence système et la pertinence utilisateur. Plusieurs mesures standards en RI ont été proposées pour évaluer les performances des SRI.

2.2.1 Mesures de Précision/Rappel

Le rappel et la précision sont deux mesures de base pour évaluer les performances des systèmes de recherche d'information. Le principe de ces deux mesures est basé sur la Connaissance à-priori des documents pertinents de la collection d'une part, et d'autre part la partition de l'ensemble des documents restitués par le SRI en deux catégories : documents

pertinents et documents non pertinents. La figure IV.1, illustre la partition de la collection de tests pour une requête.

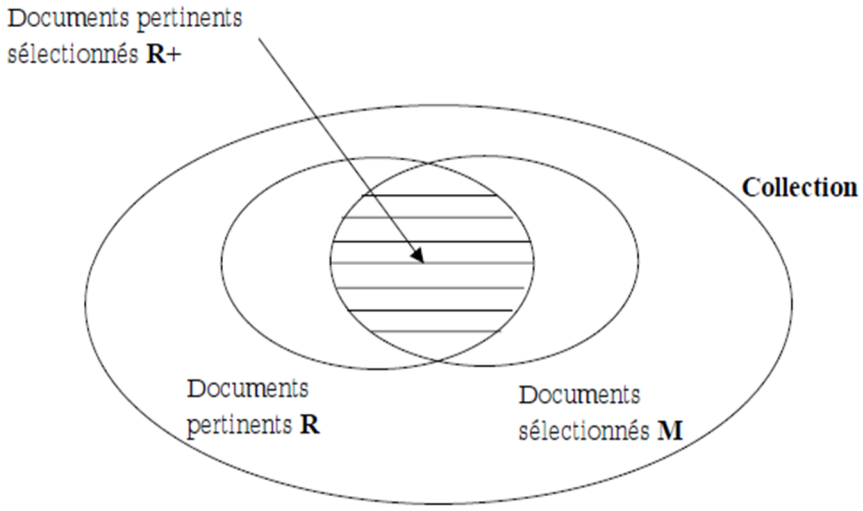

Figure IV.1 : Exemple du rappelle et précision.

2.2.2 Rappel

Cette mesure peut être vue comme une mesure de couverture du système. Elle calcule la capacité du SRI à retrouver les documents pertinents de la collection. Le rappel indique le pourcentage de documents pertinents qui ont été retrouvés par le SRI par rapport à l'ensemble des documents pertinents de la collection.

$$\text{rappel} = \frac{R_+}{R} \qquad (1)$$

2.2.3 Précision

Cette mesure calcule la capacité du SRI à retrouver uniquement les documents pertinents. La précision permet de mesurer la fraction des documents pertinents parmi ceux qui ont été retrouvés par le système.

$$\text{précision} = \frac{R_+}{M} \qquad (2)$$

Ou:

R: le nombre total de documents pertinents dans la collection.

M : le nombre de documents sélectionnés.

R+ : le nombre de documents pertinents sélectionnés.

Un SRI idéal est un système qui restitue tous les documents pertinents

(Rappel = 1), et tous les documents qu'il retrouve sont pertinents (précision =1) pour la requête de l'utilisateur. En pratique, cet idéal n'est jamais atteint puisque ces deux quantités évoluent en sens inverse.

Intuitivement, si on augmente le rappel en retrouvant plus de documents pertinents, et si on diminue la précision en retrouvant aussi plus de documents non pertinents. Inversement, une plus grande précision risque de rejeter des documents pertinents diminuant ainsi le rappel.

2.2.4 F-mesure

Une mesure populaire qui combine la précision et le rappel est leur pondération, nommée F-mesure ou F-score :

$$F = \frac{2 \cdot (\text{précision} \cdot \text{rappel})}{(\text{précision} + \text{rappel})} \qquad (3)$$

Ceci est connu comme mesureF1, car précision et rappel sont pondérés de façon égale. Il s'agit d'un cas particulier de la mesure générale Fβ (pour des valeurs réelles positives de β):

$$F_\beta = \frac{(1+\beta^2) \cdot (\text{précision} \cdot \text{rappel})}{(\beta^2 \cdot \text{précision} + \text{rappel})} \qquad (4)$$

3 Langage de programmation JAVA

Java est un langage de programmation récent (les premières versions datent de 1995) développé par Sun Microsystems. Il est fortement inspiré des langages C et C++. A l'instar de C++, Java fait partie de la « grande famille » des **langages orientés objets**. Mais Java a su se distinguer de ses frères pour beaucoup des raisons. Avant de présenter ces raisons voici un aperçu historique sur ce langage. Le langage Java trouve ses origines dans les années 1990. A cette époque, quelques ingénieurs (innovateurs) de SUN Microsystems commencent à parler d'un projet d'environnement indépendant du hardware pouvant facilement permettre la programmation d'appareils aussi variés que les téléviseurs, les magnétoscopes etc. James Grosling (SUN Microsystems) développe un premier langage, Oak, permettant de programmer dans cet environnement. En 1992, tout est prêt pour envahir le marché avec cette nouvelle technologie mais la tentative se solde par un échec. Bill Joy (co-fondateur de SUN Microsystems) sauve malgré tout le projet. Devant la montée en puissance de l'Internet, il lui semble intéressant d'insister sur l'élaboration d'un langage indépendant des plates-formes et des environnements (les principaux problèmes rencontrés sur l'Internet étant liés à l'hétérogénéité des machines et des logiciels utilisés). Dès lors tout s'accélère. Oak est renommé (en 1995) Java et soumis à la communauté Internet grandissante. Une machine virtuelle, un compilateur ainsi que de nombreuses spécifications sont données gratuitement et Java entame une conquête fulgurante. Aujourd'hui, après de nombreuses améliorations (parfois modifications) Java n'est plus uniquement une solution liée à l'Internet. De plus en plus de sociétés (ou de particuliers) utilisent ce langage pour l'ensemble de leurs développements (applications classiques, interfaces homme-machine, applications réseau…). Voici à présent les principales raisons qui ont fait le succès de Java.

- C'est un langage orienté objet dérivé du C, mais plus simple à utiliser et plus « pur » que le C++. Entendons par « pur » le fait qu'en Java, on ne peut faire que de la programmation orientée objet contrairement au C++ qui reste un langage hybride, c'est-à-dire autorisant plusieurs styles de programmation. C++ est hybride pour assurer une compatibilité avec le C.
- Il est doté, en standard, de bibliothèques de classes très riches comprenant la gestion des interfaces graphiques (fenêtres, boites de dialogue, contrôles, menus, graphisme), la programmation multithreads (multitâches), la gestion des exceptions, les accès aux fichiers et au réseau … L'utilisation de ces bibliothèques

facilitent grandement la tâche du programmeur lors de la construction d'applications complexes ;

- Il est doté, en standard, d'un mécanisme de gestions des erreurs (les exceptions) très utile et très performant. Ce mécanisme, inexistant en C, existe en C++ sous la forme d'une extension au langage beaucoup moins simple à utiliser qu'en Java ;
- Il est multi plates-formes : les programmes tournent sans modification sur tous les environnements où Java existe (Windows, Unix et Mac).
- Ce dernier point fait de Java un bon outil pour permettre l'interopérabilité.
- Java, un langage multi plates-formes qui doit être compilé et interprété. Dans une première phase on compile un programme (un ou plusieurs fichiers source *.java*) en fichiers *.class*. Le compilateur génère un fichier *.class* pour chacune des classes définies dans le(s) fichier(s) *.java*. L'ensemble des fichiers *.class* est ensuite interprété par la Machine Virtuelle Java (*Java Virtual Machine*) pour exécuter le programme (seule la JVM dépend du système). Ce principe donne à Java une portabilité maximale (Windows, Unix, MacIntosh …).

4 Implémentation de l'application

4.1 Editeur de programmation Netbeans

Est un environnement de développement intégré (EDI), placé en *open source* par Sun en juin 2000 sous licence CDDL (Common Développement and Distribution License) et GPLv2. En plus de Java, NetBeans permet également de supporter différents autres langages, comme C, C++, JavaScript, XML, Groovy, PHP et HTML de façon native ainsi que bien d'autres (comme Python ou Ruby) par l'ajout de greffons. Parmi plusieurs fonctionnalités on trouve :

- Complétion du Code.
- Coloration Syntaxique.
- Abréviations.
- Conviviale et personnalisable…etc.

4.2 Interfaces de l'application

La figure IV.2 présente la fenêtre principale de notre application dotée d'une barre de menu incluant les commandes d'exécution de différentes phases a savoir le prétraitement, le chargement et lemmatisation ainsi que l'indexation.

Figure IV.2 : Interface principale de l'application.

Ensuite, on passe à l'exécution de commandes de l'interface indexation illustrée dans la figure IV. 3.

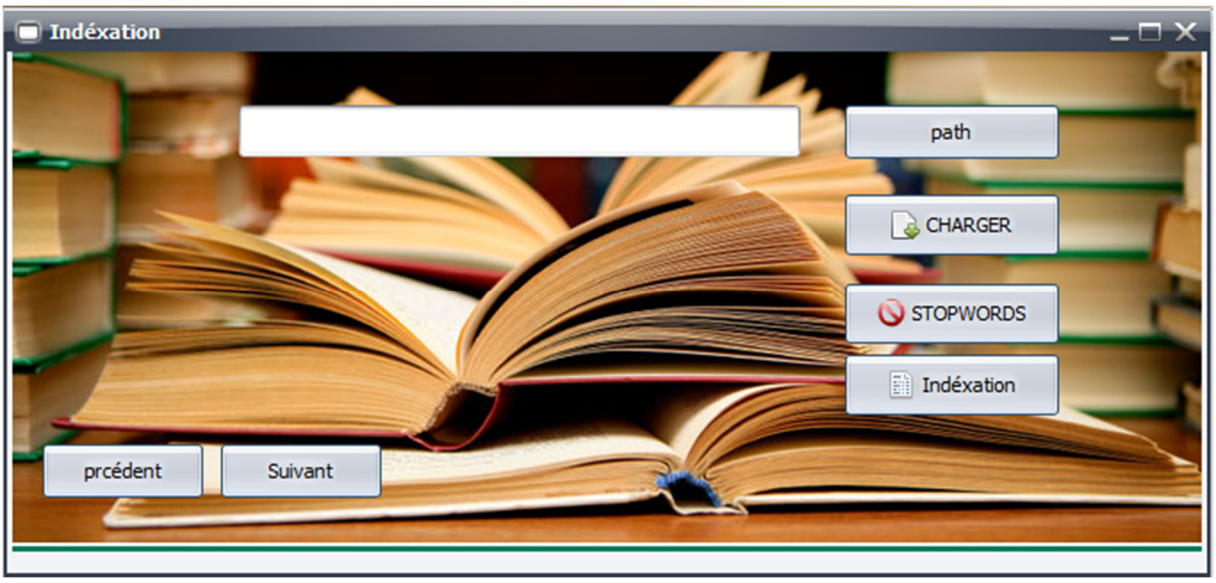

Figure IV. 3 : Interface Indexation

Chemin : L'utilisateur indique le chemin d'un corpus pour effectuer la recherche.

Charger : Ce bouton permet de remplir les contenus des fichiers du corpus.

Lemmatisation : Ce bouton a pour le but d'éliminer les mots vides et outils de l'ensemble de fichiers.

Indexation : Ce bouton sert à calculer le TF*IDF, et mettre le résultat de chaque mot et son score dans un fichier texte (index.txt).

Précédent : Ce bouton sert à revenir à la fenêtre d'accueil.

Suivant : Ce bouton transite à une nouvelle fenêtre pour exécuter la recherche.

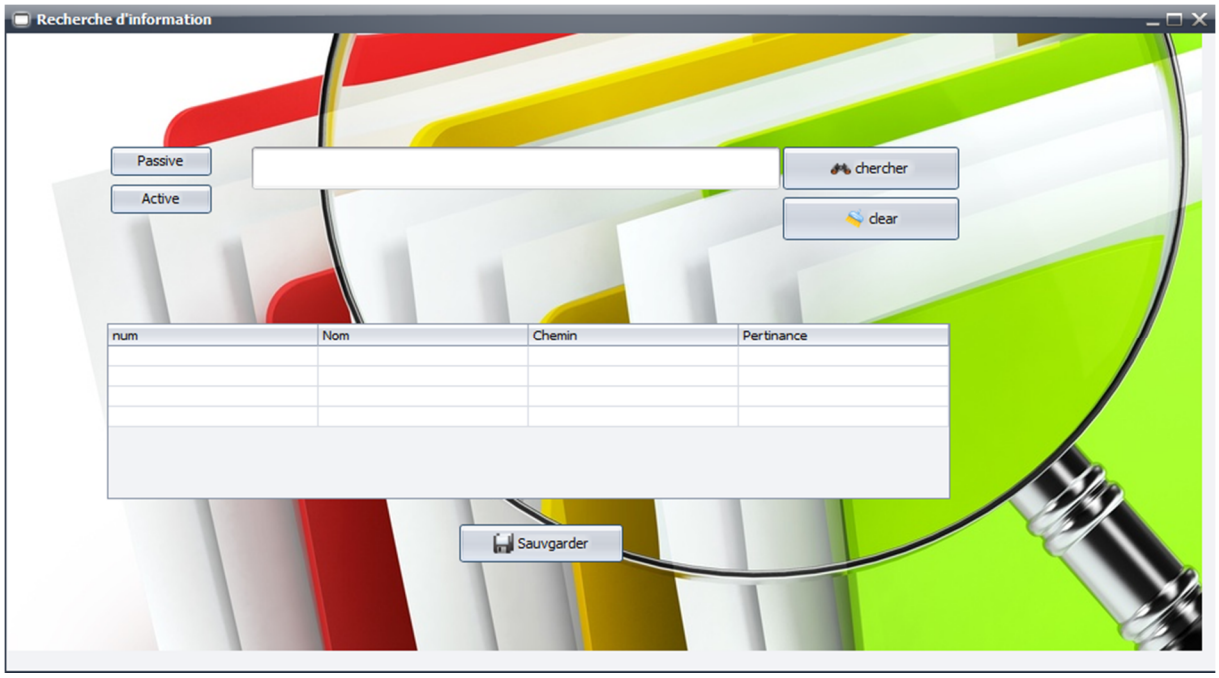

Figure IV. 4 : L**'**interface de la recherche d'information

La fenêtre recherche sert à :

Comparer les mots clés de la requête exprimée par l'utilisateur avec l'index en affichant le résultat de la recherche dans un tableau avec le nom, le chemin et la pertinence de chaque élément de corpus.

- **Chercher** : Ce bouton permet d'explorer la requête demandée par l'utilisateur.
- **Clear** : Bouton vider le champ de la requête.
- **Sauvegarder** : Ce bouton sert à enregistre le résultat dans une base de donnée pour des éventuelles recherches.

Fenêtre des résultats de quelques requêtes

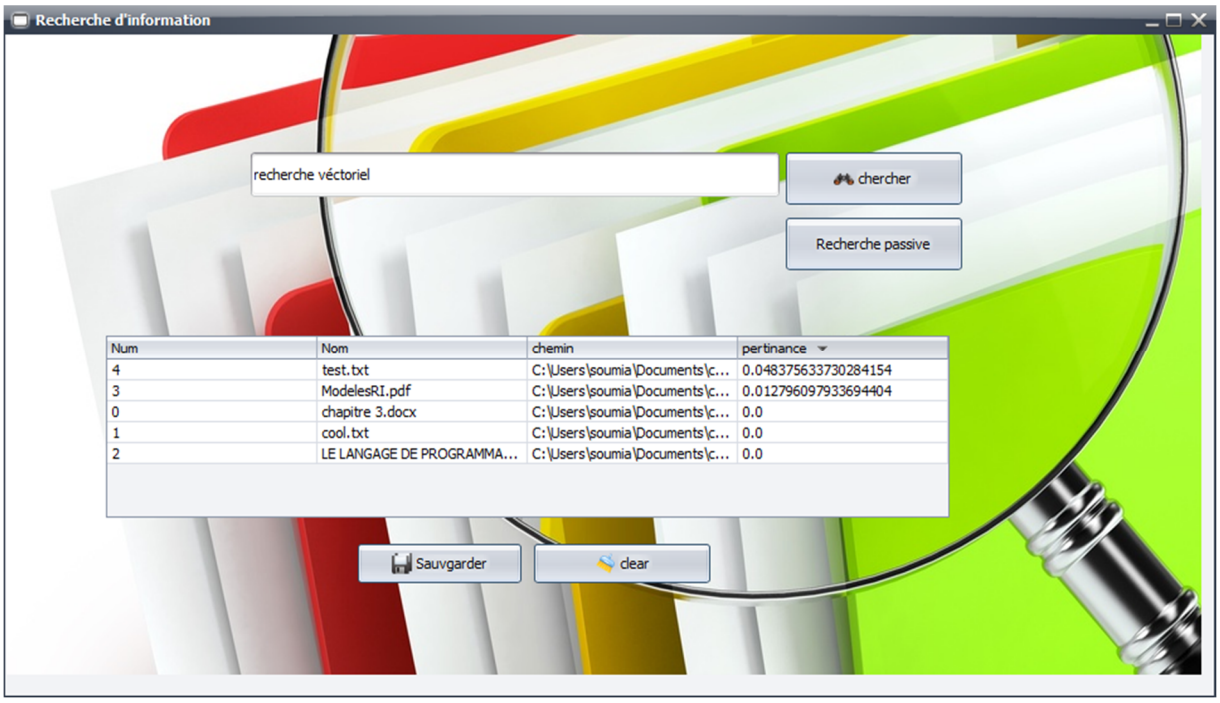

Figure IV.5 : Résultat d'une requête (recherche vectoriel).

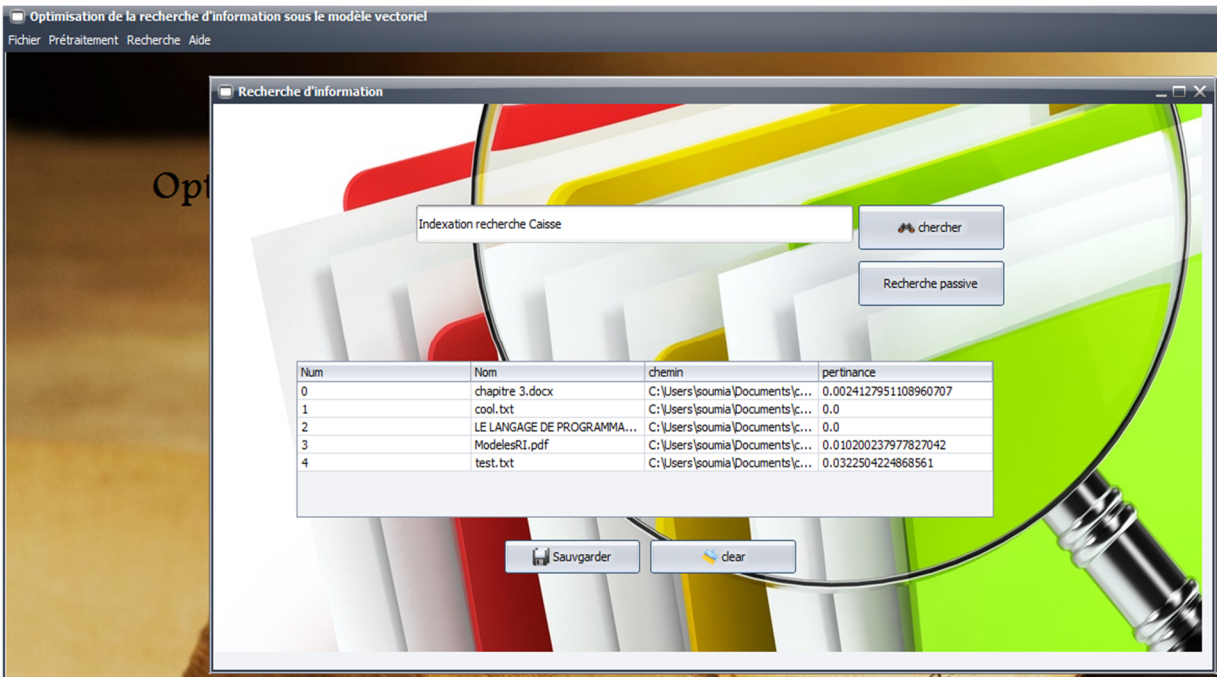

Figure IV.6 : Résultat d'un exemple de recherche (indexation recherche Caisse).

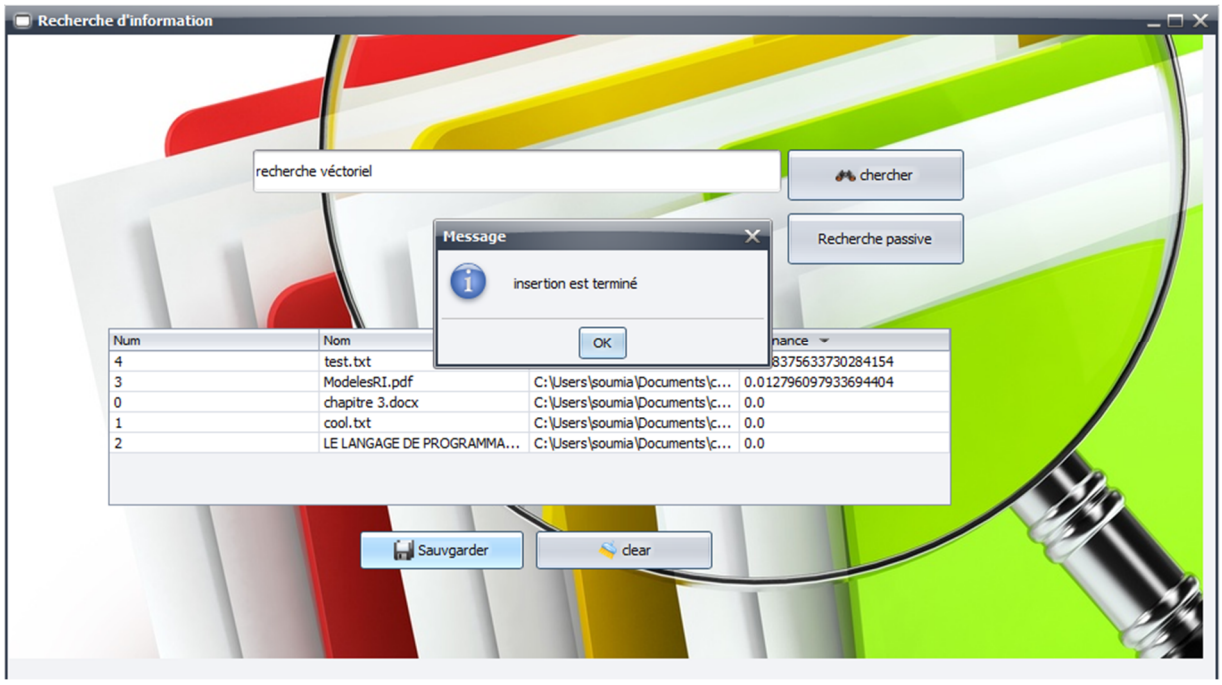

Figure IV.7 : Sauvegarde de résultat.

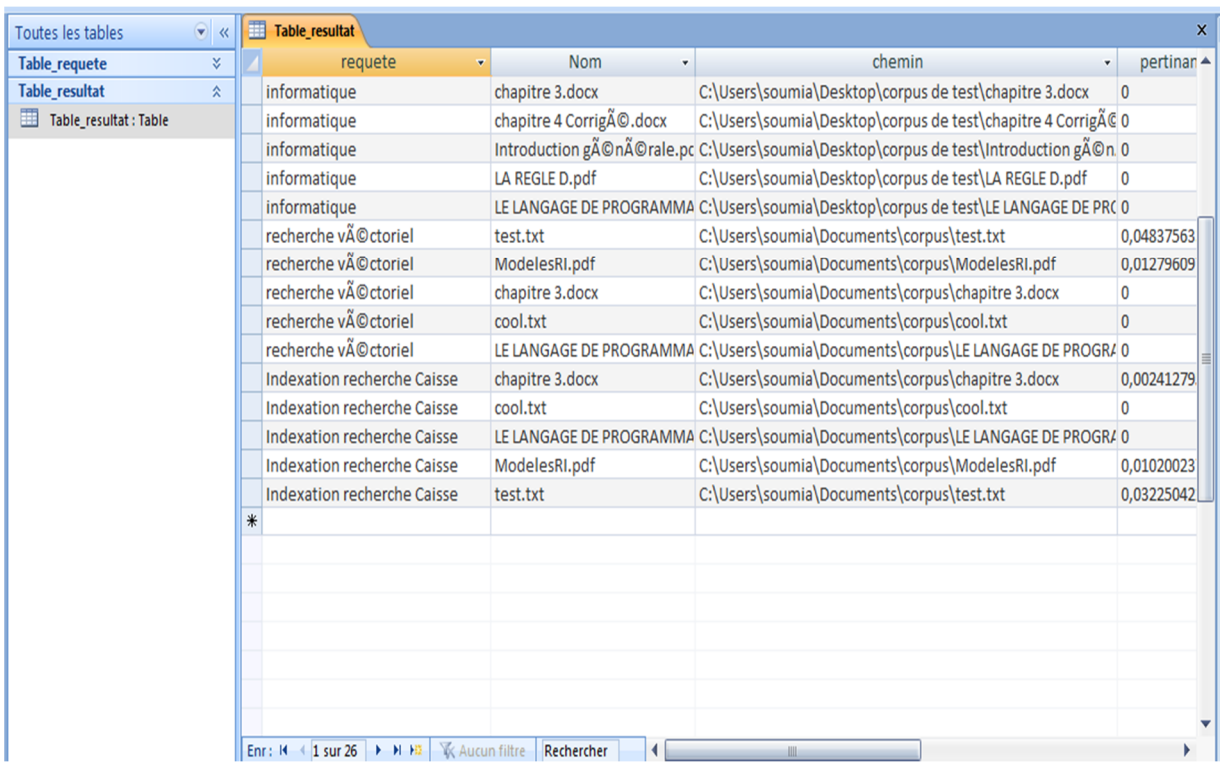

Figure IV.8: Résultat de recherche dans la base de donnée (Stockage_résultat)

5 Conclusion

Ce chapitre résume les travaux qui ont été faits pour la RI. Nous nous sommes essentiellement intéressés à compréhension des systèmes de recherche d'information

Comme nous avons présenté la conception et réalisation de notre application qui permet à l'utilisateur de faire une recherche décrivant son besoin, il lui affiche l'ensemble des documents pertinents comme un résultat retenu. .

Conclusion générale

Le monde assiste depuis ces dernières décennies, a une production massive d'informations dans tous les domaines d'intérêt à savoir retrouver une information dans un espace diversifié et de taille considérable. Dans ce contexte de travail, nous nous sommes intéressés à la mise en œuvre d'un modèle simple et très utilisé dans les systèmes de recherche d'information SRI basé essentiellement sur une technique d'appariement. *Ce* mécanisme *(l'appariement)* vise à attribuer des scores de pertinences aux éléments des documents. La majorité des approches de l'évaluation de pertinence sont fondées sur des systèmes d'indexation basés sur des mots d'un document et les mots de la requête qui sont représentés par une liste des mots clés pondérés. Notre travail se focalise en deux volets principaux, le premier consiste à implémenter intégralement le modèle vectoriel en sa version basique ce qui nous a permis de comprendre les SRI, de plus nous avons tenté de comprendre les autres modèles qui sont passé en revu en chapitre 2, le deuxième volet consiste à implémenté une astuce qui sert à garder l'historique de la requête de l'utilisateur, en d'autres termes simuler la requête comme étant un profil utilisateur à court ou moyen terme et ce dans le but de garder un certains nombres de documents pertinents relativement aux requêtes précédentes. La représentation de l'utilisateur à travers la notion de profil permet de mieux comprendre ses mécanismes cognitifs, notamment ceux permettant de percevoir le concept subjectif de la pertinence et au-delà, cibler ses besoins spécifiques dans le but d'améliorer la pertinence de l'information. Plusieurs perspectives peuvent être issues de ce travail notamment la comparaison entre le modèle vectoriel adopté et d'autres modèles tels que le modèle probabiliste ou le modèle basé sur les ensembles floues, ce qui demande une implémentation et une validation par des métriques usuelles. Or, l'introduction de l'information sémantique du profile et/ou document afin d'améliorer la qualité de recherche en rappel et en précision.

Bibliographie

[1] [Baeza-Yates and Ribeiro-Neto,]Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. 1999

[2] Baziz.M " indixation conceptuelle Guidée par ontologie pour la recherche d'information "thése de doctorat decembre 2005 pages 42-43.

[3] Bordogna et al., : Flexible Querying of Structured Documents. Proceedings of the Fourth International Conference on Flexible Query Answering Systems(FQAS), 2000.

[4] Boucham.S, Une approche basée Ontologies pour l'indexation automatique et la recherche d'information Multilingue, mémoire de magister, Université M'hamed Bougara Boumerdes, 2009.

[5] Bouramoul A " recherche d'information contextuelle et semaintique sur le web " thése pour l'optention du grade de docteur en science ,2011

[6] Caro, Budi Yuwono, Savio L.Y. Lam , Jerry H. Ying, Dik L. Lee, A World Wide Web Ressource Discovery System, 1997.

[7] Efthimiadis. Query Expansion : Annual Review on Technology. ARIST, Volume 31, pages :121-187, 1996.

[8] Elsa NEGRE CAHIER DE LAMSAD Université dauphine paris (Laboratoire d'Analyses et Modélisation de Systèmes pour l'Aide à la Décision) avril2013

[9] Fluhr et al , C. Buckley, Probabilistic document indexing from relevance feedback data. Proceedings of ACM SIGIR 90, pages : 45-61, 1990.

[10] [Jaccard] . Étude comparative de la distribution orale dans une portion des alpes et des jura. Bulletin del la Société Vaudoise des Sciences Naturelles avril 2013

[11] Hachemi.H,"Expansion-de-requete-pour-un-systeme-de-recherche-dinformation-par-croisement-de-langues".Mémoirede fin d'études Pour l'obtention du diplôme de Licence en Informatique 2013

[12] Harman D. Harman, Relevance feedback revisited. Proceedings of ACM SIGIR 92, pages : 125-132, 1992

[13] Holyoak et al., Holyoak, K. J. et Thagard, P. Analogical mapping by Constraint Satisfaction Congnitive Science ,13,p.295-355, these-Yael-Champclaux,1989

[14] Ingwersen, et al. Information retrieval interaction. London, Taylor Graham, 1992.

[15] Koczy et al., : Baranyi, P.; Gedeon, T.D.; Koczy, L.T.; Intelligent information retrieval using fuzzy approach. Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on Volume 2, 11-14 Oct. 1998 Page(s):1984 - 1989 vol.2 Digital Object Identifier 10.1109/ICSMC.1998.728188.

[16] Kohonen et al, Self-Organisation and Associative Memory3rd Edition, Springer Verlag, Berlin, pages : 80-89, 1989.

[17] *RimoucheH,*"Moteur_de_recherche_Semantique". Thése de licence juin 2013

[18] Massih-Reza amini université pierre et marie curie laboratoire d'informatique de parie 6 ACM Press, Addison Wesley, pages : 70-77,1999

[19] Nassr N ,"Croisement de langues en recherche d'information : traduction et Désambiguïsation de requêtes" these du Doctorat de l'UniversitéPaul Sabatier RI, Decembre 2002

[20] Ponte et al: Jay M. Ponte, W. Bruce Croft: A Language Modeling Approach to Information Retrieval. SIGIR 1998: 275-281

[21] Salton.G, Automatic text processing: The transfromation, analysis and retrieval of information by computer. Addison-Wesley publishing, pages : 85-92, 1989.

[22] *Xavier .T* "Modèles de recherche et Evaluation" ,thése de Master d'université paris-sud .2000

[23] Xavier Tannier modèle de RI et évaluation mémoire de master 2(Indexation et Recherche d'Information) université paris-sud11

[24] Porter, M. An algorithm for suffix stopping. Program, P130_137. 1980

[25] Voorhees E.M. TREC Overview. In the proceedings of the seventh Text retrieval Conference TREC 7, pages : 58-66,1999.

[26] Zadeh, L. A. Fuzzy sets as a basis for a theory of possibility. Fuzzy Sets and Systems 1 (1978), 3–28.

[27] Yates et al, Modern information Retrieval ACM Press ,Addision Wesley, pages 70-77,1999