Remerciements

Je tiens à remercier avant tout, Dieu de nous a prodiguée la force morale et physique et nous a permis d'achever ce travail.
Je tiens à remercier mes parents qui nous ont donné la force et le courage, pour accomplir ce travail, merci ma chère mère et merci mon cher père pour tous.
Je tiens tout d'abord à remercier* Mme Ahlem Kenniche *pour avoir encadré et dirigé mon projet. Je la remercie pour la confiance qu'il a bien voulu m'accorder, ses conseils et remarques constructives qui m'ont permis d'améliorer la qualité de mon travail. Qu'il soit ici assuré de mon très respect.
J'adresse mes plus sincères remerciements aux membres du jury d'avoir accepté d'examiner ce modeste travail.
Mes remerciements vont également à tous les enseignants du département d'informatique que nous respectons beaucoup.

Enfin, je souhaiterais adresser des remerciements plus particuliers à toute ma famille.

Je tiens de remercier aussi mes très chère amies Ghamnia Leila et Boudraf Khadidja qui m'a vraiment aidé afin de réaliser ce modeste travail.

DEDICACES

À mes très chers parents, Que Dieu les gardent.

À tous mes frères : AbdelRahmen et Rayane.
À tous mes sœurs : Amina, NourElhoudda et Douaà.

À mes cousines : Khadidja, Nadia, Amina et Hiba.
À mon cousin : Ben youcef.

À toute mes amies Leila, Ibtissem, khadidja, Karima et Sarrah.

À tous ceux qui sont proches de mon coeur.
Et dont je n'ai pas cité les noms.
Je dédie ce modeste travail.

Asmaà

Résumé

La recherche d'information par mots clés est une méthode simple et intuitive utilisée aussi bien sur le web que dans des recherches documentaires. Elle n'est cependant pas adaptée à certains besoins. En effet, de nombreuses sources de données telles que les forums de discussion, les articles de journaux et les articles de Wikipédia, portent sur un sujet particulier et sont écrites autour d'entités nommées (personne, pays, …).

Pour ce type de sources, il est pertinent d'interpréter les requêtes des utilisateurs en utilisant les entités qu'elles contiennent et d'organiser les résultats des requêtes par entités que de retourner une liste ordonnée de documents. Dans ce travail nous nous intéressons à cette problématique qui est également motivé par la disponibilité de corpus de documents et par l'apparition des systèmes d'annotation automatique.

Mots-clés : recherche d'information, indexation de corpus, recherche d'entités, recherches diversifiées.

Abstract

Information retrieval by keywords is a simple and intuitive method used on the web search and in documentary research. However, it is not adapted to some needs. In fact, many data sources such as forums, news groups and Wikipedia articles, focus on a particular category and are written about named entities (people, country, organization...).

For this type of sources, it is appropriate to interpret the queries of the users by using the entities that they contain and to organize the results of the queries by entities that to return an ordered list of documents. In this work we are interested in this issue, which is also motivated by the availability of documents and the appearance of automatic annotation systems.

Keywords: information retrieval, corpus indexing, entity retrieval, diversified search.

Table des matières

Liste de figure

Liste des abréviations

RI	**R**echerche d'**I**nformation.
SRI	**S**ystème de **R**echerche d'**I**nformation.
RE	**R**echerche d'**E**ntité.
R1E	**R**echerche par une seule **E**ntité.
RPE	**R**echerche par **P**lusieurs **E**ntités.
RMC	**R**echerche par **M**ot **C**lé.
EI	**E**xtraction d'**I**nformation.
REN	**R**econnaissance des **E**ntités **N**ommées.
QA	**Q**uestion **A**nswering (Question /réponse).
Url	**U**niform **R**esource **L**ocator.
TF	**T**erm **F**requency.
IDF	**I**nverse **D**ocument **F**requency
TAL	**T**raitement **A**utomatique du **L**angage.
NER	**N**amed **E**ntity **R**ecognizer.
KI	**K**ey word **I**ndex.
EI	**E**ntity **I**ndex.
DI	**D**ocument **I**ndex.

Introduction Générale

Tout le monde a besoin d'information dans sa vie quotidienne. Nous avons besoin de l'information pour prendre les meilleures décisions possibles. Dans chacune de nos activités personnelles, les décisions sont requises et l'information est nécessaire pour soutenir ces décisions. L'information est nécessaire dans presque tous les domaines de la pensée et l'action humaine.

La recherche d'information (RI) est aujourd'hui une activité d'une grande importance. Il faut pouvoir, parmi le volume important de documents disponibles, trouver ceux qui correspondent au mieux à nos besoins en un minimum de temps. L'opération de la RI est réalisée par des outils informatiques appelés Systèmes de Recherche d'Information (SRI). [14] Le but principal d'un SRI est de retrouver les documents pertinents en réponse à une requête utilisateur. Ces documents sont typiquement retournés sous forme d'une liste ordonnée, où l'ordre est basé sur des estimations de pertinence.

De par sa croissance et son développement, le web représente aujourd'hui une source importante de données hétérogènes (news, articles, photos, vidéos...). Les informations y sont stockées sous forme de documents identifiés d'une manière unique par des Urls et reliés entre eux par des liens hypertextes [16]. Dans certain cas l'utilisateur cherche réellement dans le web, pas des pages web ou des documents mais des informations que celles-ci contient, c'est-à-dire des entités (personne, pays, etc). Pour ce type de sources, il est plus pertinent d'interpréter les requêtes des utilisateurs en utilisant les entités qu'elles contiennent et d'organiser les résultats des requêtes par entités que de retourner une liste ordonnée de document.

D'une manière générale, lorsque l'utilisateur veut trouver une liste d'entités, par exemple des « politiciens Algériens », il est facile pour un moteur de recherche classique de retourner les documents politiques. Il est laissé à l'utilisateur d'extraire les informations sur les entités demandées à partir des résultats fournis. Notre proposition est d'interpréter les termes d'une requête formée de mots clés ou d'entités par des entités relatives. Notre objectif est de proposer une approche qui retourne des entités pertinentes aux requêtes.

Dans notre travail, nous considérons le problème de la recherche des entités et des documents les contenant en réponse aux requêtes des utilisateurs. Les entités recherchées pouvant être connues ou inconnues aux utilisateurs, Cela signifie qu'il existe différents choix pour poser sa requête : recherche par une seule entité (R1E), recherche par plusieurs entités (RPE), recherche par mots clés (RMC). [5]

Le travail réalisé dans ce mémoire s'inscrit dans ce contexte particulier. Son objectif est de construire un système de recherche d'entités. Il s'agit de retourner des documents pertinents et organisés par entité.

Ce mémoire est présenté comme suit :

Le premier chapitre que nous avons appelé **Concept de base de la recherche d'information** présente les concepts de base de la RI, les différents modèles pour effectuer ce traitement et décrit en générale le processus de la RI.

Le deuxième chapitre appelée **La recherche d'entité : L'état de l'art** aborde les Principaux concepts de la recherche d'entité et un exemple de motivation qui exprime notre problématique.

Le troisième chapitre appelé **conception de notre système de recherche d'information** détailles notre travail et donne une architecture générale et quelques algorithmes que nous avons appliqué pour l'implémentation de notre système.

Le quatrième chapitre appelé **implémentation et mise en oeuvre** expose toute les interfaces et les fonctionnalités de notre système de recherche d'information.

Nous terminons ce mémoire par une conclusion.

CHAPITRE I

Concept de base de la recherche d'information

I-Concept de base de la recherche d'information

I.1 introduction

La recherche d'information RI est le processus par lequel les informations (ou les documents qui les contiennent) sont stockées et mises à la disposition des utilisateurs et par conséquent, leur récupération soient pertinente aux besoins des utilisateurs. Les systèmes de recherche d'information SRI sont conçus pour faciliter l'accès aux informations stockées. En outre, ces systèmes sont concernés par la représentation, le stockage et l'organisation d'informations. Le système RI exploite un ensemble d'informations et de requêtes et infère par un mécanisme permettant de déterminer les informations les plus pertinentes à une requête.

I.2. La recherche d'information

La recherche d'information (RI) est le domaine qui étudie la manière de répondre pertinemment à une requête en retrouvant de l'information dans un corpus. Elle se définie généralement par l'identification de documents qui satisfont le mieux le besoin en informations d'un utilisateur. Le corpus est composé de documents d'une ou plusieurs bases de données, qui sont décrits par un contenu ou les métadonnées associées. [1] Le but de la RI est de trouver seulement les documents pertinents. La notion de pertinence est très complexe. De façon générale, dans un document pertinent, l'utilisateur doit pouvoir trouver les informations dont il a besoin. C'est sur cette notion de pertinence que le système doit juger si un document doit être donner à l'utilisateur comme réponse. [6]

L'essor du web a remis la RI face à de nouveaux défis d'accès à l'information, il s'agit cette fois de retrouver une information pertinente dans un espace diversifié et de taille considérable. Ces difficultés ont donné naissance à une nouvelle discipline appelée Recherche d'Information sur le Web. [17]

I.2.1. Recherche dans le web

L'objectif de la recherche d'information dans le web est de satisfaire les besoins des utilisateurs en information. Selon la taxonomie présentée dans [4] les utilisateurs effectuent leurs recherches de plusieurs manières : de manière Navigationnelle (atteindre un site particulier), de manière Informationnelle (acquérir certaines informations présumées être présentes sur une ou plusieurs pages Web), de manière Transactionnelle (effectuer une certaine activité web-mediated par exemple : téléchargement des fichiers, l'accès à une base de données, … etc).

I.2.1.1. Les requêtes navigationnelles

Le but de ces requêtes est d'atteindre un site particulier par saisie directe de l'url ou par parcours manuel ou automatique des liens hypertextes entre sites. Ce mode de recherche de documents du web nécessite la connaissance d'un minimum d'urls pertinentes et intéressantes pour la recherche à effectuer. Or, la taille actuelle du web ne permet pas de constituer cette connaissance. L'utilisation des moteurs de recherche représente une solution intéressante au problème soulevé par le mode de recherche précédent. Le principe consiste à décrire les documents cibles par une requête de mots clés. Après évaluation sur les documents de son index, le moteur de recherche renvoie une liste d'urls (des réponses) jugées pertinentes par rapport à la requête soumise. [5]

I.2.1.2. Les requêtes transactionnelles

Le but de ces requêtes est d'atteindre un site où des intéractions vont se passer. Ces intéractions constituent des transactions définies par ces requêtes. Les principales catégories de ces requêtes sont le shopping, le téléchargement de fichiers (images, vidéo, etc.), l'accès à certaines bases de données (par exemple, les pages jaunes), la recherche des serveurs (par exemple, les jeux), etc. Le résultat de ces requêtes est difficile à évaluer, seul le jugement binaire est possible pour savoir si les résultats sont appropriés ou non appropriés. Cependant, la plupart des informations obtenues (par exemple, prix des marchandises, etc.) ne sont pas fournies en utilisant les moteurs de recherche. [5]

I.2.1.3. Les requêtes informationnelles

Les requêtes informationnelles sont les plus proches aux requêtes classiques de la RI. Leur but est d'acquérir des informations supposées être présentes sur une ou plusieurs pages web dans une forme statique. Aucune intéraction n'est prévue sauf la lecture et aucun document n'est créé en réponse à la requête de l'utilisateur. Néanmoins, les moteurs de recherche pourraient conduire à des pages dynamiques. Pour les requêtes informationnelles du web, près de 15 % de toutes les recherches effectuées ont comme résultat une bonne collection de liens portant sur le sujet, plutôt qu'un bon document. [5]

I.2.2 Système de recherche d'informations

Un Système de Recherche d'Informations (SRI) est un système informatique qui permet de retourner à partir d'un ensemble de documents, ceux dont le contenu correspond le mieux à un besoin en informations d'un utilisateur, exprimé à l'aide d'une requête.

Un SRI inclut un ensemble de procédures et d'opérations qui permettent la gestion, le stockage, l'interrogation, la recherche, la sélection et la représentation de cette masse d'informations. [2]

I.2.2.1 Processus de recherche d'information

Les différentes étapes du processus de RI, sont représentées schématiquement par le processus en U dans la Figure 1. La figure illustre particulièrement :

- Les notions de documents et de requêtes qui sont des conteneurs d'informations.

- Les opérations d'analyse, d'indexation et d'appariement qui permettent globalement de traiter la requête dans le but de sélectionner des documents à présenter à l'utilisateur. [2]

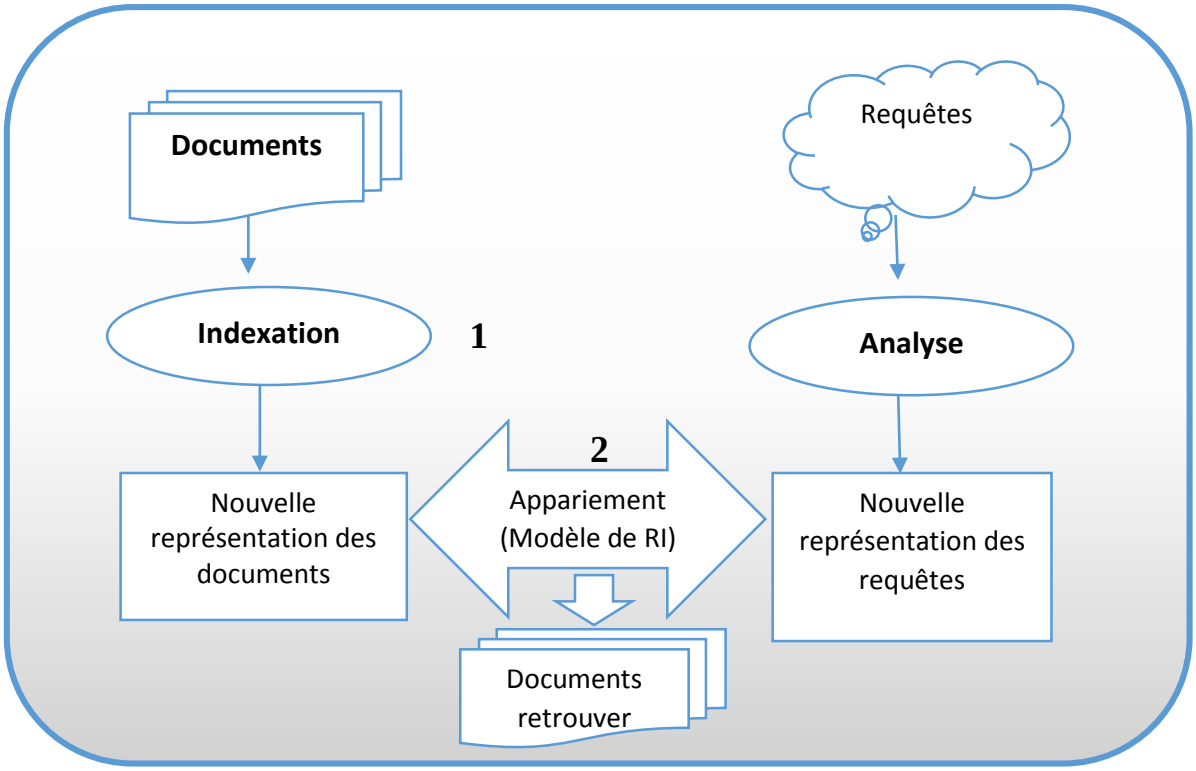


Figure 1. Processus en U d'un système de recherche d'information.

I.2.2.2 Principales phases du processus de RI

L'objectif fondamental d'un processus de RI est de sélectionner les documents « les plus proches » du besoin en information de l'utilisateur décrit par une requête. Ceci induit deux principales phases dans le déroulement du processus : l'indexation et la recherche (appariement requête/documents) [2] voir figure 1.

1-L'étape d'indexation se base sur l'analyse des documents et des requêtes afin de créer une représentation de leur contenu textuel qui soit utilisable par le SRI. Chaque document (et requête) est alors associé à un descripteur représenté par l'ensemble des termes d'indexation extraits.

2-La phase de recherche a pour objectif d'apparier les documents et la requête de l'utilisateur en comparant leurs descripteurs respectifs. Elle se base sur un formalisme précis défini par un modèle de RI. Les documents présentés en résultat à l'utilisateur, et considérés comme les plus pertinents, sont ceux dont les termes d'indexation sont les plus proches de ceux de la requête. [3]

I.2.3. Les modèles de recherche d'information

Comme nous l'avons vu, le but d'un SRI demeure dans sa capacité à établir une correspondance entre un document et une requête. De nombreux modèles ont été proposés en RI, ils sont généralement regroupés autour des trois familles. La figure 2 représente une classification détaillée des modèles de RI.

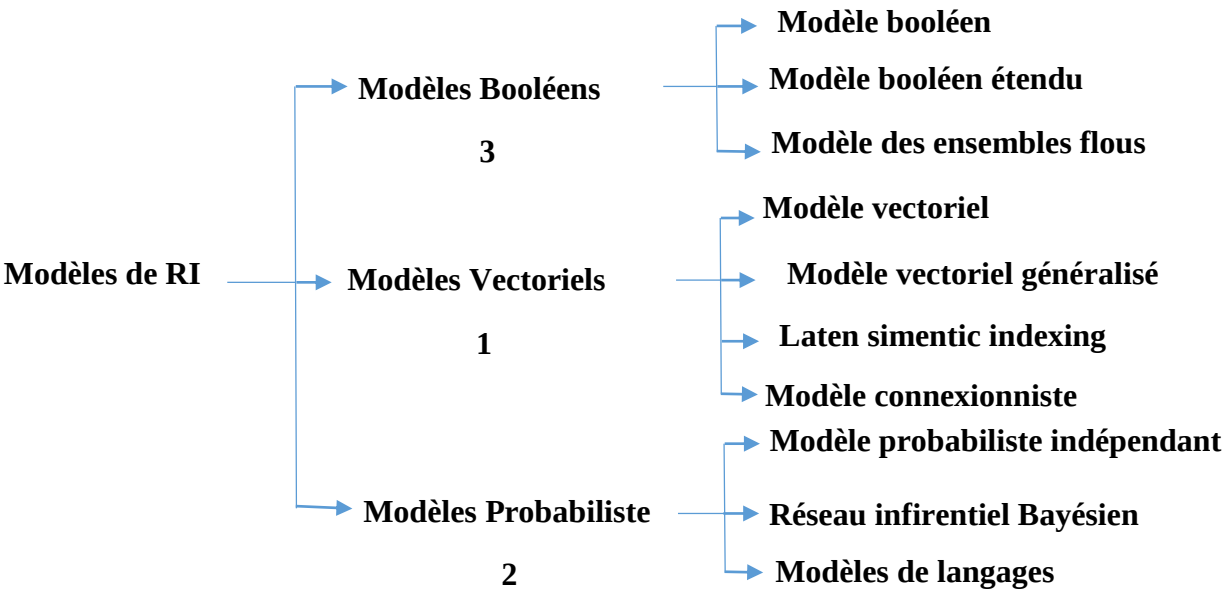


Figure 2. Taxonomie des modelés en recherche d'information. [7]

• les modèles booléens qui considèrent le processus de recherche comme une succession d'opérations à effectuer sur des ensembles d'unités lexicales contenues dans les documents.

• les modèles vectoriels au sein desquels la pertinence d'un document par rapport à une requête est envisagée à partir de mesures de distance dans un espace vectoriel.

• les modèles probabilistes qui représentent la RI comme un processus incertain et imprécis où la notion de pertinence peut être vue comme une probabilité de pertinence. [3]

I.2.3.1 Modèle vectoriel

Couramment employé en RI, Les modèles vectoriels sont des modèles algébriques. Les documents et requêtes sont représentés par des vecteurs de poids dans un espace vectoriel composé de tous les termes d'indexation. La pertinence d'un document vis à vis d'une requête est définie par des mesures de distances entre vecteurs. [7]

I.2.3.2 Modèle probabiliste

Les modèles probabilistes s'appuient sur la théorie des probabilités. La pertinence d'un document vis à vis d'une requête est vue comme une probabilité de pertinence document/requête. [7]

I.2.3.3 Modèle booléen

Les modèles booléens sont les modèles les plus ancien et également le plus simple en RI. Un document est représenté par l'ensemble d'unités lexicales qu'il contient. Une requête est représentée comme une formule logique portant sur la présence ou l'absence d'unités lexicales reliées par des connecteurs (le ou ∨, le et ∧, le non ¬). [3]

I.3 L'extraction d'information

L'Extraction d'Information consiste en une opération de repérage et de structuration en classes sémantiques, par classification consécutive ou simultanée, d'éléments informatifs spécifiques présents dans des données non structurées, notamment textuelles, menée dans le but de donner à l'information une forme adéquate pour des traitements automatiques [18]. En effet, il faut tenir compte du fait que les données textuelles contiennent souvent de l'information non structurée, ce qui rend l'extraction d'information plus complexe.

L'extraction d'information (EI) consiste à analyser des textes pour en obtenir des informations. L'extraction d'information ne cherche pas à comprendre les textes dans leur ensemble, elle fait la recherche d'une information spécifique et extrait d'un texte donné des éléments pertinents. Il s'agit ainsi d'identifier des occurrences d'évènements particuliers.

Un autre axe de l'extraction d'information dans des corpus documentaires. La reconnaissance des entités nommées (REN) consiste à rechercher des objets textuels (c'est à dire un mot, ou un groupe de mots) catégorisables dans des classes telles que noms de personnes, noms d'organisations ou d'entreprises, noms de lieux, dates… etc.

I.3.1 La reconnaissance des entités nommées

Les travaux menés en traitement automatique des langues (TAL) ont porté une attention particulière aux noms propres de personnes, de lieux et d'organisations, appelée des entités nommées. Ces éléments semblant être utiles à diverses tâches comme, par exemple, la recherche d'information. [20]

La reconnaissance des entités nommées est une sous tache de l'extraction d'informations qui prend en entrée un bloc de texte non annoté et produit un bloc de texte annoté contenant les entités nommées trouvées. Chaque entité reçoit une étiquette en fonction de son type sémantique.

La reconnaissance des entités nommées consiste à :

- Identifier des unités lexicales dans un texte.
- Les catégoriser ;
- Eventuellement, les normaliser. [5]

La plupart des systèmes de REN utilisent soit des approches orientées connaissances soit des approches orientées données. Les systèmes orientés connaissances sont fondés sur des lexiques (listes de prénom, de pays, etc.) et sur un ensemble de règles de réécriture. D'un autre côté, les systèmes orientés données sont basés sur un modèle appris à partir d'un corpus préalablement annoté. [19]

I.4 L'annotation sémantique

Les annotations sont des commentaires, des notes, des explications ou d'autres types de remarques externes qui peuvent être associées à un document Web ou à une partie sélectionnée d'un document. Comme ils sont externes, il est possible d'annoter n'importe quel document Web de manière indépendante, sans avoir besoin de modifier le document lui-même. D'un point de vue technique, les annotations sont généralement considérées comme des métadonnées, car elles fournissent des informations supplémentaires sur une donnée existante. [10]

Dans le cadre du web sémantique « une annotation est une commentaire, une note, une explication ou toute autre remarque qui peut être rattachée à un document web ou une partie de celui-ci »[1].

[1]Definition du W3C.

I.5 L'indexation

L'indexation a pour rôle de représenter un document ou une requête en utilisent les informations qu'il contient par un ensemble de descripteurs, appelés aussi mots clé. Ces descripteurs constituent une facilité d'exploitation des documents étant donné que ceux-ci sont sous forme de textes libres [8]. L'indexation peut être faite d'une manière manuelle, d'une manière automatique ou bien d'une manière semi-automatique.

I.5.1 Technique d'indexation

I.5.1.1 Manuelle

En indexation manuelle, c'est un opérateur humain, généralement expert du domaine, qui se charge de caractériser, selon ses connaissances propres, le contenu sémantique d'un document. Cette approche présente deux inconvénients :

1. elle est subjective, puisque le choix des termes d'indexation dépend de l'indexeur et de ses connaissances du domaine,

2. elle est pratiquement inapplicable aux corpus de textes volumineux.

Néanmoins, elle est plus performante que l'indexation automatique en termes de précision moyenne des documents retrouvés en réponse à une requête utilisateur donnée. [7]

I.5.1.2 Automatique

L'indexation automatique, c'est un processus complètement automatisé qui se charge d'extraire les termes caractéristiques du document. L'intérêt d'une telle approche réside dans sa capacité à traiter les textes nettement plus rapidement que l'approche précédente, et de ce fait, elle est particulièrement adaptée aux corpus volumineux. L'indexation automatique est l'approche la plus étudiée en RI. [7]

En général l'indexation automatique se fait en plusieurs étapes :

- **Analyse lexicale :** (Tokenization en anglais)
  Consiste à découper le document en unités lexicales. Chaque unité lexicale est une séquence de caractères entourée par des séparateurs d'unités.

- **L'élimination des mots vides :**
  Dans cette étape les mots d'usage général et grammatical (les mots vides) sont éliminés. Du fait que ces mots apparaissent d'une manière uniforme dans les documents, ils sont non utiles pour l'indexation et ils doivent être éliminés. On distingue deux techniques pour l'élimination des mots vides : l'utilisation des stop list ou des anti-dictionnaires et l'utilisation des mesures statistiques.

- **La lemmatisation :** (stemming en anglais)
  Consiste à prendre la forme canonique du mot. Dans le document les mots peuvent apparaître sous différentes formes. Par exemple, citer, citation, citations, etc. La lemmatisation permet de substituer chaque mot par sa racine (lemme). La racine d'un mot est soit la forme infinitive si le mot est un verbe, soit la forme singulier masculin si le mot est un nom. L'utilisation de la lemmatisation contribue à l'amélioration des performances des SRI.

- **L'analyse syntaxique et morphosyntaxique :**
  L'objectif de cette étape est de repérer des mots composés. Cette analyse syntaxique se base sur des patrons (Template) pour extraire les mots composés. Dans ce processus une catégorie grammaticale est associée à chaque mot (ou groupe de mots) et des patrons sont manuellement construits. Ces patrons sont ensuite projetés dans les documents afin de détecter les séquences qui satisfont ces patrons syntaxiques. Nous signalons que dans la littérature il existe d'autres méthodes dites méthodes statistiques qui sont utilisées pour l'extraction des mots composés. Ces méthodes utilisent des mesures statistiques et n'utilisent pas d'analyse linguistique. [9]

I.5.1.3 Semi-automatique

L'indexation semi-automatique, appelée aussi indexation supervisée, est une combinaison des deux approches d'indexation précédentes. Dans ce cas, les indexeurs utilisent un vocabulaire contrôlé sous forme de thésaurus ou de base terminologique. Le choix final des termes d'indexation à partir du vocabulaire fourni, est laissé ainsi à l'indexeur humain (généralement spécialiste du domaine). [7]

I.5.2 Indexation par Lucene

Lucene est une bibliothèque open source écrite en Java qui permet d'indexer et de chercher du texte. Il est utilisé dans certains moteurs de recherche.

Lucene est capable de traiter de grands volumes de documents grâce à sa puissance et à sa rapidité dues à l'indexation, La Figure 3 décrit le processus d'indexation d'un document avec lucene, il se compose de trois phases [1] :

- La phase d'encapsulation d'un document dont la classe Parsers le transforme sous format d'un objet Document.
- L'analyse s'applique au Document à travers l'analyseur souhaité.
- La création d'index est réalisée par IndexWriter suivant l'emplacement choisi par Directory.

Procédure d'indexation est l'une des fonctionnalités de base fournies par Lucene. Le diagramme suivant (figure3) illustre le processus d'indexation et de l'utilisation des classes. IndexWriter est le composant le plus important et le noyau du processus d'indexation.

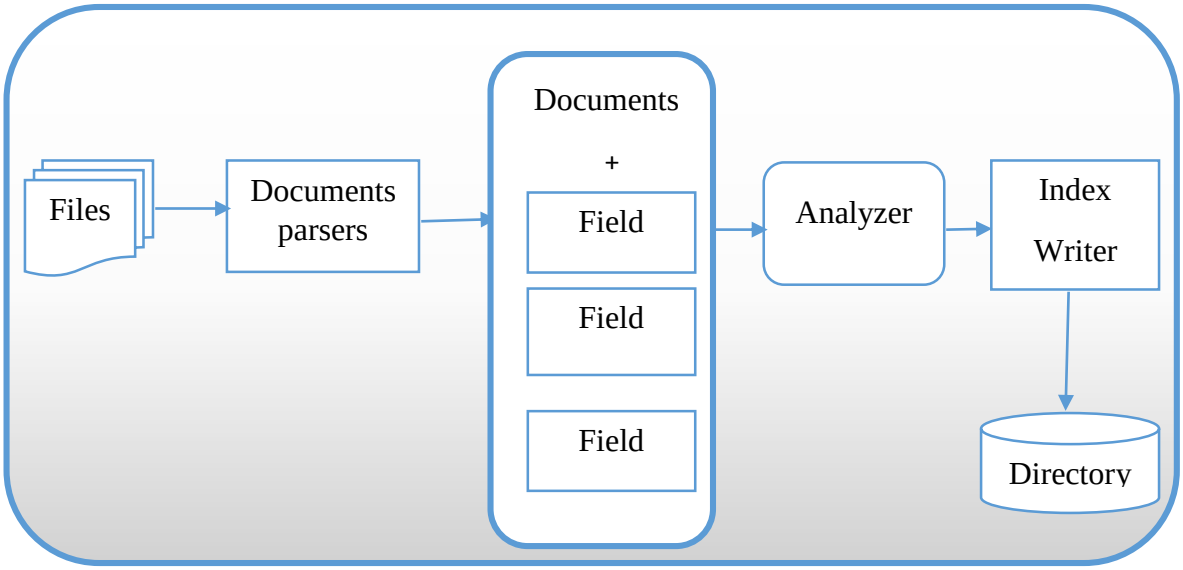


Figure 3. Processus d'indexation de Lucene. [1]

Nous ajoutons le Document (s) contenant le Field (s) à IndexWriter qui analyse le Document (s) en utilisant l'Analyzer puis crée index ouvert, modifier au besoin et stocker, les mettre à jour dans un Directory. IndexWriter est utilisé pour mettre à jour ou créer des index. Il ne sert pas à lire les index. [24]

- **IndexWriter :** La classe IndexWriter est le composant central du processus d'indexation. Cette classe crée un nouvel index et ajoute des documents à un index existant. On peut se la représenter comme un objet par lequel on peut écrire dans l'index mais qui ne permet pas de le lire ou de le rechercher.

- **Directory** : La classe Directory représente l'emplacement de l'index de Lucene. IndexWriter utilise une des implémentations de Directory, FSDirectory, pour créer son index dans un répertoire dans le Système de fichiers. Une autre implémentation, RAMDirectory, prend toutes ses données en mémoire. Cela peut être utile pour de plus petits indices qui peuvent être pleinement chargés en mémoire et peuvent être détruits sur la fin d'une application.

- **Analyzer** : Avant que le texte soit dans l'index, il passe par l'Analyseur. Celui-ci est une classe abstraite qui est utilisée pour extraire les mots importants pour l'index et supprime le reste. Cette classe tient une part importante dans Lucene et peut être utilisée pour faire bien plus qu'un simple filtre d'entrée.

- **Document** : La classe Document représente un rassemblement de champs. Les champs d'un document représentent le document ou les métadonnées associées avec ce document. La source originelle (comme des enregistrements d'une base de données, un document Word, un chapitre d'un livre, etc.) est hors de propos pour Lucene. Les métadonnées comme l'auteur, le titre, le sujet, la date, etc. sont indexées et stockées séparément comme des champs d'un document.

- **Field** : Chaque document est un index contenant un ou plusieurs champs, inséré dans une classe Field. Chaque champ (Field**)** correspond à une portion de donnée qui est interrogé intitulé ou récupéré depuis l'index durant la recherche.

I.6 Conclusion

Ce premier chapitre a porté essentiellement sur l'étude de recherche d'information de manière générale, nous avons présenté l'architecture des systèmes de recherche d'information et les principales phases du processus de recherche de plus nous avons présenté l'extraction d'information notamment la reconnaissance des entités nommées et ensuite nous avons passé à l'annotation sémantique, et enfin nous avons présentés l'indexation avec un exemple d'indexation avec lucene.

CHAPITRE II

La recherche d'entité : L'état de l'art

II.1 introduction

Le concept de la recherche d'entités a pour but d'exploiter la richesse du web afin d'en tirer les données enfouies dans les pages non structurées. La recherche d'entités deviendra une des meilleures techniques d'exploitation du contenu du web. Dans ce qui suit, nous présentons un exemple représentatif de ce qui est considéré comme entité.

Dans ce chapitre, nous présentons les travaux les plus importants sur la recherche d'entités. La section suivante portera sur la recherche d'entités, nous avons jugé qu'il était nécessaire de détailler mieux ce que c'est une entité avant de présenter les travaux relatifs à la recherche d'entités.

II.2 La recherche d'entité

La recherche d'entités sur le Web est un nouveau groupe de recherche qui va au-delà de la recherche d'un document classique. Alors que pour les tâches de recherche d'information la recherche de document peut donner des résultats satisfaisants pour l'utilisateur, différentes approches doivent être suivies lorsque l'utilisateur doit rechercher des entités spécifiques. Par exemple (voir la figure 4.), lorsque l'utilisateur veut trouver une liste des "femmes politiques européens" il est facile pour un moteur de recherche classique de retourner des documents sur la politique en Europe, et c'est à l'utilisateur d'extraire l'information sur les entités demandées dans les résultats fournis. [11]

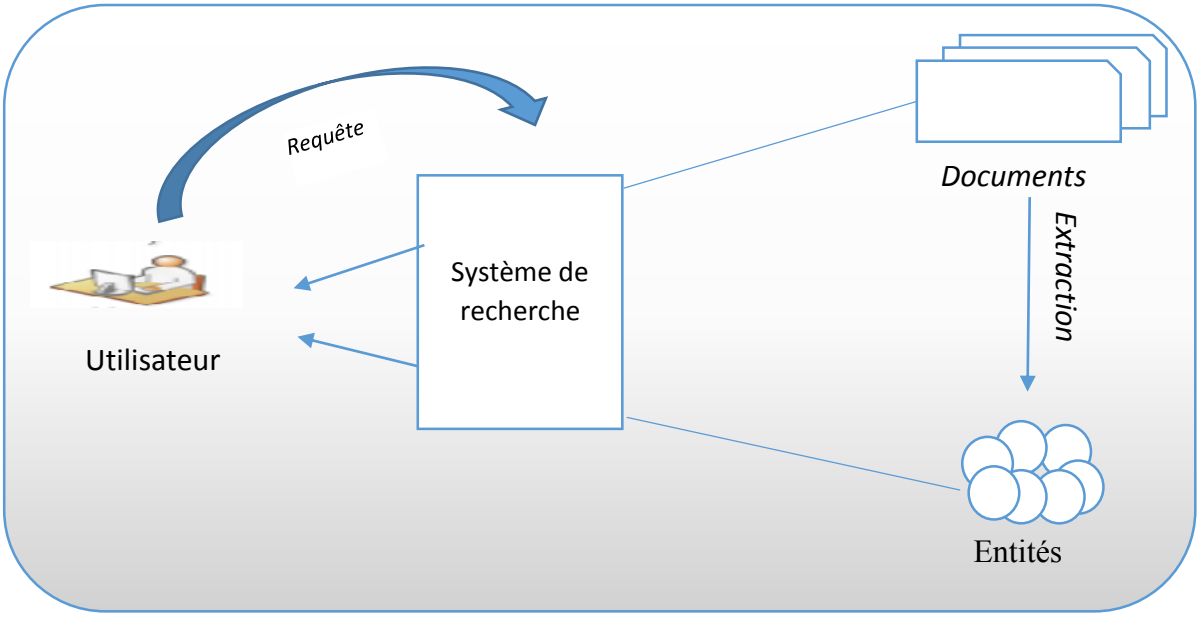


Figure 4. Recherche entités vs recherche documents. [11]

Être en mesure de trouver les entités sur le Web, peut devenir une nouvelle caractéristique importante des moteurs de recherche actuels. Il peut permettre aux utilisateurs de trouver plus que des pages Web, mais aussi des gens, des numéros de téléphone, livres, films, voitures, etc. La recherche d'entités dans une collection de documents n'est pas une tâche facile. Par conséquent, afin de trouver des entités, il est nécessaire de faire une étape de prétraitement d'identification des entités dans les documents. En outre, nous avons besoin de construire des descriptions de ces entités pour permettre aux moteurs de recherche de classer et retrouver les entités pertinentes pour une requête. L'application des méthodes de recherche RI classique pour la recherche d'entités peuvent mener à une faible efficacité. C'est parce que la recherche d'entité, est une tâche différente de celle de la recherche de documents. Un exemple d'une requête est "Aéroports" en Allemagne où un résultat pertinent est, par exemple, "l'aéroport de Frankfurt-Hahn". [11]

Il est intéressant de noter qu'un recensement des différents types de requêtes réelles du web a été effectué. Les auteurs de ce travail classifient les requêtes en quatre types et présentent le pourcentage de chaque type, comme suit :

- Requête d'entité « Entity query » (40 %), exemple : « 1978 cj5 jeep » (marque de voiture).
- Requête de type « Type query » (12 %), exemple : « doctors in barcelona ».
- Requête d'attribut « Attribute query » (5 %), exemple : « zip code atlanta ».
- Autre « Other query » (36 %), même si (14 %) de ces (36 %) contiennent un contexte d'entité ou un type. [5]

Entité, c'est quoi au juste

Les travaux en recherche d'information ont porté une attention particulière aux noms propres de personnes, de lieux et d'organisations, appelés entités nommées. Les entités nommées sont des séquences lexicales qui font référence à une entité unique et concrète, appartenant à un domaine spécifique (humain, social, politique, économique, géographique, etc). [13]

II.2.1 Exemple de motivation

Les entités recherchées pouvant être connues ou inconnues aux utilisateurs, Cela signifie qu'il existe différents choix pour poser sa requête : recherche par une seule entité (R1E), recherche par plusieurs entités (RPE) et recherche par mots clés (RMC).

Remarque : si la requête est un mélange d'entités et de mot clés, elle est alors considérée comme une requête de mots clés (RMC), car nous supposons que lorsque l'utilisateur forme une requête d'entités (RPE), c'est qu'il cherche un lien ou veut faire une comparaison (ex., Renault ou Peugeot, infection et tumeur, etc.). [5]

Nous présentons dans ce qui suit un exemple de motivation. Cet exemple est tiré d'un contexte d'informations politiques.

Supposons qu'un utilisateur souhaite avoir des informations sur la politique algérienne. L'utilisateur peut poser différentes requêtes

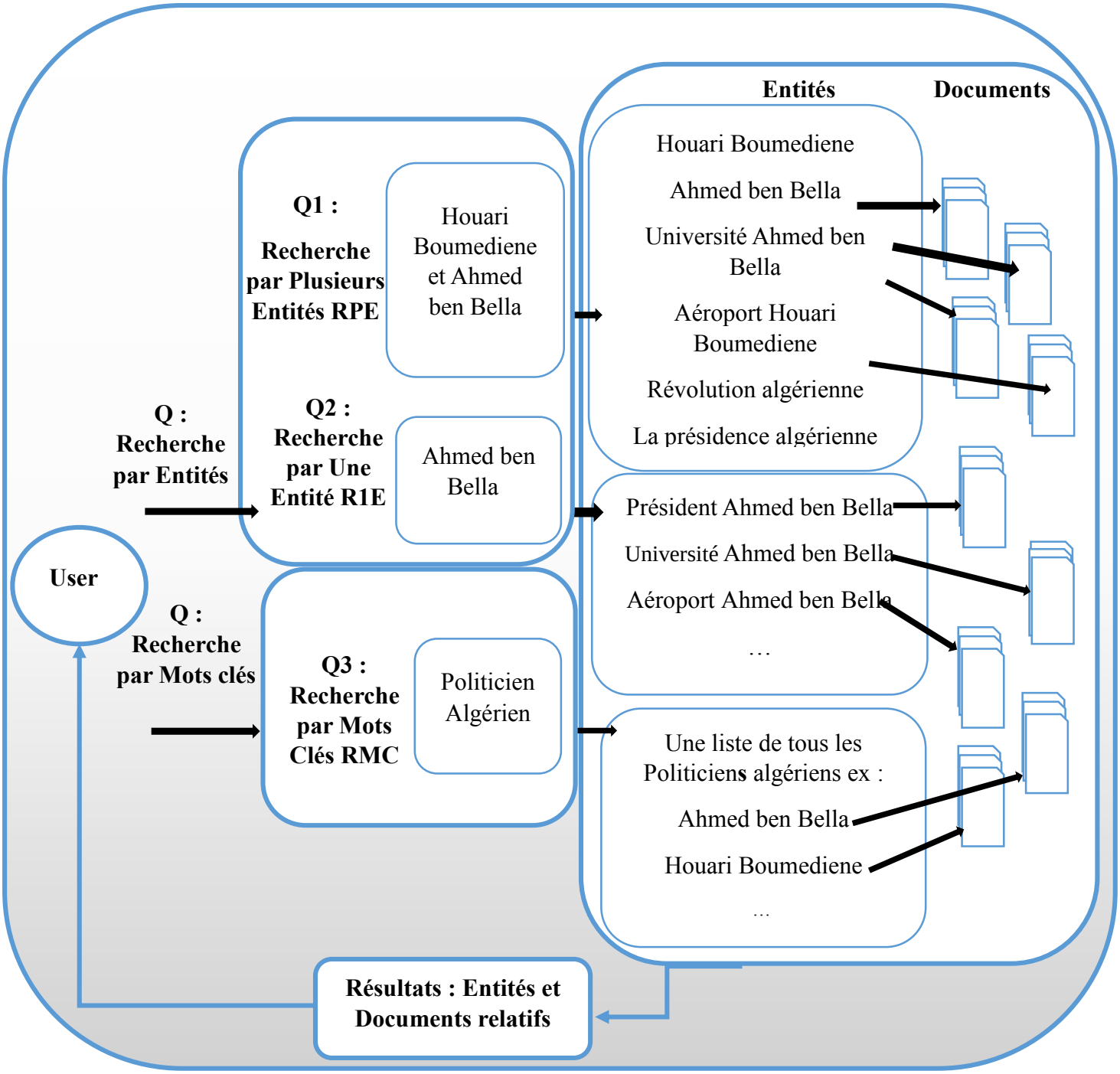


Figure 5. Exemple de motivation

Dans la première requête, l'utilisateur veut avoir des informations sur deux personnes, il s'intéresse aux informations communes entre ces deux personnes. Ce cas représente la recherche par plusieurs entités (RPE). L'utilisateur saisit les entités (connues) : "Houari Boumediene et Ahmed ben Bella ", il aura en résultat les entités relatives à sa recherche ainsi que leurs documents. L'utilisateur pourra explorer les documents relatifs aux entités trouvées.

La deuxième requête est un cas spécial de la recherche précédente RPE. Ce cas consiste en la recherche d'une seule entité (R1E). L'utilisateur voudrait des informations sur une entité particulière. Pour cela, il saisit l'entité (connue) qu'il veut rechercher, par exemple : « Ahmed ben Bella ». Il est intéressant de retourner à l'utilisateur, en plus de sa requête : « Ahmed ben Bella », les entités composées par cette dernière (par exemple, Ahmed ben Bella aéroport, Ahmed ben Bella université, etc.). L'utilisateur pourra alors explorer leurs documents relatifs.

Dans la troisième requête, l'utilisateur veut avoir des informations sur une requête formée de mots clés, par exemple : « président de l'Algérie ». Ce cas représente la recherche d'entités (inconnues) par mots clés (RMC). Les résultats sont les entités relatives (pertinentes et contextuelles) à cette requête et les documents répondant à chaque entité.

II.3 Travaux relatifs à la recherche d'entités

Dans les travaux de l'état de l'art, les entités sont recherchées pour réaliser différentes tâches pour de différents buts (objectifs). Nous commençons par donner une taxonomie des différentes tâches présentée dans un travail de recherche. [11]

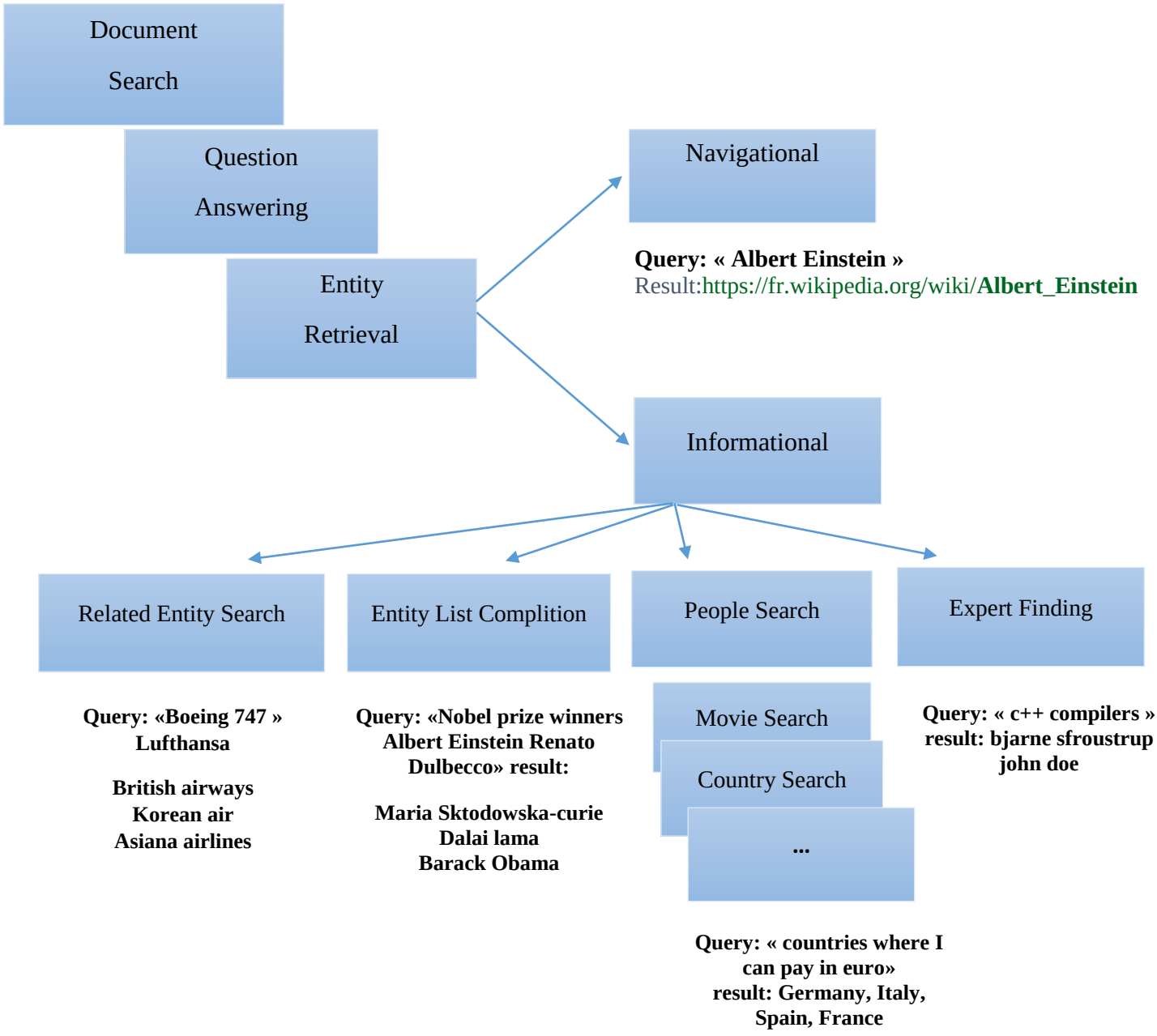


Figure 6. Taxonomie des tâches de recherche de l'entité [11]

II.3.1 Taxonomie des tâches de recherche de l'entité

Avec la taille actuelle du Web et la diversité des données qu'il contient, les moteurs de recherche traditionnels se limitent à de simples besoins d'information. Vue que les requêtes sont de plus en plus commun et ont besoin, en général, de beaucoup d'effort du côté de l'utilisateur pour qu'il soit satisfait. Dans cette catégorie on peut trouver plusieurs tâches de recherche connexes de l'entité : [11]

II.3.1.1 Recherche d'expert « Expert Finding »

Cette tâche est exécutée dans un contexte d'entreprise. Elle consiste à trouver des personnes, par exemple, un expert (employé, associé ou de toute autre personne dans l'entreprise) qui a une connaissance particulière sur un sujet donné. [5]

II.3.1.2 Recherche d'entités « Entity Retrieval »

Trouver des entités pour de différents types est un challenge qui va au-delà de la recherche d'information classique et au-delà de la recherche d'un seul type d'entités. La motivation est que les requêtes ne cherchent pas des documents, mais une liste d'entités spécifiques : pays, acteurs, etc. Un outil commercial a été présenté par Google pour effectuer cette tâche, Google Squared[2]. Pour des requêtes différentes, les entités peuvent être de différents types : personne, pays, etc. mais pour une même requête, les entités sont d'un seul type. [5]

II.3.1.3 Complétion[3] de la liste d'entité « Entity List Completion »

C'est une tâche similaire à la recherche d'entités. Dans ce cas, l'utilisateur en plus de sa requête, doit fournir au système de recherche des exemples d'entités pertinentes. Par exemple, pour la requête "pays où je peux payer en Euro" l'utilisateur doit sélectionner Allemagne, Italie, Espagne. Le système retourne les entités complémentaires qui ne sont pas fournies par l'utilisateur.

II.3.1.4 Question/réponse « Question Answering »

Une tâche de recherche d'entité est liée à la réponse aux questions (QA). Les questions dans le contexte de QA, sont caractérisées par le qui, quand, où, pourquoi et combien, et comme dans le cas de recherche d'entités, les réponses attendues sont précises et l'utilisateur s'attend à une liste d'items. [5]

II.3.1.5 Entités relatives « Related Entities »

Une autre tâche relative est de trouver des entités similaires ou liées à d'autres entités. Dans ce cas, l'utilisateur peut soumettre une requête composé d'une entité. Pour une entité donnée, telles que "New York", on devrait s'attendre que les entités associées sont : lieux à visiter à New York (par exemple, "Empire State Building", "Statue de la Liberté"), événements historiques (ex : "11 septembre 2001") ou des gens célèbres (ex : "Rudy Giuliani"), etc.

Les entités associées peuvent être présentées à l'utilisateur comme des listes regroupées par type : endroits, personne, etc. Pour une requête "Albert Einstein", le système peut retourner les entités connexes comme, par exemple, "l'Allemagne", "Prix Nobel", "Physique", "Lieserl

[2]http://www.google.com/squared

[3]Le terme d'origine française est le complètement, c'est-à-dire l'action de compléter.

Einstein", etc. Cette tâche est différente de la recherche d'entité, que l'ensemble de résultats peuvent contenir des entités de types différents.

Ici, le système fournit à l'utilisateur une possibilité de navigation plutôt qu'avec une liste d'entités extraites comme pour les recherche d'entité. Un prototype commercial qui effectue cette tâche est Yahoo ! Correlator[4]. [11]

II.4. Evaluation du système de recherche d'entités

Les systèmes de recherche sont toujours évalués en fonction de la pertinence des documents retrouvés. Afin de procéder à des évaluations automatiques, nous avons besoin de corpus de test « standard ». Chaque corpus contient :

- L'ensemble de documents.
- L'ensemble de requêtes de test sur l'ensemble de documents du même corpus.
- La liste de documents pertinents pour chaque requête.

Un système de recherche quelconque peut utiliser ce corpus pour trouver des documents pour les requêtes données, et nous pouvons comparer ces documents retrouvés avec la liste de documents pertinents pour évaluer la qualité du système.

Les deux principales mesures utilisées pour évaluer un système de recherche sont la précision et le rappel, qui sont définis comme suit [1] :

- **Précision = nombre total de documents pertinents retrouvés par le Système ／nombre total de documents retrouvés par le système**

- **Rappel = nombre total de documents pertinents retrouvés par le système ／nombre total de documents pertinents dans le corpus**

- Précision indique la précision du système par rapport aux informations retournées, c'est à dire, le nombre d'informations correctes parmi celles extraites.

- Rappel indique le pourcentage des informations pertinentes que le système a retourné, c'est à dire, le nombre d'entités trouvées parmi celles qui sont supposées être trouvées.

[4]http ://sandbox.yahoo.com/what-is-correlator

Dans la recherche d'entité, la pertinence est calculée à partir de résultats des entités retrouvés, c'est la mesure de la qualité des résultats obtenus pour vérifier les résultats satisfont le besoin d'un utilisateur [5].

Si on applique la formule classique du calcul de la précision et du rappel sur nos résultats, nous aurons à calculer ce qui suit :

- **Précision = entités correctes trouvées / nombre total des entités trouvés.**
- **Rappel = entités correctes trouvées / nombre total des entités correctes.**

II.5 Conclusion

Dans notre travail, nous nous intéressons à des domaines d'applications spécifiques où la plupart des documents sont écrits autour d'entités nommées. Parmi ces domaines, nous pouvons trouver les forums de discussion, les articles de journaux, les wikis news, etc. Dans notre travail, nous visons à offrir à l'utilisateur la possibilité de trouver des entités pertinentes aux différentes requêtes tout en augmentant la diversité des résultats.

CHAPITRE III

Conception de notre système de recherche d'information

III-Conception de notre système de recherche d'information

III.1 Introduction

La recherche d'informations est la science de la recherche des documents, d'information dans les documents, et pour des métadonnées sur des documents. Généralement effectuée en indexant préalablement tous les documents selon les mots qu'ils contiennent, pour but de faciliter la récupération des informations.

L'objectif de notre travail est de créer un système capable de retourner une liste d'entités et les documents qu'ils contiennent comme résultat de recherche. Nous nous basons principalement sur un annotateur qui identifie les entités existantes dans le corpus afin de les stocker dans des index, et aussi dans un deuxième temps pour diversifier les documents des entités trouvées selon leurs types ou selon les catégories des documents.

III.2 Indexation et Annotation

Dans ce chapitre, nous abordons les étapes de conception de notre système de recherche d'information basé sur l'annotation d'entité nommés. Pour cela nous devons passer par deux étapes principales qui sont : l'indexation de corpus de document pour la recherche d'information et l'annotation d'entité nommée pour affiner l'étape de la recherche.

Pour l'indexation nous avons utilisons un logiciel libre de recherche d'information Lucene. Et pour l'annotation nous utilisons le Stanford NER API.

III.2.1 Moteur de recherche lucene[5]

Est un moteur de recherche et d'indexation développé dans le projet Apache. C'est un logiciel open source signifiant que son code source est libre et accessible gratuitement. Ce logiciel est une librairie de fonctions de recherche dans le contenu textuel des documents. Il inclut une interface de programmation (API).

A la base, Lucene est écrit en Java mais il est maintenant disponible pour d'autres langages de programmation tels que Python, PHP, Delphi, Perl, C++, C# et Ruby. Lucene peut être utilisé avec de nombreux systèmes, c'est une multiplateforme pour : Windows, MacOS et Linux. Lucene est capable de traiter de grands volumes de documents grâce à sa puissance et à sa rapidité dues à l'indexation. [23]

III.2.1.1 Pourquoi Lucene ?

Les modules de recherche proposés par le dépôt de données que nous utilisons, s'avérant relativement sommaires, nous avons décidé d'externaliser la recherche plein texte au cœur du corpus : un seul logiciel open source répondait vraiment à nos exigences : Lucene, il est utilisé comme moteur d'indexation de notre corpus : toute la partie de recherche simple dans le corpus est assumée par Lucene.

Lucene a beaucoup d'atouts :

- Il utilise des algorithmes puissants, exacts et efficaces.
- Il calcule un score pour chaque document et retourne une liste classée par pertinence

[5] http://lucene.apache.org/

- Il propose de nombreux modèle de requêtes, tels que FuzzyQuery, BooleanQuery, et d'autres encore.
- Il permet aux utilisateurs d'étendre le comportement de la recherche en utilisant des tris personnalisés, des filtres et l'analyse d'expressions de requêtage.
- Il permet d'indexer et de rechercher simultanément.

III.2.1.2 Architecture et fonctionnement de Lucene

Lucene, une librairie de recherche open source très populaire, est caractérisé par deux fonctionnalités ou deux tâches principales qui sont : L'indexation et la recherche.

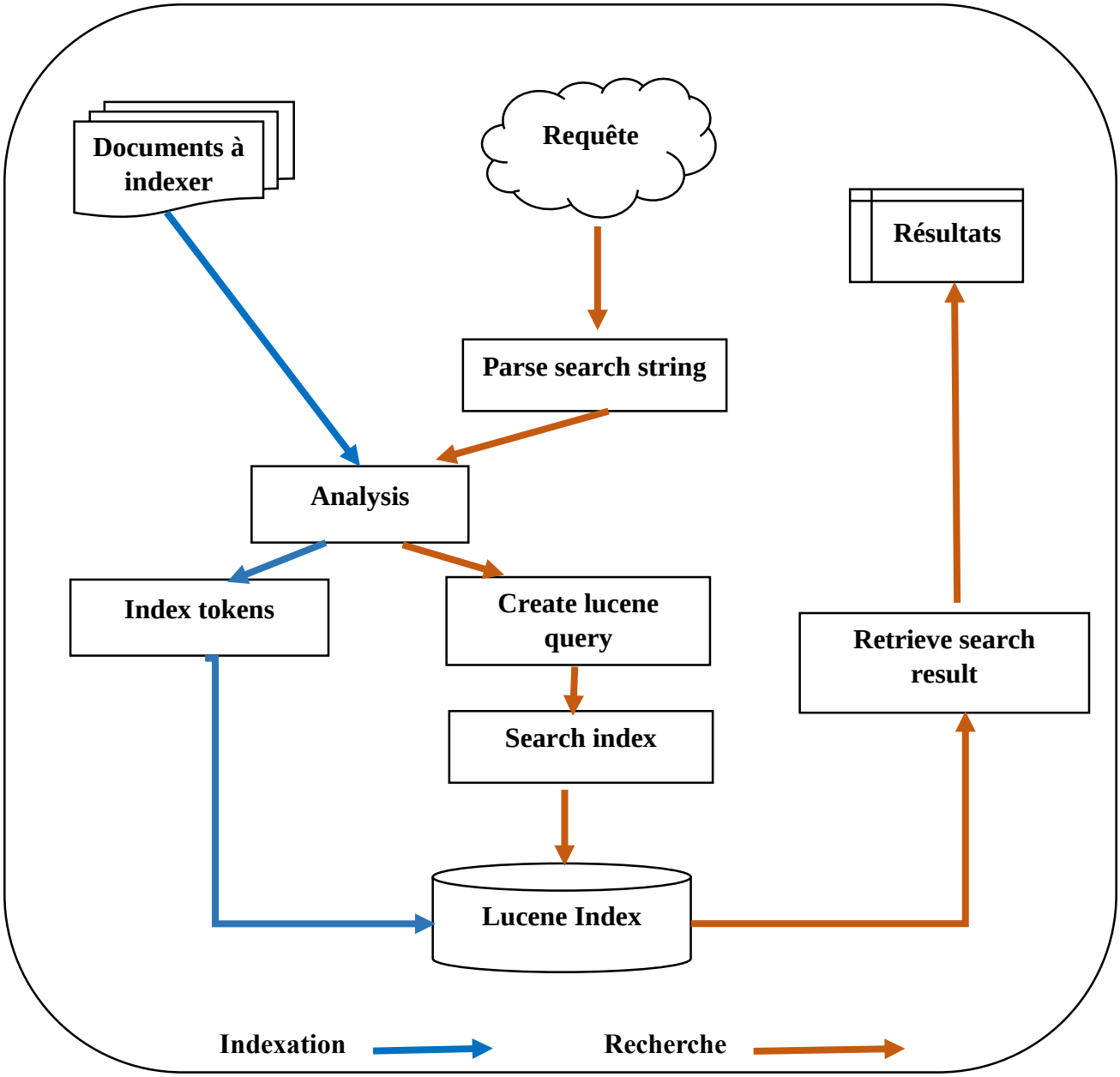


Figure 7. L'architecture générale de lucene

III.2.1.2.A Processus d'indexation

L'indexation est le processus de conversion des données textuelles vers un format qui facilite une recherche rapide. Lucene stocke les données en entrée dans une structure de données appelée index inversé, qui est lui-même stocké comme un ensemble de fichiers d'index.

Permet de Créer un **IndexWriter** utilisé pour écrire le fichier d'index en choisissant un **Analyseur** compatible avec ce dernier dans notre cas c'est le **standardAnalyzer**.

Les classes principales pour l'indexation sont définies comme suit :

- **Directory :** Une classe abstraite qui représente l'emplacement où les fichiers d'index sont stockés. Il existe principalement deux sous-classes couramment utilisées :
  - **FSDirectory-** une implémentation de directory qui stocke les index dans le système de fichiers. Ceci est utile pour les grands indices.
  - **RAMDirectory**- Une application qui stocke tous les indices dans la mémoire. C'est adapté pour les plus petits indices qui peuvent être entièrement chargé en mémoire et détruits lorsque l'application se termine. Comme l'index est conservé dans la mémoire, c'est comparativement plus rapidement.

- **Analyzer** : est une classe abstraite livrée avec plusieurs implémentations. Avant l'indexation, le texte est passé à travers l'analyseur spécifié dans le constructeur IndexWriter. Permet de convertir des données textuelles en unités fondamentales de recherche appelées termes. Pendant l'analyse, le texte subit plusieurs opérations : extraction des mots, réduction du mot à leur racine, le passage en minuscule, etc…

- **IndexWriter** : cette classe est l'élément central du processus d'indexation. Elle permet de créer un nouvel index (ou ouvrir un index existant), Son constructeur accepte une valeur booléenne qui détermine si un nouvel indice est créé ou si un index existant est ouvert. Il fournit des méthodes pour ajouter, supprimer ou mettre à jour les documents dans l'index.

III.2.1.2.B Processus de recherche

Rechercher, c'est examiner l'index pour trouver des mots puis obtenir les documents qui contiennent ces mots.

Cette phase permet de lire l'index créé à l'aide de **l'IndexReader**, elle permet aussi de créer un **IndexSearcher** prêt à rechercher en choisissant un Analyseur qui va être interrogé par **QueryParser.** Les classes principales pour effectuer la recherche sont définies comme suit :

- **L'IndexReader :** IndexReader est une classe abstraite qui propose diverses méthodes pour accéder à l'index. Lucene fait référence à l'interne des documents avec les numéros de document qui peut changer au fur et à mesure que les documents sont ajoutés ou supprimés de l'index. Le numéro de document est utilisé pour accéder à un document

dans l'index. IndexReader ne peut pas être utilisé pour mettre à jour les indices dans un répertoire pour lequel IndexWriter est déjà ouvert. IndexReader recherche toujours l'instantané de l'index lorsqu'il est ouvert. Toute modification de l'indice n'est pas visible jusqu'à ce que l'IndexReader est réeouvert. Il est important que les applications utilisant Lucene réouvre leur IndexReaders à voir les dernières mises à jour de l'index.

- **IndexSearcher** : Est une classe qui ouvre un index en lecture seule, elle retourne un tableau de référence vers les résultats de la recherche, tels que les documents qui satisfont une requête donnée on peut décider du nombre du résultat à retourner en le spécifiant dans la méthode de recherche d'IndexSearcher. Les classes les plus importantes en ce qui concerne la manipulation des résultats sont ScoreDoc et TopDocs.
    - **TopDocs** : cette classe est un simple conteneur de pointeurs vers les N premiers documents des résultats de recherche, qui correspondent à une requête donnée.
    - **ScoreDoc :** un simple pointeur vers un document satisfaisant la recherche. Il encapsule la position de la recherche dans l'index et le score calcule par lucene.

- **QueryParser :** QueryParser est utile pour l'analyse des requêtes entrées par l'utilisateur. On peut l'utiliser pour analyser les expressions de requête entrée par l'utilisateur dans un objet de requête Lucene, ce qui peut être passée à la méthode de recherche de l'IndexSearcher. Il peut analyser des requêtes riches et les convertir en une des sous classes de requêtes concrètes.

III.3 Annotation

Les systèmes d'annotation automatique de documents apparus récemment, rattachent automatiquement des métadonnées sémantiquement riches à des documents, en catégorisant et en liant ces documents à des entités.

Dans ce travail, nous avons utilisé l'annotateur Stanford NER pour l'annotation de corpus. L'utilisation de ce système permet d'annoter automatiquement le corpus et extraire les entités nommées existantes, les catégories des documents ainsi que d'autres informations. Pour but d'exploiter les différences entre les entités (les types d'entités) et entre les documents (les catégories des entités dans un documents) pour proposer une nouvelle approche de diversification des résultats dans le contexte de la recherche d'entités.

L'extraction des entités consiste à les référencés à des personne, des lieux, des organisations, etc. qui sont contenues dans un texte.

III.3.1 Stanford Named Entity Recognizer (NER)

Stanford NER[6] est une implémentation en Java pour la Reconnaissance entité nommée. La reconnaissance des entités nommées (REN) permet l'étiquetage des séquences des mots dans un texte qui sont les noms propres, telles que les noms de personne et d''entreprise, des lieux ou bien des organisations etc... Il est livré avec des extracteurs de fonctions bien conçus pour la reconnaissance d'entité nommée. La reconnaissance entité nommée sont bien reconnus pour l'anglais, en particulier pour les 3 classes (types d'entités) (PERSONNE, ORGANISATION, LOCALISATION).

Stanford NER est également connu sous le nom de CRFClassifier CRFClassifier (Conditional Random Field (CRF)). Le logiciel fournit une implémentation générale de modèles de séquence de champ aléatoire conditionnel (CRF) de chaîne linéaire (ordre arbitraire). C'est-à-dire en formant vos propres modèles sur les données étiquetées.

Il fonctionnant comme un serveur, et une API Java. Stanford NER code est sous licence double (d'une manière similaire à MySQL, etc.). [25]

- **Exemple: annotation avec Stanford NER (Stanford Named Entity Tagger)[7]**

coa is still available as harvesting has practically come to an end. With total Bahia crop estimates around 6.4 mln bags and sales standing at almost 6.2 mln there are a few hundred thousand bags still in the hands of farmers, middlemen, exporters and processors. There are doubts as to how much of this cocoa would be fit for export as shippers are now experiencing dificulties in obtaining Bahia superior+ certificates. In view of the lower quality over recent weeks farmers have sold a good part of their cocoa held on consignment. Comissaria Smith said spot bean prices rose to 340 to 350 cruzados per arroba of 15 kilos. Bean shippers were reluctant to offer nearby shipment and only limited sales were booked for March shipment at 1,750 to 1,780 dlrs per tonne to ports.

ORGANIZATION
LOCATION
PERSON

Figure 8 : Exemple d'un texte annote (avec Stanford Named Entity Tagger).

[6] https://nlp.stanford.edu/software/CRF-NER.shtml
[7] http://nlp.stanford.edu:8080/ner/process

III.3.2 Open Calais

Calais est une boîte à outils pour de plus en plus de fonctionnalités qui vous permettent d'intégrer facilement des fonctionnalités sémantiques au sein de votre blog, système de gestion de contenu, site web ou application.

Le Service Web OpenCalais crée automatiquement des métadonnées sémantiques riches pour le contenu que vous soumettez en moins d'une seconde. En utilisant le traitement du langage naturel, l'apprentissage automatique et d'autres méthodes, Calais analyse votre document et conclut les entités au sein de celui-ci. Mais, Calais va bien au-delà de l'identification de l'entité classique et renvoie les faits et événements cachés dans votre texte. [22]

Open Calais : une initiative de Thomson Reuter lancée en 2007, elle a pour but d'enrichir sémantiquement les textes. Cette initiative permet d'identifier et méta taguer les personnes, les compagnies, les faits et événement et retourne des résultats sous format RDF (Resource Description Framework). Elle permet également de se connecter aux sources de données du web de données « Linked Data Cloud », qui incluent Wikipédia, [8]DBPedia, Shopping.com, the internet Movie Data base (IMDB), etc.

Mr. <enamex type=« person » > Dooner </enamex> met with<enamex type=« person » > Martin Puris </enamex>, president and chief executive officer of <enamex type=« organization » >Ammirati & Puris </enamex>, about <enamex type=« organization » > McCann </enamex>'s acquiring the agency with billings of <numex type=« money » > $400 million </numex>, but nothing has materialized

Figure 9. Exemple d'un texte annote (avec open calais)

III.4 Indexation

L'indexation consiste à traiter le corpus de documents pour créer un index des termes importants (index classique de mots clés). L'indexation consiste aussi à stocker les informations extraites par l'annotation dans d'autres index. D'autres tâches sont exécutées aussi, telles que le calcul des fréquences d'apparition des mots clés localement (dans le document) et globalement (dans tout le corpus). Donc nous aurons besoin de trois index : l'index de mot clé, l'index d'entité et l'index de document, Ces trois index sont détaillés dans ce qui suit :

[8]DBpedia est un projet universitaire et communautaire d'exploration et extraction automatique de données dérivées de Wikipédia. Http : // fr.wikipédia.org/wiki/DBpedia

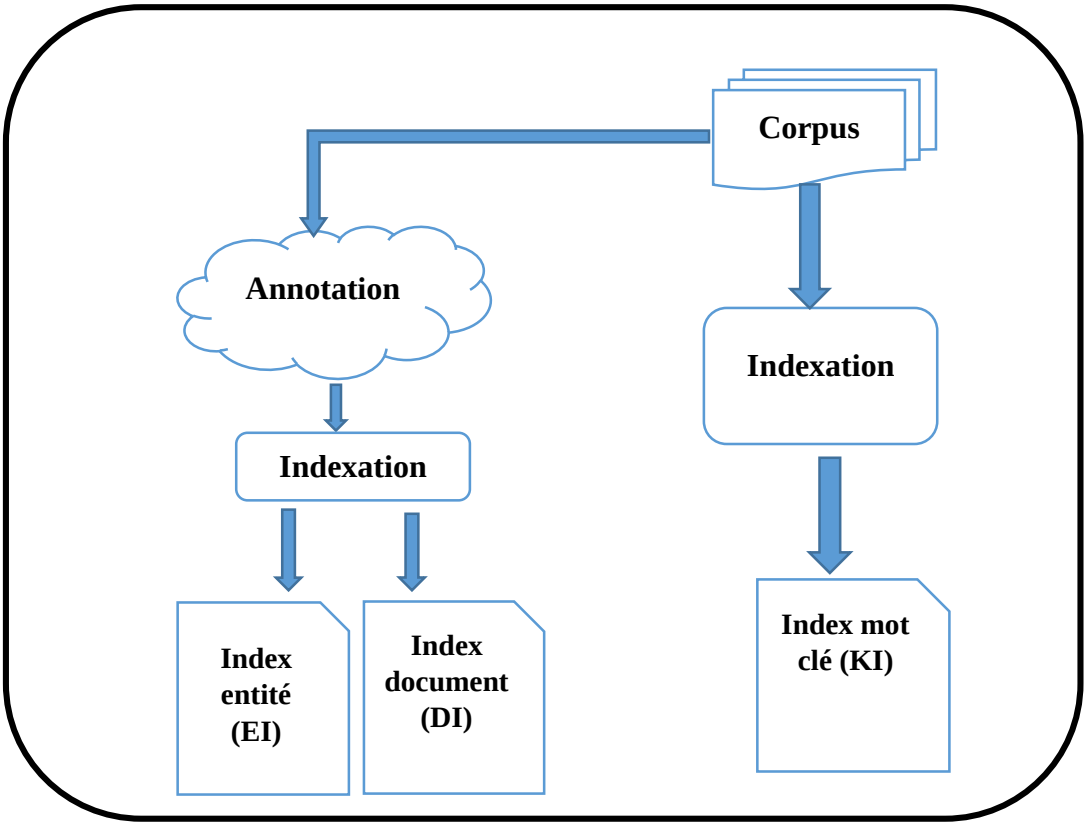


Figure 10. La phase d'indexation de notre système.

Nous considérons, un corpus de documents semi ou non structurés (par exemple, wiki news, articles de journaux, etc dans notre cas nous utilisons le corpus reuters21578[9]). Notre approche consiste à faire un prétraitement pour préparer les informations pour effectuer la recherche (Figure 8).

Nous commençons par annoter le corpus en utilisant un système d'annotation automatique des entités nommées dans notre cas nous avons choisi le Stanford NER, pour extraire les entités, leurs types et les catégories des entités dans des documents avec les scores d'extraction.

Nous créons différents index pour stocker les informations, i.e. les mots clés, les entités, les types, les catégories et les scores.

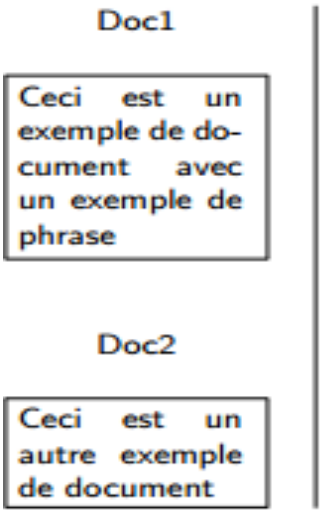
Les scores sont : le score du mot clé, le score pour les entités et le score pour la catégorie d'entité du document.

Trois index sont créés, à savoir : un index inversé classique pour les mots clés (KI, Keyword Index), un index inversé pour les d'entités du corpus (EI, Entities Index), i.e. une entité apparait dans un document avec quel type (type d'entité dans le document). Le troisième index pour les catégories des documents (DI, Document Index), i.e. quelle est la catégorie d'un document.

[9] http://igm.univ-mlv.fr/~mconstan/enseignement/m2pro/tal/tal-td2/node1.html

III.4.1 Keyword Index (KI)

Est un index inversé qui fait correspondre à chaque mot trouvé dans un corpus, la liste des documents où il se trouve et les documents le contenant avec un score. Pour cela on doit calculer le TF/IDF fréquence d'apparition du mot dans le document (figure 9).



	Dictionnaire		Assignations	
Terme	Nbr docs	Fréq totale	Doc Id	Fréq
Ceci	2	2	1	1
			2	1
est	2	2	1	1
			2	1
exemple	2	3	1	2
			2	1
autre	1	1	2	1
...	...	...	...	...

Figure 11. Exemple d'index inversé

III.4.1.1 Le calcule de score

TF-IDF: sont les acronymes de « Term Frequency » et « Inverse Document Frequency», est une méthode de pondération souvent utilisée en recherche d'information . Cette mesure statistique permet d'évaluer l'importance d'un terme contenu dans un document, relativement à un corpus. Le poids augmente proportionnellement au nombre d'occurrences du mot dans le document. Il varie également en fonction de la fréquence du mot dans le corpus.

En ce qui concerne TF-IDF, on applique une relation entre un document, et un ensemble de documents partageant des similarités en matière de mots clés. On recherche en quelque sorte une relation de quantité / qualité lexicale à travers un ensemble de documents

Pour une requête avec un terme X, un document a plus de chances d'être pertinent comme réponse à la requête, si ce document possède une certaine occurrence de ce terme en son sein, et que ce terme possède une rareté dans d'autres documents reliés au premier. [21]

a) **TF** : la fréquence d'un terme (term frequency) donne l'importance d'un terme dans un document est le nombre d'occurrences du terme dans le document considéré, normalisée

$$tfi,j = ni,j / \sum nk,j$$

Où :

j : un document, i : un terme,

ni,j : le nombre d'occurrences du terme ti dans dj.

b) **IDF** :la fréquence inverse de document (inverse document frequency) donne l'importance d'un terme dans l'ensemble du corpus des documents considère plus discriminants les termes les moins fréquents est le logarithme de l'inverse de la proportion de documents du corpus qui contiennent le terme

$$idfi = log\,|D| \,/|\{dj : ti \in dj\}|$$

Où

$|D|$: est le nombre total de documents dans le corpus

$|\{dj : ti \in dj\}|$: le nombre de documents contenant le terme ti

L'algorithme suivant représente la création de l'index des mots clés "KI" (Keyword Index).

```
Entrée : D ; Corpus de documents
Sortie : KI /*Keyword Index*/
Début
 Créer fichier(KI) ;/*Créer le fichier Keyword Index*/
Pour d in D do
Extraire termes () ;/*Extraire les termes k (Keywords) du document*/
Pour k in d do
Si (k. existe (KI))
/*Si le terme k a été déjà trouvé dans un autre document (existe dans
KI) */
Ajouter (did, Tf, KI) ;
/*Ajouter le document contenant le terme et son score à k*/
/* Seul le Tf est stocké puisque idf peut être calculé une seule fois car
il est inchangeable pour un terme. */
Sinon
Écrire (k, did, Tf, KI);
/*Ecrire le terme k, son document, son Tf dans KI*/
Fin si
Fin Pour
Fin Pour
Fin
```

Algorithme 1. Création de l'index de mot clé

III.4.2 Entities Index (EI)

Les entités sont indexées de la même manière que les mots clés. Un index inversé (EI) est construit pour les entités comme un index traditionnel des mots clés (KI). L'index EI retournera une liste qui contient les informations des entités. Stanford NER est utilisé pour l'extraction des entités avec leurs types et leurs scores. Cet index inversé stocke pour une entité donnée les types de l'entité avec les documents les contenant avec un score (score du type de l'entité dans un document).

L'index EI est utilisé pour trouver les documents relatifs aux entités dans le cas de la recherche par entités, il est utilisé également pour trouver les documents relatifs aux entités une fois l'ensemble des entités et leur documents construit. Il est utilisé aussi dans le cas la diversification par types.

Nous aurons, e : {(did, type, escore (did, e))}

Exemple :
Albert Einstein : {(575, person, 0.332), (810, person, 0.341), (856, person, 0.315), (881, person, 0.331)}.

III.4.3 Document Index (DI)

Cet index contient les entités d'un document et sa catégorie. Stanford NER est utilisé pour l'extraction des entités des documents. Stanford NER extrait une ou plusieurs catégories mais dans notre travail nous considérons la catégorie qui a le score le plus élevé. Cet index fait correspondre à chaque document, les entités qui apparaissent dans ce document ainsi que sa catégorie avec son score.

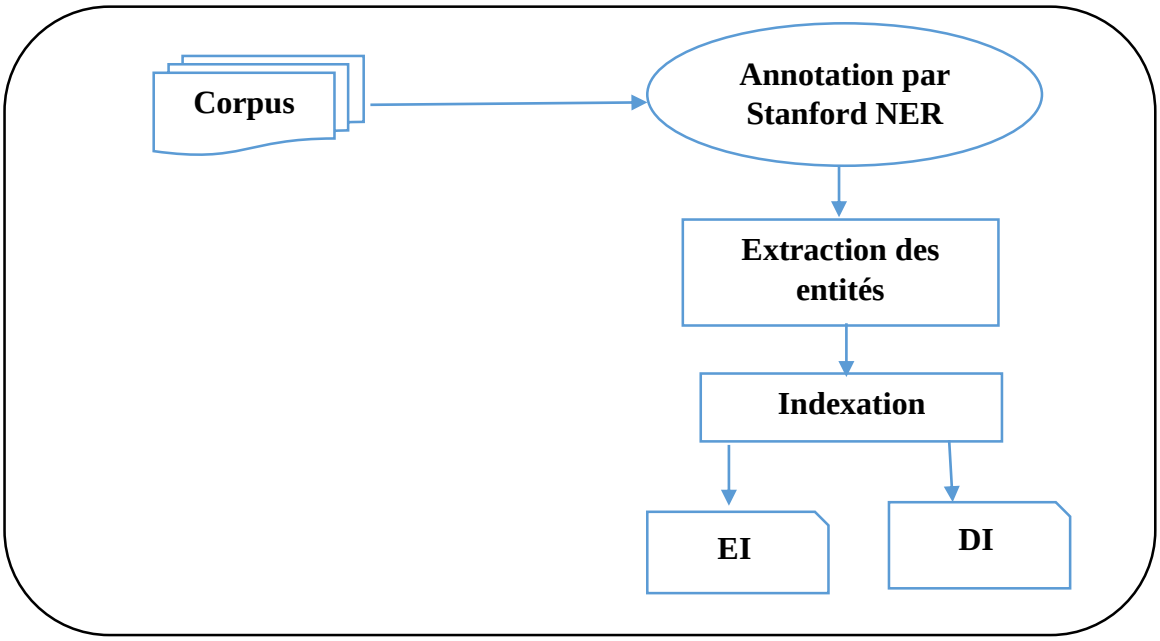


Figure 12. Création d'index d'entité EI et d'index document DI.

L'algorithme suivant représente la création des deux index l'index d'entité EI (Entity Index) et l'index mots de document DI (Documents Index).

```
Entrée : D (Corpus de documents) ;
Sortie : EI (Entities Index) ;
Début
/*la création de DI*/
Créer fichier(EI) ;
/*Créer le fichier Document Index*/
Pour d in D do
/*Pour toutes les entités extraites lors du lancement de l'annotateur avec
les documents du corpus*/
Écrire (did, c, cscore, DI) ;
/*Ecrire dans DI, le nom de document avec sa catégorie*/
/*la création de EI*/
Créer fichier(EI) ;
/*Créer le fichier Entity Index*/
Pour e in E do
/*Pour toutes les entités extraites lors du lancement de l'annotateur avec
les documents du corpus*/
Si (e. existe(EI))
/*Si l'entité e a été déjà trouvée (existe dans EI) */
Ajouter (did, type, escore, EI) ;
/*Ajouter le document contenant l'entité et le score extrait par le
système `a l'entité déjà existante dans EI*/
Sinon
Écrire (e, did, type, escore, EI) ;
/*Ecrire l'entité, son document, son type et son score dans EI*/
Fin si
Fin pour
Fin
```

Algorithme 2. Création de l'index d'entité et l'index de document

- **Les problèmes trouvent avec Stanford NER dans notre travail**

C'est un annotateur qui ne connaitre pas tous les entités nommées :

- Les entités qui commence par minuscule sont considéré comme des termes (mots).
- Les entités composées par exemple pour l'entité composé université Ahmed ben Bella il donne le type personne à Ahmed ben Bella.
- Pour une entité de différent type, mais pour Stanford NER donne un seul type par exemple Washington (localisation ou bien personne) mais dans notre cas donne le type localisation.
- Stanford NER ne connaitre pas les catégories des documents.
- Pas de calcule de score de l'entité et connaitre seulement trois types des entités nommées (localisation, organisation et personne).

III.5 La phase de recherche

III.5.1 La requête

La requête est une suite de caractère qui peut exprimer le besoin de l'utilisateur en information, dans notre système de recherche d'information nous avons effectuées quelques traitements sur la requête, pour trouver la liste d'entités pertinentes. La requête peut être constituée d'entités ou de mots clés.

III.5.2 La recherche des entités

Étant donnée la requête, il s'agit de trouver un ensemble d'entités relatives à la requête et pour chaque entité, de classer les documents qui lui sont associés. Nous avons d'exprimer les trois types de requêtes décrits précédemment comme suit :

- **Pour une requête d'entité**

Si l'utilisateur cherche par une seule entité(R1E), donc on va trouver l'entité de la requête et les entités composées avec cette entité.

Si l'utilisateur cherche par une requête de plusieurs entité (RPE), les documents qui répondent à la requête sont les documents portant sur toutes les entités de la requête, c à d les documents communs entre les entités de la requête. La construction de l'ensemble des entités pertinentes est commencée en cherchant les entités relatives aux documents communs.

- **Pour une requête de mot clé**

Pour une requête formée de mots clés, la première étape consiste à trouver les meilleurs documents qui répondent à la requête on utilise index de mots clé (KI) pour la récupération des documents qui satisfont la requête. Il s'agit ensuite de sélectionner les entités relatives aux documents trouvés.

III.5.3 La recherche des documents

Les résultats sont généralement présentés à l'utilisateur sous forme de listes de documents avec leurs informations, par exemple, le titre, un extrait du document et une adresse (dans le cas des moteurs de recherche). Une autre méthode de présentation des résultats consiste à regrouper les résultats par catégories calculées dynamiquement (clustering) ou statiquement (selon des catégories existantes au départ). Dans notre travail nous proposons de classer les résultats par entités.

Après la construction de l'ensemble des entités retournées, on va construire l'ensemble des documents relatifs à ces entités.

Pour chaque entité retourner on va récupérer les documents contenants cette entité. Le classement des documents selon le score

III.6 Conclusion

Dans ce chapitre nous avons décrire en détail le fonctionnement de moteur de recherche lucene et l'annotateur Stanford NER qui représentent la base de notre travail, puis nous avons présenté les principales fonctionnalités de notre système de recherche d'où la tâche nécessaire dans notre moteur de recherche est de retourner une liste de documents regrouper selon le type d'entités retourner ou bien leurs catégories.

Nous abordons dans le chapitre suivant l'étape d'implémentation de notre logiciel.

CHAPITRE IV

Implémentation et mise en œuvre

IV.1 Introduction

Ce chapitre vise à expliquer les différentes étapes d'implémentation et d'expérimentation de notre système de recherche ainsi que les résultats obtenus. Nous commençons par la présentation de l'environnement de développement, en détaillant les différents outils utilisés. Puis nous passons à la présentation de l'architecture de notre application, et enfin nous interprétons et commentons les résultats obtenus.

IV.2 Environnement de l'application

L'implémentation et les tests de notre application ont été réalisés dans l'environnement matériel et logiciel suivant :

- Processeur : Intel ® Core ™i3-3110M CPU @ 2.40 GHz
- Mémoire installée (RAM): 4.00 GO
- Windows: Windows 7.
- Type de système : système d'exploitation 64 bits, processeur x64.
- Java sous l'environnement Netbeans 8.1

IV.2.1. Langage d'application

Java est un langage de programmation et une plate-forme informatique créée par Sun Microsystems en 1995, racheté plus tard par Oracle. Il s'agit de la technologie sous-jacente qui permet l'exécution des applications modernes sur différentes plateformes. La portabilité, des programmes Java sur différents systèmes d'exploitation, représente son atout principal. Java est utilisée sur plus de 850 millions d'ordinateurs de bureau et un milliard de périphériques dans le monde, dont des périphériques mobiles et des systèmes de diffusion télévisuelle.

- **Pour quoi java**

Java est un langage de programmation très utilisé, notamment par un grand nombre de développeurs professionnels, ce qui en fait un langage incontournable actuellement. On a travaillé avec java car il a beaucoup de caractéristiques parmi lesquelles :

- Son excellente portabilité : une fois votre programme crée, il fonctionnera automatiquement sous Windows, Mac, Linux etc.
- On peut faire de nombreux types de programmes avec Java :
  - Des applications sous forme de fenêtre ou de console ;
  - Des applets, qui sont des programmes Java incorporé à des pages Web ;
  - Des applications pour appareils mobiles, comme les Smartphones, avec Java ME (Java Micro Edition) ;
  - Des sites Web dynamiques avec J2EE (Java 2 Entreprise Edition) ;

IV.2.2. IDE Netbeans 8.1

NetBeans, créé à l'initiative de Sun Microsystems (Noyau de Forte4J/SunOne), présente toutes les caractéristiques indispensables à un IDE de qualité, que ce soit pour développer en Java, Ruby, C/C++ ou même PHP.

De licence Open Source, NetBeans permet de développer et déployer rapidement et gratuitement des applications graphiques Swing, des Applets, des JSP/Servlets, de l'architecture J2EE, dans un environnement fortement personnalisable.

IV.3 Architecture de notre système

Nous présentons dans la (Figure **13**) l'architecture globale de notre application :

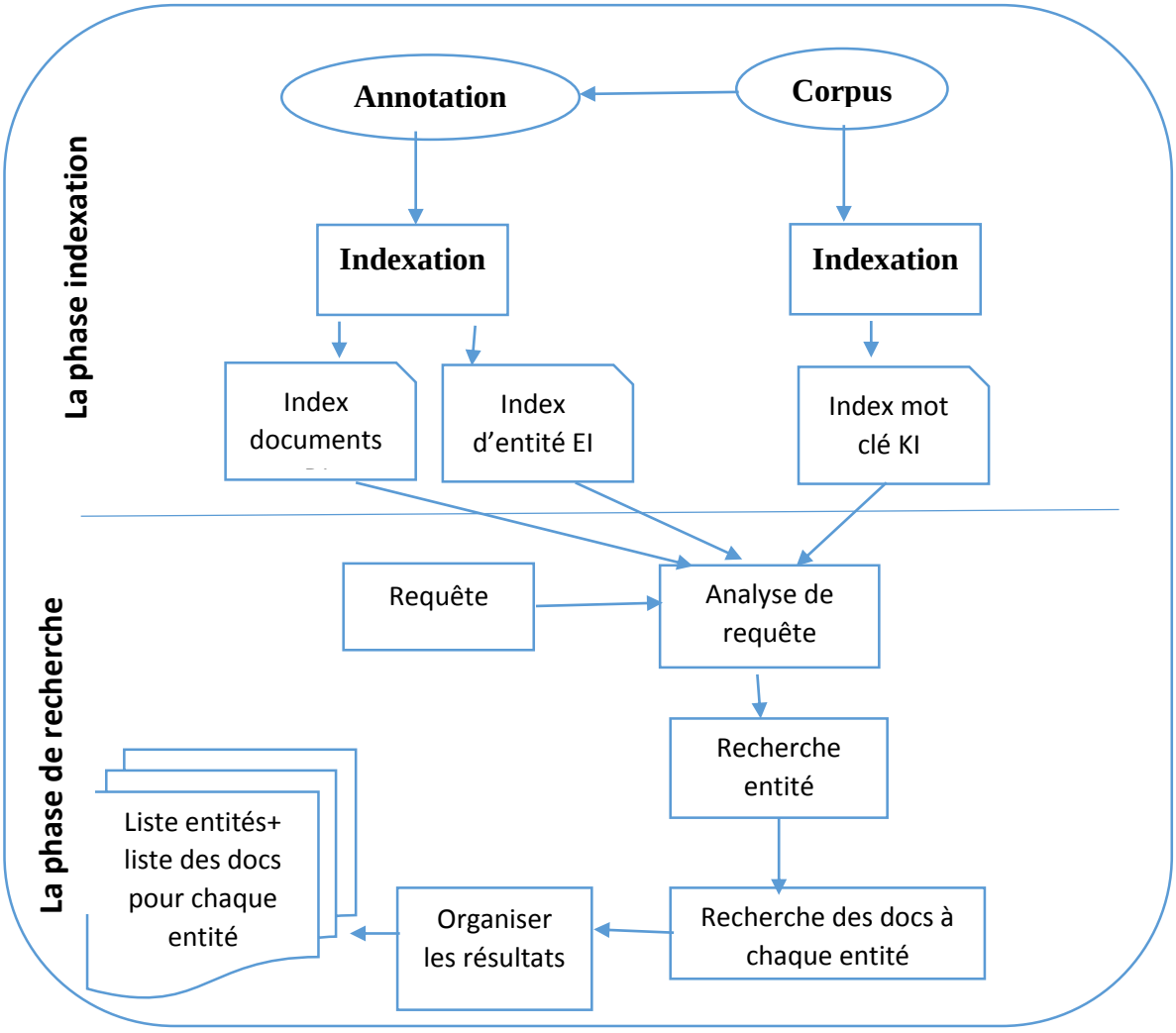


Figure 13. Architecture de notre système.

IV.4 Présentation de l'application

Dans cette partie on va décrire les différentes parties de notre application coté interface graphique et les différentes opérations de chaque bouton et menu.

La figure suivante représente l'interface principale de notre application :

Figure 14. L'interface principale de notre application

IV.4.1 Menu Principal

Notre application se compose d'un menu principal à partir de lequel l'utilisateur peut effectuer les traitements.

Le menu principal est montré dans la figure 13 :

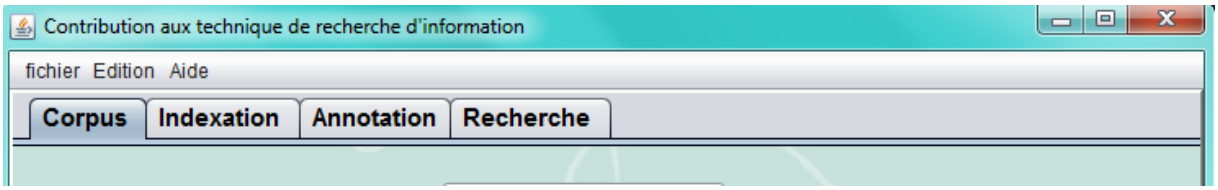


Figure15. L'interface du menu principale de notre application

IV.4.2 Corpus

L'interface dédiée pour les différentes tâches générales sur un corpus telles que l'ouverture et la consultation du contenu de chaque document.

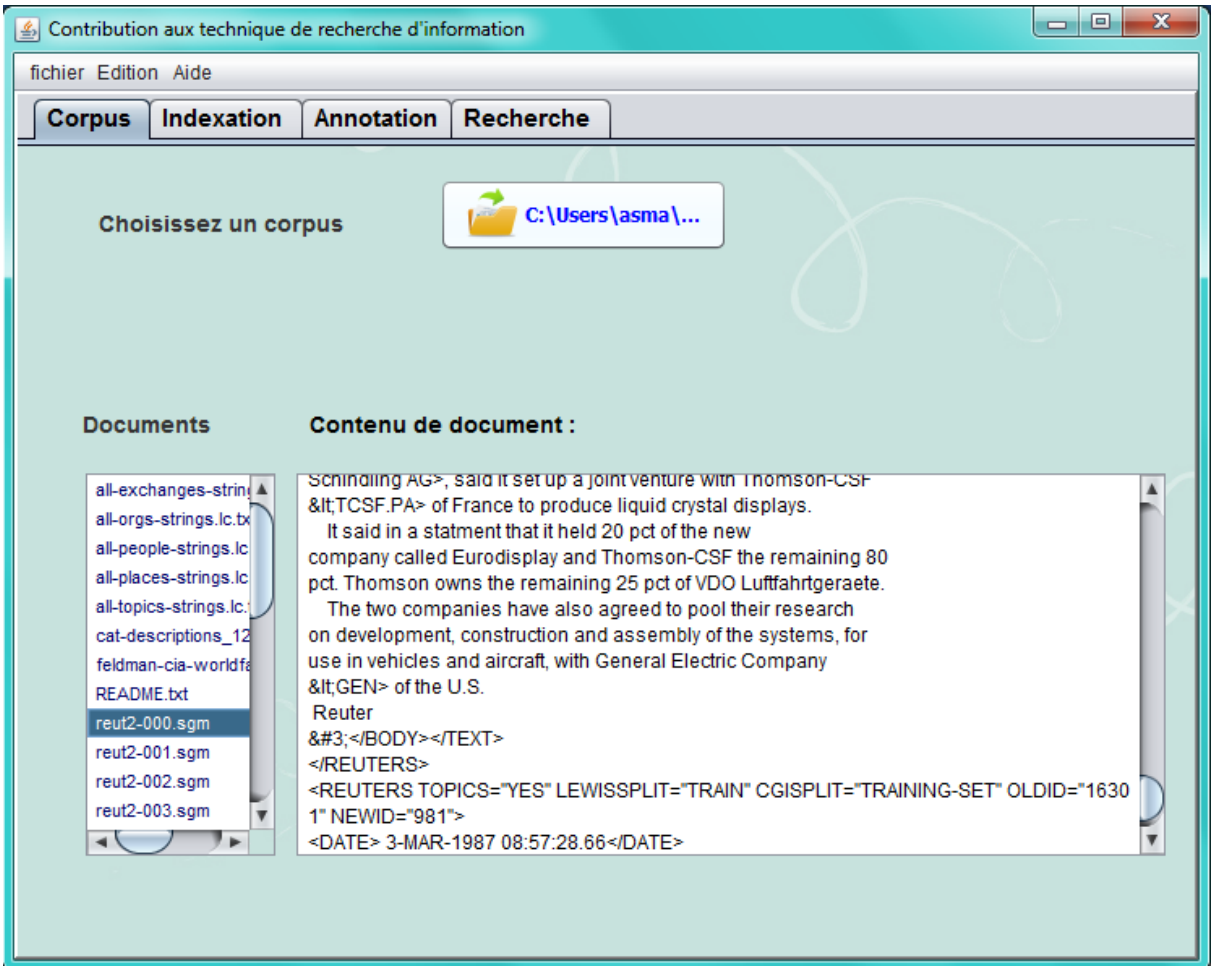


Figure 16. Interface de consultation d'un corpus.

IV.4.3 Indexation

L'indexation du corpus sélectionné pour créer l'index de mot clé (KI), permet de crée automatiquement les fichiers d'index Lucene dans un dossier fils rattaché au corpus. Des statistiques sommaires sont affichées en fin de processus et une fenêtre de dialogue informer que l'indexation et terminer.

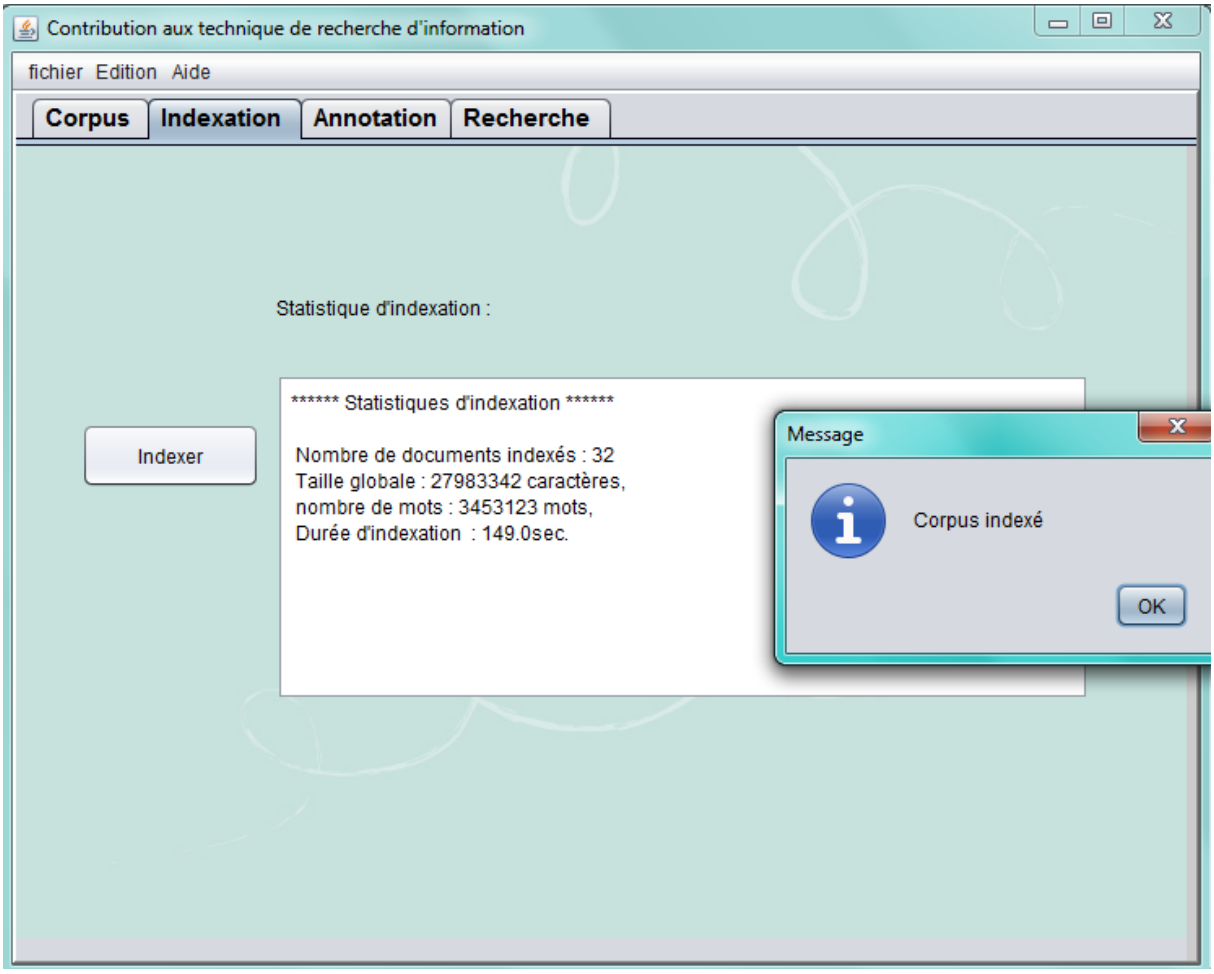


Figure 17. Fenêtre d'indexation d'un corpus

IV.4.4 L'annotation du corpus

L'annotation du corpus sélectionné, permet de crée automatiquement des fichiers annotés par les entités nommées qu'il contient. Une fenêtre de dialogue est affichée en fin de processus d'annotation qui informer que l'annotation et terminer.

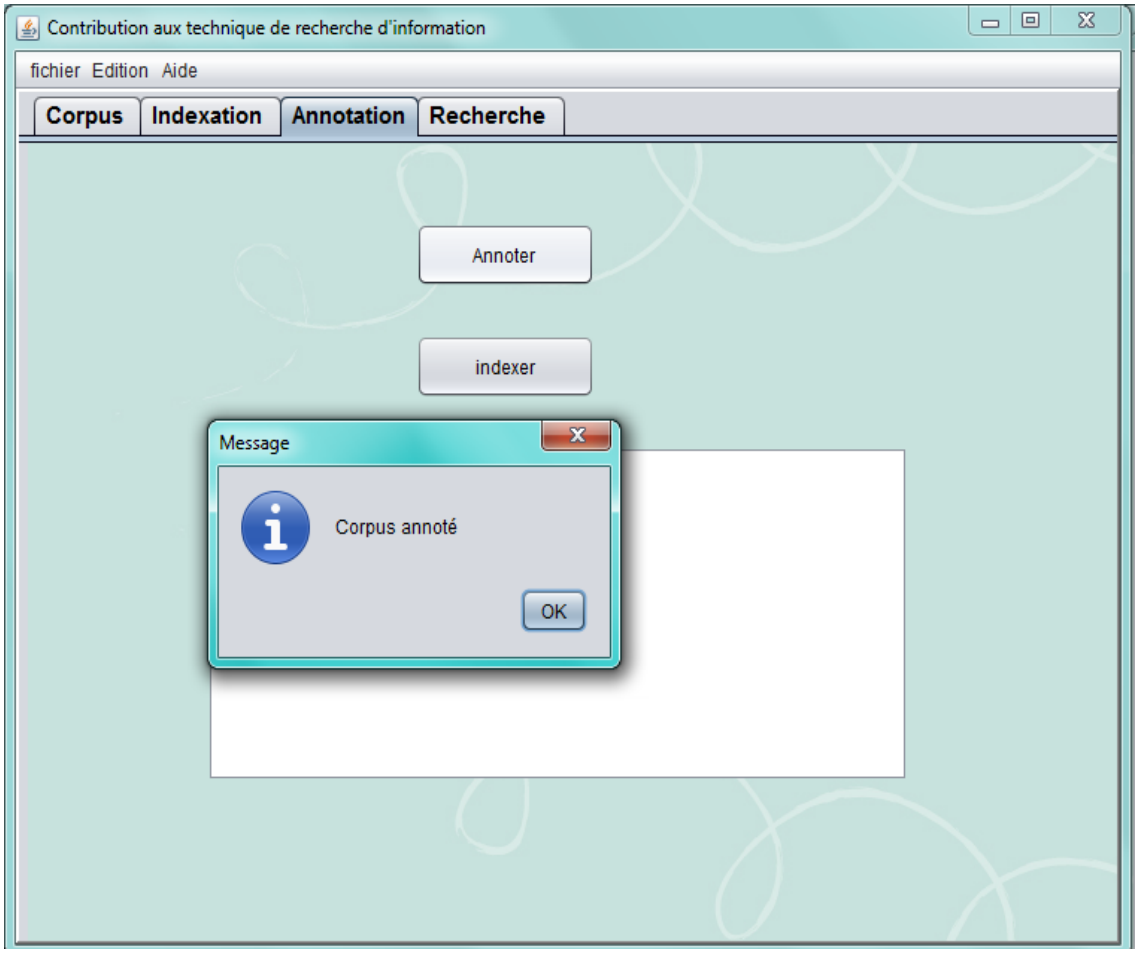


Figure 18. Fenêtre de processus d'annotation d'un corpus

- **Exemple d'un texte annoter par Stanford NER**

L'annotation dans notre travail permet d'ajouter des informations sur les entités nommées contient dans les documents. Ces informations sont des types d'entités nommées.

Nous illustrons un exemple d'un texte annoter de notre collection par la figure **19**.

<AUTHOR> by/O Janie/PERSON Gabbett/PERSON,/O Reuters/ORGANIZATION</AUTHOR>
<DATELINE> LOS/LOCATION ANGELES/LOCATION,/O Feb/O 26/O -/O </DATELINE><BODY>BankAmerica/ORGANIZATION Corp/ORGANIZATION is/O not/O under/O
pressure/O to/O act/O quickly/O on/O its/O proposed/O equity/O offering/O and/O would/O do/O well/O to/O delay/O it/O because/O of/O the/O stock/O's/O recent/O poor/O
performance/O,/O banking/O analysts/O said/O./O
Some/O analysts/O said/O they/O have/O recommended/O BankAmerica/ORGANIZATION delay/O
its/O up/O to/O one-billion-dlr/O equity/O offering/O,/O which/O has/O yet/O to/O be/O
approved/O by/O the/O Securities/ORGANIZATION and/ORGANIZATION Exchange/ORGANIZATION Commission/ORGANIZATION./O
BankAmerica/ORGANIZATION stock/O fell/O this/O week/O,/O along/O with/O other/O banking/O
issues/O,/O on/O the/O news/O that/O Brazil/LOCATION has/O suspended/O interest/O payments/O
on/O a/O large/O portion/O of/O its/O foreign/O debt/O./O
The/O stock/O traded/O around/O 12/O,/O down/O 1/8/O,/O this/O afternoon/O,/O

Figure 19. Exemple d'un texte annoté

IV.4.5 L'indexation du corpus annoté

Après l'annotation de corpus, L'indexation du corpus annoter pour créer l'index d'entité (EI). Des statistiques sommaires sont affichées en fin de processus et une fenêtre de dialogue informer que l'indexation est terminée.

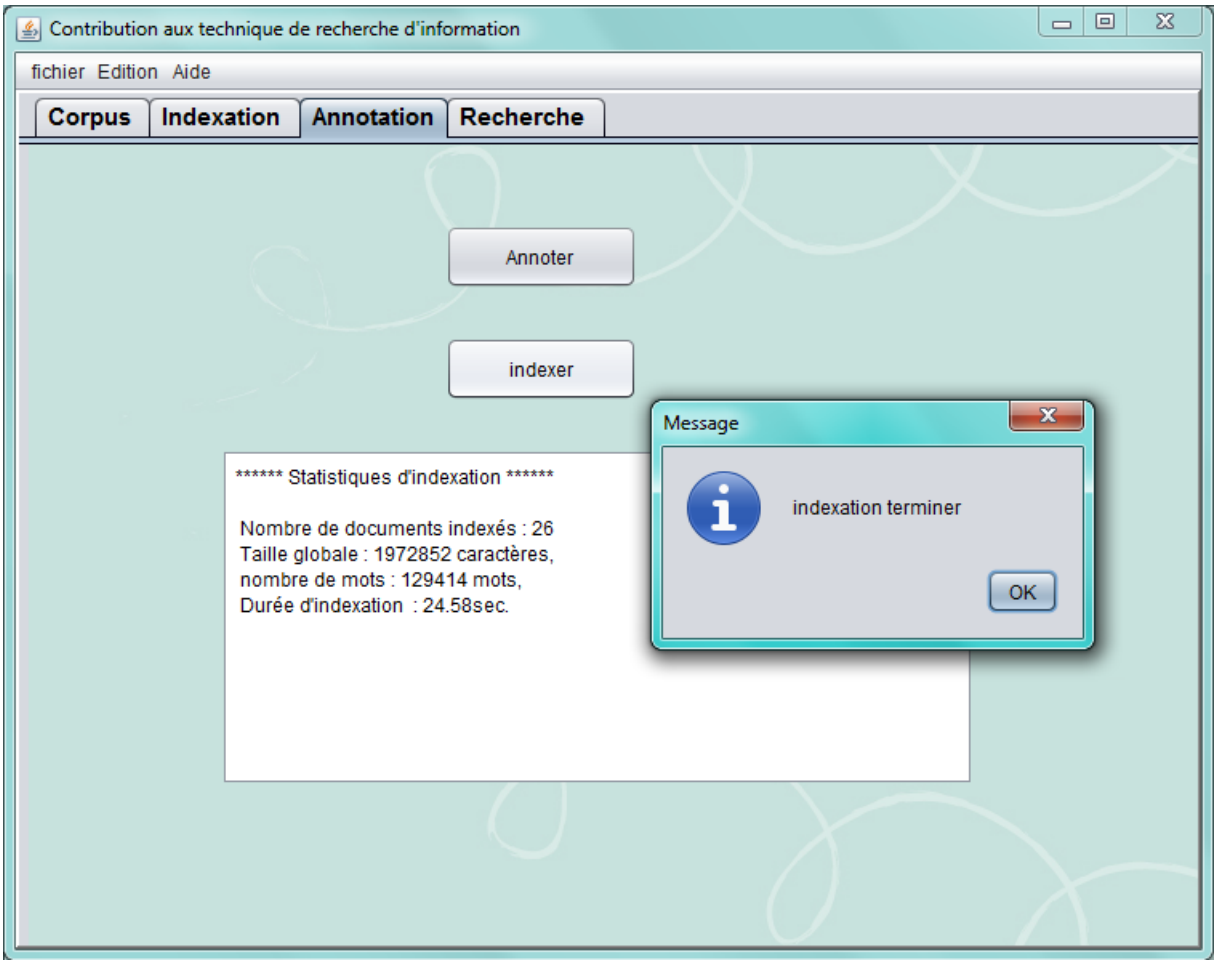


Figure 20. Fenêtre d'indexation d'un corpus annoté

IV.4.6 La phase de recherche

Le volet de recherche propose à l'utilisateur de saisir une requête libre avant d'effectuer une recherche selon les deux alternatives (simple et étendue).

Enter votre requête *:* pour saisir la requête à rechercher

Le bouton Rechercher *:* permet de lancer la recherche dans les documents indexés.

Recherche par mot clé : permet de lancer la recherche en utilisons le moteur de recherche lucene.

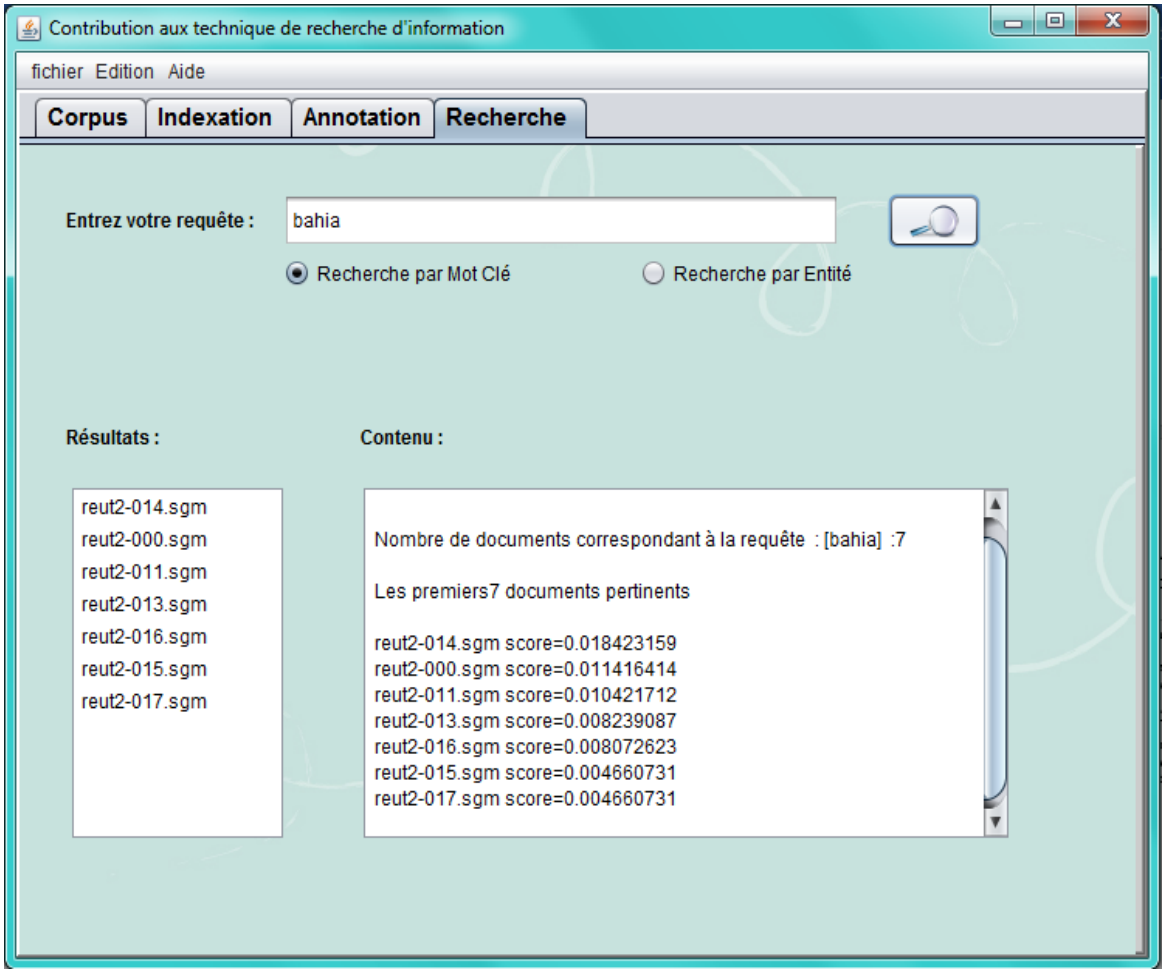


Figure 21. Fenêtre de recherche dans un corpus

IV.5 Le corpus utilisé

Collection de Reuters : Ce corpus est une collection de documents textuels téléchargée du site de l'Institut d'informatique et d'électronique Gaspard-Monge[10]. Cette collection de textes a été extraite automatiquement de la collection Reuters-21578[11]. Chaque texte est classé dans une ou plusieurs des 135 catégories utilisées dans la collection. Cette catégorisation est donnée dans un fichier nommé categorisation.txt où chaque ligne correspond à la catégorisation d'un texte. Ce corpus contient environ 20 000 fichiers et a été choisi pour sa richesse et la variété des catégories aussi.

IV.6 Conclusion

Ce chapitre a été consacré à la description conceptuelle de notre application. Différent outils et logiciels ont été intégrés. Notre implémentation a pu mettre en œuvre plusieurs concepts et approches, qui paraissaient abstraites, dans une seule interface simplifiée.

[10]http://igm.univ-mlv.fr/~mconstan/enseignement/m2pro/tal/tal-td2/node1.html
[11]Http ://www.daviddlewis.com/resources/testcollections/reuters21578/

Conclusion Générale

Le problème principal de la recherche d'information (RI) est de trouver les documents satisfaisant aux besoins d'un utilisateur (habituellement exprimés sous forme d'une requête) en termes d'informations. Afin d'atteindre cet objectif, une comparaison entre les mots contenus dans la requête et ceux représentant le contenu de chaque document doit être faite. Notre travail s'inscrit dans le cadre de l'amélioration de l'indexation et de la recherche d'information des entités nommées.

Le thème que nous avons traité intitulé « contribution aux technique de recherche d'information » rentre dans le cadre de la RI, dont l'objectif principale est d'appliquer un modèle de recherche d'information afin d'arriver à créer un système de recherche d'entité qui permet de trouver à partir de volume important d'information disponible celle qui sont pertinents à l'utilisateur.

Ce projet a fait l'objet d'une expérience intéressante, qui nous a permis d'améliorer nos connaissances et nos compétences dans le domaine de la recherche d'information.

Nous avons commencé à définir les concepts de base de la recherche d'information et la recherche d'information dans le web et ces différents types de requêtes. Ensuite nous sommes passé à expliquer le fonctionnement de système de recherche d'information tout en incluant les étapes de son processus, puis nous avons cité les modèles de RI (modèle booléen, vectoriel, probabiliste), ensuite nous sommes passé à l'extraction d'information et l'annotation, et puis nous avons détaillé l'indexation.

Dans le deuxième chapitre nous avons abordé la recherche des entités comme une solution afin de répondre au mieux à l'attente de l'utilisateur ayant un besoin en information pour cela nous avons défini la recherche d'entité et cité un exemple de motivation pour détailler notre problématique. Et nous avons présenté les travails relatifs à ce domaine. Enfin nous avons terminé par l'évaluation du système de recherche d'entité.

Ensuite, dans troisième chapitre nous avons présenté la conception de notre système de recherche d'information qui est basé sur la conception de moteur de recherche lucene et l'annotateur des entités nommées Stanford NER. On a schématisé les différentes fonctionnalités de lucene puis on est passé à la présentation de l'annotateur Stanford NER.et ensuite nous avons détaillé les étapes de notre travail.

Enfin, on a abordé le quatrième chapitre ou on a implémenté notre système de recherche d'information, on présentant ses différentes interfaces d'indexation, d'annotation et de recherche.

Les perspectives pour notre travail :

Notre système est motivé par l'apparition des systèmes d'annotation automatique de textes et permet à l'utilisateur de poser différents types de requêtes sur les documents annotés.

Trois types de requêtes sont considérés : recherche par une entité (R1E), recherche par plusieurs entités (RPE) et recherche par mots clés (RMC).

On peut avoir en réponse : des entités relatives aux requêtes, des entités pertinentes et les documents qui sont associés ses entités.

Nous proposons également de diversifier les documents relatifs aux entités retournées par type d'entité (ex. Washington interprété comme ville, personne) et par catégorie de documents (ex. Médecine, politique) pour faciliter l'interprétation de l'utilisateur.

Bibliographie

[1] Ould Hadri Imene Mansouria, « Conception Et Intégration D'un Analyseur Morphologique Arabe Dans Un Moteur De Recherche » Mémoire fin d'étude université de Mostaganem. **(2016).**

[2] Zemirli W. Nesrine, « Vers Le Développement D'Un Système De Recherche D'Information Personnalisé Intégrant Le Profil Utilisateur » Formation Doctorale en informatique Université Paul Sabatier Toulouse III **(2004).**

[3] Siham Boulaknadel, « Traitement Automatique Des Langues et Recherche D'Information En Langue Arabe Dans Un Domaine de Spécialité : Apport Des Connaissances Morphologiques et Syntaxiques Pour L'Indexation » Thèse doctorat **(2010).**

[4] A Taxonomy of Web Search. URL http://www.cis.upenn.edu/~nenkova/Courses/cis430/p3-broder.pdf

[5] Saidi Imen « Contribution Aux Technique De Recherche D'Informations » Thèse doctorat **(2015).**

[6] Introduction à La RI URL http://www.iro.umontreal.ca/~nie/IFT6255/Introduction.pdf .

[7] Fatiha Boubekeur-Amirouche « Contribution à La Définition De Modèles De Recherche D'information Flexibles Basés Sur Les CP-Nets » Thèse doctorat **(2008).**

[8] Herzallah Abdelkrim « Cours recherche d'information » Université Bouira.

[9] Rami Harrathi « Recherche D'information Conceptuelle Dans Les Documents

Semi-Structurés » Thèse doctorat **(2010).**

[11] Gianluca Demartini « From People to Entities: Typed Search in the Enterprise and the Web» **(2011)**.

[13] Luc Grivel **« La recherche d'information en contexte : Outils et usages applicatifs » livre** (1 février 2011).

[14] Abdelkrim Bouramoul « Recherche D'information Contextuelle Et Sémantique Sur Le Web » **(2011).**

[15] Kokou DEDZOE « Traitement de Requetés Top-k dans les Communautés Virtuelles P2P de Partage de Données » **(2011).**

[16] Amar-Djalil MEZAOUR « **Recherche ciblée de documents sur le web** » **LRI, Université Paris Sud.**

[17] ABBASSI MEFTAH **chapitre 1 « Les systèmes de recherche d'information »**

[18] Rosa Stern « Identification Automatique d'Entités pour l'Enrichissement de Contenus Textuels » Thèse doctorat **(2014).**

[19] Mohamed HATMI « Reconnaissance des entités nommées dans des documents multimodaux **» (2014).**

[20] Damien Nouvel, Jean-Yves Antoine, Nathalie Friburger, Arnaud Soulet « Fouille de règles d'annotation pour la reconnaissance d'entités nommées » Université François Rabelais Tours.

Webographie

[10] https://www.w3.org/Amaya/User/doc/Annotations.html Consulté le 25/01/2017

[12] URL https://staff.fnwi.uva.nl/m.derijke/content/publications/ranlp2007-fp-entity.pdf

[21]http://pageperso.lif.univmrs.fr/~andreea.dragut/enseignementWebMining/r/crs2.pdf

[22] https://www.crunchbase.com/organization/opencalais#/entity Consulté le 25/03/2017

[23] http://www.torrefacteurjava.fr/content/utiliser-apache-lucene-pour-effectuer-des-recherches-textuelles

[24] http://www.w3ii.com/fr/lucene/lucene_indexing_process.html Consulté le 24/04/2017

[25] https://nlp.stanford.edu/software/CRF-NER.shtml