REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur de la Recherche Scientifique

Université Mouloud MAMMERI de Tizi-Ouzou

Faculté de Géni Electrique et d'Informatique
Département d'Informatique

Mémoire de fin d'études

En vue de l'obtention du diplôme de MASTER EN INFORMATIQUE

Spécialité: Conduite de Projet Informatique

THEME

Extraction de concepts dans la littérature biomédicale

Dirigé et proposé par :

Mme F. AMIROUCHE

Réalisé par :

Melle LAZIB Katia

Promotion 2012-2013

Remerciements

Je tiens à remercier Dieu de m'avoir donné la force et la bonne santé, la patience, la volonté et le courage de mener à bon terme ce mémoire de fin d'étude.

Au terme de ce travail je tiens à adresser mes plus sincères et vifs remerciements et gratitudes à ma promotrice Mme AMIROUCHE pour toutes ses orientations et son l'encadrement.

J'associe à mes remerciements :

Les membres du jury, pour leur participation et pour avoir accepté de juger mon travail.

Merci

Dédicace

*J*e dédie ce travail à toute personne m'ayant aidé dans son élaboration, et qui sans leur contribution, sa réalisation aurait été presque impossible.

A ma famille, mes sœurs et mademoiselle AZZOUG Wassila qui m'a aidé théoriquement et pratiquement

A cette occasion, je remercie et j'exprime ma gratitude à toutes les personnes qui ont contribués de prés ou de loin à la réalisation de ce travail.

Katia

Sommaire

Table des figures

Introduction générale

De nos jours, la recherche d'information (RI) est une activité primordiale qui répond à des besoins diversifiés de l'utilisateur comme la consultation de l'information sur des services Web.

Cependant, avec le développement de nouvelles technologies de l'information et de la communication, les sources d'information deviennent de plus en plus abondantes, et rendent la recherche de plus en plus fastidieuse. Ceci est particulièrement vrai dans le domaine médical où l'information est fortement abondante et fortement disséminée à travers diverses sources d'information publiées, à titre d'exemple, la base de données MEDLINE (Medical Literature Analysis and Retrieval System Online) qui contient plus de 20 millions de citations (résumés) issues d'articles de multiples journaux et ouvrages scientifiques biomédicaux publiés. Pour faciliter l'accès à l'information biomédicale, l'indexation par un vocabulaire biomédical contrôlé, issu de diverses terminologies développées à cet effet, a été perçue comme une première solution. Dans le domaine biomédical, les ressources terminologiques les plus utilisées pour l'indexation sont : MeSH (Medical Subject Headings), SNOMED (Systematized Nomenclature of Medicine), ICD-10 (International Classification of Diseases), UMLS (Unified Medical Language System) ... Cependant, cette solution est insatisfaisante du fait que le vocabulaire est par nature

ambigu, et que cette ambiguïté implique que les résultats de la recherche ne sont souvent pas conformes aux besoin des utilisateurs.

Pour remédier aux limites de l'indexation classique basée sur des mots clés, deux alternatives ont été proposées: l'indexation sémantique et l'indexation conceptuelle (Mihalcea et al., 2000). Le terme d'indexation sémantique désigne une indexation basée sur les sens des mots, tandis que l'indexation conceptuelle se base sur des méthodes d'identification de concepts dans un corpus textuel en utilisant une ressource sémantique externe comme les ontologies. Nous considérons dans la suite que l'indexation conceptuelle peut être vue comme une généralisation de l'indexation sémantique, dans la mesure où les concepts aussi véhiculent des sens.

L'indexation conceptuelle est généralement basée sur deux étapes : une première étape d'extraction de concepts et une seconde étape d'extraction de relations entre concepts.

L'extraction de concepts est une étape primordiale qui consiste à extraire à l'aide de ressources terminologiques les concepts qui représente au mieux un document.

Notre travail s'inscrit dans ce contexte et consiste en la proposition d'une approche d'extraction efficace de concepts à partir de la littérature biomédicale. En particulier, nous proposons d'utiliser une approche d'extraction de concepts dans la littérature générale (Boubekeur et al., 2011) et de l'adapter à la littérature biomédicale. Dans cette adaptation

que nous proposons, nous remplaçons la ressource terminologique WordNet initialement utilisée par une ressource terminologique biomédicale normalisée, le thésaurus MeSH.

Ce mémoire est organisé en quatre chapitres :

Chapitre 1 : Dans ce chapitre, nous décrivons les concepts de base de la RI et en particulier le processus d'indexation. Puis nous nous intéresserons à l'indexation conceptuelle basée sur l'extraction de concepts dans la littérature biomédicale.

Chapitre 2 : Dans ce chapitre, nous décrivons la recherche d'information biomédicale, ainsi que les principales approches d'extractions de concepts.

Chapitre 3 : Dans ce chapitre, nous présentons notre approche d'extraction de concepts biomédicaux, et nous la déroulons sur un exemple détaillé.

Chapitre 4 : Dans ce chapitre, nous détaillons l'implémentation et la mise en œuvre de notre approche, ainsi que les résultats de son évaluation.

Nous conclurons ce travail par une conclusion générale et des perspectives.

Chapitre I : De la RI classique à la RI conceptuelle

I.1 Introduction

La recherche d'information (RI) est une branche de l'informatique qui s'intéresse à la collecte, à l'organisation et à la sélection d'information répondant aux besoins des utilisateurs. La RI est une discipline relativement ancienne qui s'intéresse à répondre pratiquement à la problématique suivante : *comment retrouver un document parmi une collection de documents qui satisfait un besoin utilisateur ?*

La RI est mis en œuvre à travers un système de recherche d'information (SRI) qui offre des techniques et outils permettant de localiser et de visualiser l'information pertinente, répondant à un besoin en information exprimé par un utilisateur sous forme de requête.

L'indexation est une phase très importante pour un SRI car de sa qualité dépend la qualité des réponses du système et donc les performances de ce dernier. Afin de palier l'ambiguïté de la langue dans l'indexation classique du à une indexation par entités lexicales du vocabulaire, de nouvelles approches ont été introduites qui s'intéressent principalement à la représentation des documents et requêtes par des concepts plutôt que par les mots eux mêmes.

Le présent chapitre a pour objectif d'introduire la RI classique, puis l'indexation conceptuelle.

I.2 Processus général d'un SRI

L'objectif d'un SRI est de fournir à partir d'une collection de documents (ou fond documentaire) l'ensemble de documents sensés répondre à un besoin exprimé à travers une requête. Pour cela, le SRI met en œuvre un processus d'appariement qui permet de réaliser la mise en correspondance des informations contenues dans un fond documentaire d'une part, et des besoins en information des utilisateurs d'autre part. La Figure I.1 qui représente les étapes du processus général d'un SRI.

La figure fait ressortir trois processus : le processus d'indexation, le processus d'appariement et le processus de reformulation.

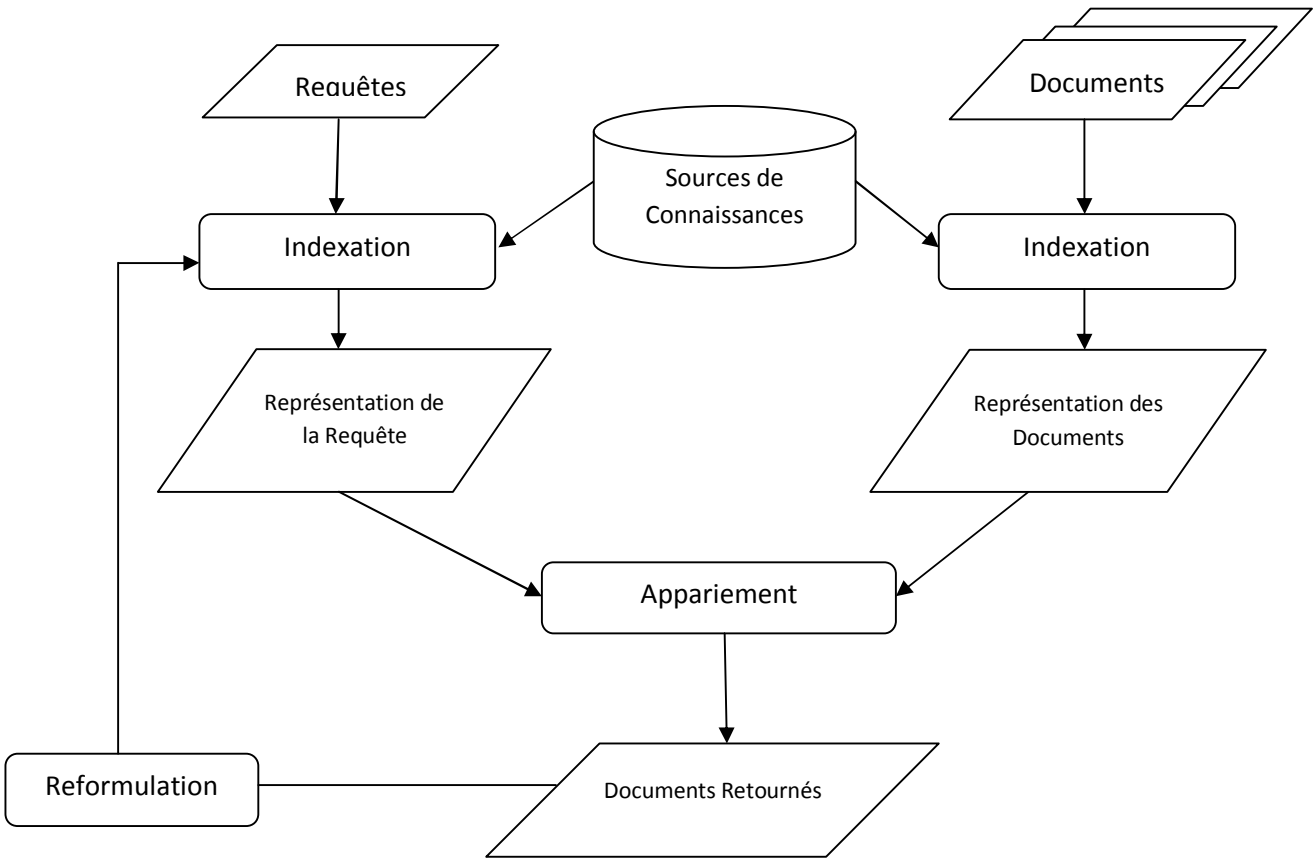


Figure I.1 : Processus général d'un SRI

§ Le processus d'indexation

L'indexation est un processus permettant d'extraire d'un document ou d'une requête, une représentation paramétrée qui couvre au mieux son contenu sémantique. En pratique, cette représentation est définie par l'ensemble des termes les plus importants (appelé aussi descripteurs) du document (ou de la requête).

L'indexation peut être :

- Manuelle.
- Semi-automatique.
- Automatique.

Dans une *indexation manuelle*, chaque document de la collection est examiné par un spécialiste du domaine ou un documentaliste afin d'identifier les descripteurs (Baziz, 2005), (Roussey, 2001).

Dans une *indexation automatique*, le processus d'indexation, entièrement informatisé, met en œuvre des méthodes et des techniques issues de Traitement Automatique de la Langue Naturelle (TALN) pour identifier et extraire les descripteurs du document (ou de la requête). Comparés aux résultats obtenus par une indexation manuelle, les résultats obtenus par une indexation automatique sont souvent jugés insatisfaisants (Jacquemin et al., 2002). Pour remédier à cette limite le plus souvent, en couple l'indexation automatique avec l'indexation manuelle (Jacquemin et al., 2002). Pratiquement, les résultats obtenus par une indexation automatique sont soumis à un documentaliste pour les valider ou pour les enrichir. Ce type d'indexation est appelé *indexation semi-automatique* ou indexation supervisée.

Bien que jugée non entièrement satisfaisante, l'indexation automatique reste l'approche la plus utilisée en RI pour des raisons pratique évidentes. Nous la détaillons dans le paragraphe suivant.

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

- **L'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 (signes de ponctuations, espaces,….)
- **L'élimination des mots vides :** Les mots vides sont en général les mots outils de la langue, souvent des prépositions (de, à), pronoms (aucun, tout, on), certains adverbes (ailleurs, maintenant), adjectifs (certain, possible), etc. Ces mots sont éliminés par l'utilisation d'un anti-dictionnaire (ou Stoplist). La Stoplist est consultée lors du processus d'indexation : si un terme existe dans la Stoplist, il sera systématiquement éliminé du texte sinon il sera considéré comme terme d'index.
- **La lemmatisation :** 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 (Boubekeur, 2008).
- **La pondération des termes :** La pondération est une fonction très importante en RI, elle permet de mesurer l'importance d'un terme dans un document. Les techniques de pondérations les plus utilisées sont basées sur les facteurs Tf et Idf, qui permettent de combiner les pondérations locale et globale d'un terme :

  – **Tf (Term Frequency)** : C'est la fréquence du terme dans un seul document (pondération locale).

  – **Idf (Inverse of Document Frequency)** : C'est la fréquence du terme dans un ensemble de documents (pondération globale).

La pondération d'un terme ti dans un document dj est souvent notée $TF * IDFij$. Elle est donnée par le produit de la pondération locale (reflète l'importance locale du terme dans le document) de ti dans dj par sa pondération globale (exprime l'importance globale du terme dans la collection) dans la collection des documents.

§ Le processus d'appariement document – requête :

La comparaison entre un document et une requête revient à calculer un score ou fonction de similarité *RSV* (Q,d) *(Retrieval Status Value),* qui représente la pertinence entre le document d et la requête Q.

La fonction de similarité permet ensuite d'ordonner les documents renvoyés à l'utilisateur.

La qualité de cet ordonnancement est primordiale, car l'utilisateur se contente généralement d'examiner les premiers documents renvoyés. Si les documents recherchés ne sont pas présents dans cette tranche, l'utilisateur considérera le SRI comme mauvais vis-à-vis de sa requête.

La mesure de pertinence d'un document vis-à-vis d'une requête est formellement définie par le modèle de RI. On distingue trois principaux modèles en RI :

- **Le modèle booléen :** est le modèle le plus simple des modèles de RI, c'est aussi le premier qui s'est imposé dans le monde de la recherche d'information. Il est basé sur la théorie des ensembles et l'algèbre de Boole. Il considère que les termes de l'index sont présents ou absents d'un document. Le document est représenté par un ensemble de termes reliés par l'opérateur « et », soit :

$$D = t_1 \wedge t_2 \wedge \ldots \wedge t_n$$

Une requête Q est un ensemble de mots avec des opérateurs booléens : AND(∧), OR(V), NOT(¬).

Exemple : Q= (t1∧t2) V (t3 V¬ t4).

Pour qu'un document corresponde à une requête, il faut que l'implication d ⇒ Q soit valide.

La similarité entre un document et une requête est définit par :

RSV (Q, d) =1 si d appartient à l'ensemble décrit par la requête, 0 sinon.

Ainsi le modèle booléen affirme que chaque document est soit pertinent, soit non-pertinent sans autre possibilité d'ordonnancement.

- **Le modèle vectoriel :** le modèle vectoriel a pour but d'arriver à retourner une liste ordonnée de documents selon un degré mesuré par la pertinence d'un document vis-à-vis d'une requête. Contrairement au modèle booléen où les termes de la requête doivent être reliés par des connecteurs logiques, l'utilisateur peut ici exprimer son besoin en information en langage naturel ou sous forme d'une liste de mots clés.

  La mesure de similarité entre le document fourni et la représentation des documents de la collection est utilisée pour ordonner ces documents. Le critère de similarité défini est que "plus deux représentations contiennent les mêmes éléments, plus la probabilité qu'elles représentent la même information est élevée". Une telle définition revient en fait à compter le nombre d'éléments que partagent la requête et la représentation du document.

  Pour ce faire, considérant la représentation d'un document comme un vecteur $\overrightarrow{dJ}$= {$w_{1,j}$,$w_{2,j}$,. . . , $w_{t,j}$} où wi,j est le poids (0 ou 1) des termes dans le document, t étant le nombre total de termes de l'index, et considérant la représentation de la requête comme un vecteur $\vec{q}$ = {$w_{1,q}$,$w_{2,q}$, . . . ,$w_{t,q}$}, avec les mêmes notations, la mesure de similarité la plus simple est alors le produit scalaire :

$$\mathrm{RSV}(\vec{d}_j, \vec{q}) = \sum_{i=1}^{t} w_{i,j} * w_{i,q}$$

- **Le modèle probabiliste :** Le premier modèle probabiliste a été proposé par Maron et Kuhns au début des années 1960. Le principe de base consiste à présenter les résultats de recherche d'un SRI dans un ordre basé sur la probabilité de pertinence d'un document vis-à-vis d'une requête.

  Pour ce faire, le processus de décision complète le procédé d'indexation probabiliste en utilisant deux probabilités conditionnelles :

  P (w_{ji} */pert) :* Probabilité que le terme t_i occure dans le document D_j sachant que ce dernier est pertinent pour la requête.

  P (w_{ji} */Nonpert) :* Probabilité que le terme t_i occure dans le document D_j sachant que ce dernier n'est pas pertinent pour la requête.

  Robertson (Robertson, 94b) définit son modèle PRP (Probability Ranking Principle), sur ce même principe. Etant donnée une requête utilisateur notée Q et un document d, le modèle probabiliste tente d'estimer la probabilité que le document d appartienne à la classe des documents pertinents (non pertinents). Un document est alors sélectionné si la probabilité qu'il soit pertinent à Q, notée P(R/d), est supérieure à la probabilité qu'il soit non pertinent à Q, notée P (NR/d). Le score d'appariement entre le document d et la requête Q, noté RSV (d,Q), est donne par :

$$RSV(d,Q) = \frac{P(R/d)}{P(NR/d)}$$

§ Le processus de reformulation de la requête :

Afin de faire correspondre au mieux la pertinence utilisateur et la pertinence système, une étape de reformulation de requête est souvent utilisée. Ce processus consiste généralement à modifier la requête initiale de l'utilisateur (*relevance feedback*) ou à étendre la requête initiale (*query expansion*). En effet l'utilisateur doit formuler sa requête et la soumettre au système de RI. La plupart des utilisateurs ont des difficultés pour formuler leur requête de sorte que les premiers documents retournés par le système de RI soient pertinents. Le problème vient du fait que le langage (les mots-clés) de l'utilisateur est souvent différent du langage utilisé dans la collection. La reformulation de la requête est donc un processus ayant pour objectif d'améliorer les performances du SRI et donc la précision dans les réponses du système.

I.3 Evaluation des Systèmes de Recherche d'information

L'évaluation des SRI est une étape importante qui permet de mesurer la performance du système. Son principal aspect concerne la capacité du système à sélectionner le maximum de documents pertinents et un minimum de documents non pertinents.

Elle est principalement basée sur deux paramètres : le rappel et la précision.

- La précision mesure la proportion de documents pertinents retrouvés parmi tous les documents retrouvés par le système.
- Le rappel mesure la proportion de documents pertinents retrouvés parmi tous les documents pertinents dans la base.

Formellement :

$$Précision = \frac{|DPR|}{|DR|}$$

$$Rappel = \frac{|DPR|}{|DP|}$$

Où :

- P : est l'ensemble des documents de la collection qui sont pertinents vis-à-vis à la requête,
- R : est l'ensemble des documents retournés par le SRI,
- PR : est l'ensemble des documents pertinents vis-à-vis de la requête et qui sont retournés par le SRI,

Pour comparer des SRI, il est naturel que les tests soient réalisés sur les mêmes collections. Ces collections sont fournies par des campagnes d'évaluation. En plus des collections, la campagne fournit des requêtes résolues c'est-à-dire des requêtes pour lesquelles on connait les documents pertinents.

Le protocole le plus utilisé pour évaluer les performances de la RI est celui de TREC[1] (Text Retrieval Conference) qui est un des fournisseurs principaux des collections de *test* pour évaluer les performances des SRI.

I.4 Indexation conceptuelle en RI

L'indexation conceptuelle consiste à représenter des documents et requêtes par des concepts au lieu de mots simples comme c'est le cas dans l'indexation classique. Ces concepts sont extraits d'ontologies et autres ressources linguistiques. L'indexation

[1] http://trec.nist.gov/

conceptuelle se base essentiellement sur l'extraction de concepts et l'extraction de relations entre concepts. Nous nous intéressons pour notre part à l'extraction de concepts.

I.4.1 Notions élémentaires

Dans ce qui suit, nous donnons quelques définitions et vocabulaires que nous rencontrons dans tout ce qui suivra.

I.4.1.1 Terminologie

Une terminologie est l'ensemble des termes, rigoureusement définis, qui sont spécifiques d'une science, d'un domaine particulier. C'est une discipline qui a pour objet l'étude théorique des dénominations des objets ou des concepts utilisés par tel ou tel domaine du savoir. Le principal intérêt d'une terminologie est de réduire, voire supprimer, l'ambiguïté qui existe entre les termes d'un domaine et de pouvoir donc, mieux partager de l'information.

I.4.1.2 Ontologie

La définition la plus citée présente une ontologie comme étant « une spécification explicite et formelle d'une conceptualisation partagée » (Gruber, 1993). En d'autre terme, une ontologie est une représentation formelle d'un domaine. C'est une conceptualisation dans le sens ou elle fournit un vocabulaire formalisé de concepts et de leurs relations. L'objectif principal d'une ontologie est de partager, réutiliser des connaissances propres à un domaine donné. Les ontologies peut être employées dans l'intelligence artificielle, le Web sémantique, le génie logiciel, l'informatique biomédicale, comme une forme de représentation de la connaissance au sujet d'un monde ou d'une certaine partie de ce monde.

Les ontologies les plus utilisées sont WordNet dans le domaine général et aussi le méta thesaurus UMLS, Gene Ontology (GO), et MeSH dans le domaine biomédical.

I.4.1.3 Thésaurus

Un thésaurus est un répertoire alphabétique qui réunit les termes et mots clés techniques pour le classement documentaire.

Un thésaurus est dit aussi comme un langage documentaire fondé sur une structuration hiérarchisée des termes désignant les concepts. Les termes y sont organisés de manière conceptuelle et reliés entre eux par des relations sémantiques. Plus concrètement, un thésaurus est un ensemble structuré de termes d'un vocabulaire, par exemple les termes techniques utilisés en médecine, représentés de façon normalisée par des descripteurs ou des mots-clés (Foskett, 1997).

Il fournit des informations sur chaque terme et ses relations à d'autres termes, il peut être construit manuellement par les experts (linguistes, documentalistes, bibliothécaires) ou automatiquement à partir d'une collection de documents.
Un thésaurus médical est une représentation d'un ensemble de concepts médicaux, chaque concept est décrit par un ou plusieurs termes synonymes.
Dans un thésaurus multilingue, un concept sera représenté par des termes dans chaque langue.

I.4.2 Ressources terminologiques pour l'indexation

L'une des ressources les plus exploitées pour l'indexation conceptuelle dans la littérature générale est la base lexicale WordNet.

§ WordNet

WordNet[2] est une base lexicale électronique développée en 1985 à l'université de Princeton par une équipe de psycholinguistes et de linguistes sous la direction de G. Miller (FELLBAUM, 1998). A l'origine WordNet est conçu comme une base lexicale. Ensuite, WordNet a été perçu comme un réseau sémantique. Dans ce réseau sémantique, chaque nœud représente un concept. Un nœud est constitué par un ensemble de termes synonymes (ou synsets). Ces termes désignent le concept représenté par le nœud. Dans WordNet, les concepts sont reliés

[2] http ://www.cogsci.princeton.edu/~wn/

par des relations sémantiques. Telles que la relation de composition méronymie-holonymie (*part of*) et la relation hyponymie-hyperonyme (*is-a*) (FELLBAUM, 1998).

– **Hyperonymie** désigne une classe de concepts englobant des instances de classes plus spécifiques : *Y* est un hyperonyme de *X* si *X* est un type de *Y*. Par exemple, "fruit" est un hyperonyme de "pomme" et de "cerise".
– **Hyponymie** désigne un membre d'une classe de concepts : *X* est un hyponyme de *Y* si *X* est un type de *Y.* Par exemple, "France" est hyponyme de "pays", "cheval" est hyponyme de "animal".
– **Holonymie** est le nom de la classe globale dont les noms méronymes font partie. *Y* est un holonyme de *X* si *X* est une partie de *Y*. Par exemple, "corps" est un holonyme de "bras", de même que "maison" est un holonyme de "toit".
– **Méronymie** est le nom d'une partie constituante, substance de ou membre d'une autre classe : *X* est un méronyme de *Y* si *X* est une partie de *Y*. Par exemple, "voiture" a pour méronymes "porte", "moteur", "roue", etc.

Dans sa version 3.0 WordNet contient 155287 termes organisés en 117659 synsets.

Couvrant la majorité des noms, verbes, adjectifs et adverbes de la langue Anglaise. Sa dimension ainsi que le domaine de la langue générale qu'elle traite lui permettent souvent de couvrir les sujets traités dans les collections de test conventionnelles de la RI (TREC, CLEF).

Dans (Baziz, 2005), l'indexation conceptuelle a été définie comme une approche d'indexation basée sur le *sens des mots* (appelée *sense-based indexing* en anglais). Plusieurs travaux d'indexation conceptuelle ont été menés dans ce sens comme (Baziz, 2005; Boubekeur, 2008) qui utilisent tous deux la base lexicale WordNet pour identifiée les concepts représentatifs des documents

Dans (Boubekeur et al., 2011) les auteurs propose une approche d'indexation conceptuelle des documents. Qui utilise WordNet comme ressource linguistique afin de retrouver les concepts représentatifs du contenu d'un document. Leur contribution se porte sur trois aspects: une approche d'identification des concepts en utilisant la base lexicographique WordNet, une approche de désambiguïsation à deux niveaux, basée sur l'utilisation conjointe de WordNetDomains et de WordNet, et une approche de pondération des concepts basée sur une nouvelle notion d'importance. Dans la suite de notre travail nous utilisons cette approche d'identification des concepts que nous adaptons pour le domaine biomédical.

I.6 Conclusion

Dans ce chapitre nous avons présenté les concepts généraux de la RI, ainsi que les méthodes et modèles fondamentaux utilisée en RI.

Dans le chapitre suivant, nous nous intéressons à la recherche d'information et à l'indexation conceptuelle dans le domaine biomédical, objet de notre étude.

Chapitre II : Indexation conceptuelle dans la littérature Biomédicale

II.1 Introduction

Du fait des énormes progrès réalisés tant dans le domaine biomédical que dans la technologie de l'information, le domaine biomédical a montré une évolution significative du volume d'information stockée. La recherche d'information dans ce volume grandissant est de plus en plus lourde.

L'indexation en RI biomédicale a pour objectif de faciliter l'accès à la littérature biomédicale en affectant à chaque document une liste de termes désignant les concepts issus d'une ou des terminologies biomédicales (Névéol *et al.*, 2006; Darmoni *et al.*, 2009). Ainsi l'indexation conceptuelle est venu comme une solution pour l'indexation classique elle permet de mieux organiser, de structurer et surtout de faciliter l'accès à l'information biomédicale. Elle utilise des ressources terminologiques comme outils afin de mettre en œuvre cette dernière. Les ressources les plus utilisées pour cela sont : MeSH (Medical Subject Headings), SNOMED (Systematized Nomenclature of Medicine), UMLS (Unified Medical Language System) ...

Dans ce chapitre, nous allons définir les spécificités de la littérature biomédicale puis nous introduisons les ressources terminologiques utilisées dans ce dernier, aussi nous présentons l'une des étapes de base de l'indexation conceptuelle qui est l'extraction de concepts qui est fait l'objet de notre étude.

II.2 La littérature biomédicale

La littérature biomédicale est un ensemble d'informations issues essentiellement des bases de données bibliographiques qui font références à des revues, articles scientifiques, livres et des comptes-rendus des conférences dans le milieu biomédical à l'exemple de MEDLINE (Medical Literature Analysis and Retrieval System Online) qui contient plus de 21 millions de références d'articles en sciences de la vie, notamment de la biomédecine. Cette base a été créée et gérée par la National Library of Medicine (NLM) aux États-Unis regroupant des références d'une vingtaine de millions d'articles scientifiques indexés depuis 1966 jusqu'à aujourd'hui, principalement en langue anglaise. Les documents de MEDLINE ont été utilisés pour évaluer les performances des systèmes de RI biomédicale dans le cadre de TREC Genomics de 2003 à 2005. L'interface PubMed permet de consulter gratuitement la base de données à partir d'un navigateur web.

II.3 Ressources terminologiques biomédicales

Dans cette section, nous présentions quelques ressources terminologiques utilisées dans la recherche d'information biomédicale.

3.1 Thésaurus MeSH

Afin d'indexer, classer et rechercher des documents (notamment ceux de la base MEDLINE), la NLM a créé en 1954 le thésaurus MeSH (Medical Subject Headings). Depuis, il a été régulièrement mis à jour. La traduction de MeSH vers le français est assurée par l'INSERM[1] qui met la version bilingue à la disposition de la communauté francophone. MeSH comprend essentiellement des termes qui désignent les concepts biomédicaux, des relations, des descripteurs et des qualificatifs.

[1] Institut Nationale de la Santé Et de la Recherche Médicale

- **Terme :** Un terme est un mot ou un ensemble de mots exprimant une notion particulière,

- **Relation** : MeSH est basée sur des relations hiérarchiques. La hiérarchie est représentée par un code reflétant l'arborescence auquel le concept appartient et peut véhiculer plusieurs sens :

  1. relation "est une partie de" (méronymie), par exemple le concept "*doigt*" (A01.378.800.667.430) est une partie de "*main*" (A01.378.800.667).

  2. relation "est un type de" (hyponymie), par exemple le concept *pouce* (A01.378.800.667.430.705) est un type de "*doigt*" (A01.378.800.667.430).

- **Descripteur** : Un descripteur est constitué d'un ou de plusieurs concepts de significations proches et porte le nom d'un de ces concepts, dit préféré. Les autres concepts, dits subordonnés. Tous les termes d'un descripteur sont des équivalents documentaires ou termes d'entrée et renvoient au descripteur dans le cadre de l'indexation et de la recherche documentaire. Les descripteurs MeSH sont répartis en 16 catégories recouvrant la biologie, l'anatomie, l'organisme..etc. Chaque catégorie de descripteurs est structurée en arborescence pouvant comprendre jusqu'à onze niveaux de hiérarchie. Certains descripteurs ont une seule localisation, alors que les autres en ont plusieurs, dans la même catégorie ou dans des catégories différentes. Chaque localisation est représentée par un code alphanumérique, la lettre indiquant la catégorie et la séquence numérique précisant la localisation dans la hiérarchie. La figure qui suit représente les différentes catégories MeSH.

[A]	Anatomie
[B]	Organisme
[C]	Maladies
[D]	Produits chimiques et pharmaceutiques
[E]	Techniques analytiques, diagnostiques, thérapeutiques et équipements
[F]	Psychiatrie et psychologie
[G]	Phénomènes et processus
[H]	Disciplines et professions
[I]	Anthropologie, enseignement, sociologie, et phénomènes sociaux
[J]	Technologie, industrie et agriculture
[K]	Sciences humaines
[L]	Sciences de l'information
[M]	Individus
[N]	Santé
[V]	Caractéristiques d'une publication
[Z]	Lieux géographiques

Figure II.1 : Les différentes catégories de MeSH

Un exemple de hiérarchie dans le domaine A est donnée dans la figure qui suit :

A <Anatomie>

A01 <Régions du corps>

A01.378 <Membre>

A01.378.800 <Membre supérieur>

A01.378.800.667 <Main>

A01.378.800.667.430 <Doigts>

A01.378.667.430.705 <Pouce>

A01.378.800.750 <Epaule>

A01.378.610 <Membre inférieur>

A01.456 <Tête>

A03 <Système digestif>

Figure II.2 : Extrait de l'arborescence A (domaine 'Anatomie') de MeSH.

3.2 Méta-thésaurus UMLS

Développé par la NLM, UMLS (Unified Medical Language System) est un méta-thésaurus multilingue qui couvre le domaine médical. Cette ressource a été crée dans le but de faciliter la recherche et l'intégration d'informations provenant des multiples sources d'information biomédicales électroniques (NLM, 2009). Il résulte de la fusion de nombreuses ressources (plus de 110), thésaurus et terminologies, décrites dans différentes langues. Le *National Library of Medicine* (NLM) réalise et maintient cette ressource. C'est actuellement l'une des ressources les plus importantes en termes de taille et de couverture pour la médecine. Elle contient plus de 1 million de concepts reliés à plus de 5,5 millions de termes dans 17 langues. L'anglais est la langue la plus représentée avec 68% du vocabulaire, l'allemand couvre 2,84 % du vocabulaire et le français 2,55%. UMLS réunit trois bases de connaissances : un méta-thesaurus, un réseau sémantique, et le SPECIALIST Lexicon qui peuvent être utilisés séparément ou ensemble.

– **Le méta-thesaurus** constitue la base unifiée des concepts issus de plus de 150 terminologies biomédicales (dont MeSH, CIM-10 et SNOMED) en 17 langues. UMLS regroupe les différents termes synonymes (issus des différentes terminologies) sous un même concept.

– **Le réseau sémantique** fournit une catégorisation cohérente de tous les concepts représentés dans le méta-thésaurus UMLS et donne aussi un ensemble de relations utiles entre ces concepts. Le réseau offre également des informations sur l'ensemble des types sémantiques, ou catégories, qui peuvent être affectés à ces concepts, et il définit l'ensemble des relations qui peuvent exister entre eux. Il contient 133 types sémantiques et 54 relations.

– **Le SPECIALIST Lexicon** contient les informations syntaxiques, lexicales et orthographiques nécessaires au traitement automatique de la langue anglaise.

II.4 Extraction des concepts biomédicaux :

L'extraction de concepts biomédicaux est un processus de reconnaissance des expressions pertinentes dans un document afin de mettre en évidence des sujets les plus significatifs du document. Ceci est une des tâches les plus difficiles qui représentent de grands défis pour les chercheurs dans le domaine de l'extraction d'information (Cowie et Lehnert, 1996; Gaizauskas *et al.*, 2000; Krauthammer et Nenadic, 2004). Malgré la grande disponibilité des ressources terminologiques qui ont été développées, plusieurs travaux de recherche ont rapporté qu'il existe un nombre de termes qui ne sont pas pertinents pour le document même s'ils sont définis dans la ressource (Gaizauskas *et al.*, 2000; Hirschman *et al.*, 2002; Tuason *et al.*, 2004). En général, les termes extraits du document doivent être vérifiés et confirmés par un ou plusieurs experts du domaine en raison de l'ambiguïté des termes. Dans certains cas cette étape de validation par un expert humain est remplacée par une étape de désambiguïsation automatique des sens des termes.

4.1 Principales approches d'extraction de concepts :

Nous développerons dans les sections suivantes les différentes approches d'extraction automatique des concepts, qui peuvent être divisées en quatre catégories :

- *approches basées sur les règles linguistiques*,
- *approches basées sur l'apprentissage automatique*,
- *approches basées sur des mesures statistiques*.
- *approches basées sur la recherche dans un dictionnaire*,

4.1.1 Approches basées sur des règles linguistiques :

Les méthodes d'extractions basées sur des règles linguistiques consistent à définir des règles pour décrire les termes techniques qui désignent les concepts d'un domaine particulier.

En général des règles sont définis manuellement par des linguistes et experts du domaine en se basent sur les caractéristiques de la langue. Par exemple, le système décrit dans (Fukuda *et al.*, 1998), appelé PROPER (PROtein Propernoun phrase Extracting Rules), utilise les caractéristiques de la description des noms propres dans les documents médicaux et biologiques, et ne nécessite aucun dictionnaire de termes spécifiques. Les auteurs ont observé que les mots caractéristiques contenant des majuscules, des chiffres numériques et de symboles spéciaux sont souvent utilisés pour décrire les noms de protéines. D'autres systèmes d'extraction à base de règles simples utilisent des caractéristiques orthographiques et lexicales pour reconnaître les noms de protéines (Hou, 2003; Narayanaswamy *et al.*, 2003).

L'avantage des méthodes d'extraction de concepts basées sur des règles est que les règles sont capables de reconnaître les termes en fonction des caractéristiques orthographiques et lexicales d'une langue spécifique. Ceci peut être facilement fait en utilisant des expressions régulières. Toutefois, ces approches sont connues pour être extrêmement coûteuses pour le développement, et nécessitent les connaissances linguistiques des développeurs.

4.1.2 Approches basées sur l'apprentissage automatique :

Les approches basées sur l'apprentissage automatique utilisent des corpus manuellement annotés pour entraîner les classificateurs qui considèrent plusieurs caractéristiques des instances textuelles pour associer les termes techniques à des classes prédéfinies en exploitant plusieurs caractéristiques dans les instances textuelles. Par exemple les modèles de Markov cachés (HMM – Hidden Markov Models) ont été utilisés pour extraire les noms des gènes et des protéines (Collier *et al.*, 2000; Morgan *et al.*, 2003; Shen *et al.*, 2003). Chaque HMM, qui est essentiellement un automate à *n* états, se compose essentiellement d'un ensemble de paramètres (*cf.* la figure III.6) :

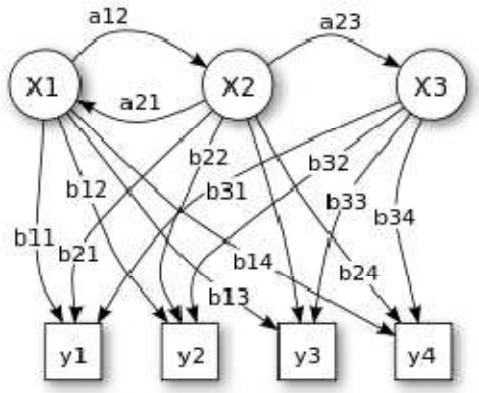


Figure II.3 : paramètres d'un modèle de Markov caché

– *x* : ensemble des états,
– *y* : observations possibles,
– *a* : probabilités de transition d'un état à l'autre,
– *b* : probabilités d'émission.

Les auteurs on intégré les caractéristiques orthographiques (par exemple, "termes composés des lettres majuscules et des chiffres", "termes commencés par une lettre majuscule", etc.) au modèle HMM.

Cependant, les méthodes basées sur l'apprentissage supervisé sont confrontés à une grande difficulté comme l'insuffisance de données car les données d'entraînement ne sont pas toujours disponibles et ne sont pas facilement mises à jour.

4.1.3 Approches basées sur des mesures statistiques :

Les approches basées sur des mesures statistiques ont été proposé pour identifier des termes complexes qui sont des séquences de mots qui se suivent et dont la probabilité d'apparition ensemble est plus fréquente que par le simple fait du hasard.

Dans (Frantzi et al., 2000) les auteurs proposent une méthode d'extraction de termes techniques appelée C-NC value, pour identifier automatiquement des termes dans plusieurs domaines (e.g., arts, médecine). Cette méthode a été utilisée plus tard pour reconnaître les termes désignant les concepts biomédicaux de la littérature (Hliaoutakis *et al.*, 2009). Cette méthode combine les informations statistiques et linguistiques pour extraire des termes et des sous-termes.

En premier lieu les mesures *C/NC-value* sont définies pour déterminer si une chaîne de caractères *t* est un terme ou un sous-terme significatif ou non.

La partie C_{value}, qui indique l'importance du terme candidat dans le texte, est calculée comme suit :

$$C_{value}(t) = \begin{cases} log_2|t| \times f(t) & si\ t\ nest\ pas\ un\ sous-terme \\ log_2|t| \times \left(f(t) - \frac{\sum_{b \in Tt} f(b)}{P(Tt)}\right) & sinon \end{cases}$$

Où:

– |t| est la longueur du terme *t*,

– *f*(t) est la fréquence du terme *t* dans le corpus,

– *Tt* désigne l'ensemble de termes contenant t,

– *P*(*t*) désigne le nombre de termes contenant t.

Ensuite la partie *NC*, qui prend en compte le contexte du terme candidat *t* et qui affecte un poids à chaque mot composant de *t* (appelé mot contextuel), est calculée comme suit :

$$NC_{value}(t) = 0.8 \times C_{value}(t) + 0.2 \times \sum_{b \in Ct} ft(b) \times \frac{t(b)}{n}$$

Où :

– *b* est le mot contextuel (nom, verbe ou adjectif) du terme candidat *t*,

– *Ct* est le contexte du terme candidat *t* (ensemble de mots contextuels uniques de *t*),

– *t*(*b*) est le nombre de termes contenant le mot contextuel *b*,

– *n* est le nombre total des termes considérés,

– *ft*(*b*) est le nombre de co-occurrences du mot *b* qui apparaît comme un mot contextuel du terme candidat *t*, c.-à-d. le nombre de fois le mot *b* apparaît ensemble avec *t*.

4.1.4 Approches basées sur la recherche dans un dictionnaire :

Les approches d'extraction de concepts basées sur un dictionnaire utilisent des ressources terminologiques pour comparer les instances textuelles à des entrées de concepts dans le dictionnaire, c'est-à-dire on effectue une comparaison exacte ou approximative des chaines de caractères entre les fragments du document et les entrées du dictionnaire. Un des premiers travaux d'extraction des concepts à partir de textes biomédicaux en exploitant des ressources terminologiques externes, est l'algorithme d'identification de concepts mis en œuvre dans le système SAPHIRE (Hersh et Greenes, 1990). Leur algorithme essaie de chercher tous les synonymes de chaque mot dans un texte et associe toutes les combinaisons possibles des mots synonymes afin de générer un terme complexe qui est par la suite comparé aux entrées d'un dictionnaire des termes qui désignent les concepts dans la ressource. Inspiré de cette idée, MetaMap est un programme développé à la NLM pour

associer aux textes biomédicaux les concepts du Méta-thésaurus UMLS (Aronson *et al.*, 2004b). Pour chaque terme complexe (groupe de mots consécutifs), les variantes sont générées en utilisant les connaissances dans la base lexicale SPECIALIST Lexicon de l'UMLS et une base de données supplémentaire de synonymes. Une variante d'un terme est composée de ses acronymes, ses abréviations, ses synonymes, les combinaisons significatives de ces derniers, les variantes flexionnelles et l'orthographe (Aronson, 1996). Les termes candidats dénotant des concepts UMLS sont récupérés lorsqu'elles contiennent au moins une des variantes générées.

Dans (Zhou et al., 2006) les auteurs ont proposé une approche d'extraction des concepts appelé MaxMatcher basée sur la recherche approximative des entrées dans un dictionnaire en tenant compte des variations lexicales et morphologiques des termes. MaxMatcher définit une mesure de similarité cité ci-après :

$$I(w_i, c) = max \{ I(w_i, t_j) \mid j \leq n \}$$

Étant donné un terme *t* qui est constitué par un ensemble de mots simples t ={w_1, w_2, …, w_m}. dans le texte, la similarité entre chaque mot constituent et le concept *c*, dénoté par un ensemble de termes c= {t_1, t_2,…,t_n}, a été définie comme suit :

Où :

-w_i est le mot constituent le terme t,

-t_j est le terme d'entrée j du concept c,

-n est le nombre de termes d'entrée du concept c,

-I (w_i,t_j) est la similarité entre le mot w_i et le terme t_j, calculée comme suit :

$$I(w_i, t_j) = \begin{cases} 0 & w \notin tj \\ \frac{1/N(wi)}{\sum_i 1/N(wij)} & w \in tj \end{cases}$$

N (w_i) est le nombre de concept dont le terme contient wi et wij désigne le i-éme mot dans le j-éme nom du concept.

La fonction permet de faire une recherche approximative afin de trouver les termes qui représentent potentiellement les concepts candidats.

Dans (DIHN, 2012) l'auteur a proposé une approche d'extraction de concepts biomédicaux qui exploite le thésaurus MeSH, pour identifier les meilleurs concepts dans les textes biomédicaux. Dans cette approche, chaque concept dans la ressource terminologique est désigné par un ensemble de termes d'entrée préférés ou non-préférés. Le concept est vu comme un document, la figure qui suit illustre un concept MeSH représentés comme un document différents qui est identifiés grâce à la balise "DOCNO" qui correspond à l'identifiant du concept dans le thésaurus MeSH. Le document est constitué des mots simples issus des termes d'entrée préférés ou non-préférés. Les balises "MH" et "ENTRY" correspondent aux termes préférés, appelé également *termes vedettes* ou *Main Headings* en anglais, et aux termes non-préférés (e.g., synonymes, abréviations, acronymes, ou variantes lexicales, etc.).

```
<DOC>
<DOCNO>C0039986</DOCNO>
<MH>Thoracic Surgery</MH>
<ENTRY>Surgery Thoracic</ENTRY>
<ENTRY>Surgery Heart</ENTRY>
<ENTRY>Cardiac Surgery</ENTRY>
<ENTRY>Heart Surgery</ENTRY>
<ENTRY>Surgery Cardiac</ENTRY>
</DOC>
```

Figure II.4 Exemples des concepts du thésaurus MeSH vus comme les documents d'une collection de concepts.

L'extraction de concepts terminologiques à partir d'un texte revient à sélectionner, parmi les concepts de la terminologie, ceux qui sont pertinents au contenu textuel. La pertinence est calculée selon une combinaison de scores thématiques (liée aux thèmes) et structurelle (score de corrélation d'ordre de mot constituant des termes) entre le texte et les concepts dans la terminologie. De ce fait, le poids du concept *C* vis-à-vis du texte Ø (document ou requête), dénoté *Rel*(*C, Ø*), peut être donné par :

$$Rel(C, Ø) = (1 + Sim(C, Ø)) \times (1 + p(C, Ø))$$

Où :

- *Sim*(*C, Ø*) correspond à la similarité thématique entre le concept *C* et le texte Ø,
- *ρ*(*C, Ø*) correspond à la similarité structurelle qui modélise la corrélation en termes d'ordre de mots entre le texte et chaque entrée d'un concept particulier dans la ressource terminologique. Pour ce faire, l'auteur à utiliser la mesure de Spearman en statistiques pour modéliser la corrélation des rangs liés à deux objets (e.g., deux textes).

Le score thématique *Sim*(*C, Ø*) a pour but de récupérer et d'ordonner dans un premier temps les premiers concepts qui sont les plus similaires à un texte Ø (e.g., le document ou la requête). Étant donné un texte, le système de RI, qui est essentiellement basé sur la mesure Cosinus, retourne une liste de concepts qui sont thématiquement similaire au texte original. Formellement, la similarité entre Ø et *C*, dénotée *Sim*(*C, Ø*), est donnée par :

$$Sim(C, \Theta) = \frac{\sum_{j=1}^{N_c} w_j * d_j}{\sqrt{\sum_{j=1}^{N_c} w_j^2} * \sqrt{\sum_{j=1}^{N_c} d_j^2}}$$

Où :

- N_c est le nombre total de concepts dans la terminologie,
- w_j et d_j sont le poids du mot *wj* dans le document représentant le concept *C* et dans le texte Ø. Ce poids, dénoté *pj*, est calculé en utilisant le schéma de pondération BM25 (Robertson et Walker, 1994)

La similarité structurelle entre le concept candidat et le texte donné est retenue comme la corrélation maximale entre chacun de ses termes d'entrée et le texte :

$$P(c, \emptyset) = \max_{E \in Entries\,(c)} p(E,\emptyset)$$

- *Entries*(*c*) désigne l'ensemble de termes d'entrée du concept *c*,
- $\rho(E, \emptyset) = max_{p \in \emptyset \wedge W \in p}\ \rho\,(W,E)$, où *p* est une phrase contenant la fenêtre *W*. Ici, la corrélation entre un terme d'entrée et le texte est la **corrélation maximale** entre le terme d'entrée et la fenêtre correspondante. Et la similarité ρ (*W,E*) est calculer comme suit :

$$\rho(W,E) = 1 - \frac{6 * \sum_i^L [r(w_i,W) - r(w_i,E)]^2}{L * (L^2 - 1)}$$

- r (w_i,*W*) représente l'ordre ou la position relative du mot-clé w_i dans la fenêtre du fragment de texte délimitée par le premier et le dernier mot qui constituent le terme d'entrée *E* du concept,
- r (w_i,*E*) est l'ordre du mot-clé w_i constituant l'entrée *E*,
- *L* est le nombre de mots dans les vecteurs représentant le fragment de texte et l'entrée du concept.

Si $\rho = -1$, la corrélation entre les deux rangs est nulle.

Si $\rho = 1$, les deux rangs sont parfaitement corrélés.

Dans les autres cas, la valeur de ρ est comprise entre -1 et 1 en fonction de la corrélation entre les deux rangs.

II.5 Conclusion

Dans ce chapitre nous avons vu un état de l'art sur les différentes approches d'extractions de concepts dans la littérature biomédicale.

Dans le chapitre qui suivra nous présentons notre approche d'extraction de concepts dans le domaine biomédical basée sur l'utilisation du thésaurus MeSH.

Chapitre III :

Proposition d'une approche d'extraction de concepts biomédicaux à partir de MeSH

III.1 Introduction

Dans ce présent chapitre nous présentons notre approche d'extraction de concepts biomédicaux à base du thésaurus MeSH, cette approche est basée sur l'approche d'indexation conceptuelle de (Boubekeur et al., 2011) que nous adaptons au domaine biomédical.

Ce chapitre est organisé comme suit :

- En section 1 nous donnons un aperçu général de notre approche,
- En section 2 nous détaillons notre proposition ainsi qu'un exemple illustratif détaillé de notre approche.

III.2 Description global de l'approche

Notre méthode de conception se base sur l'approche d'indexation conceptuelle de document (Boubekeur et al., 2011) mais au lieu d'utiliser WordNet comme ressource nous utilisons le thésaurus MeSH pour en extraire les concepts biomédicaux.

Le processus d'indexation s'effectue en 3 étapes :

2.1 Identification des concepts :

Le but de cette étape est d'identifier en se basant sur le thésaurus MeSH, l'ensemble des concepts représentatifs du document. A l'issue de cette étape nous obtenons une liste de termes désignant les concepts avec différentes localisation dans l'arborescence de MeSH, qu'il faudra par la suite désambigüiser.

2.2 Désambiguïsation des concepts :

Un concept est constituer d'un ou de plusieurs termes, un terme peut être identifié avec une seule localisation comme il peut avoir plusieurs localisations dans l'arborescence de MeSH, dans ce cas là il faudra sélectionner le terme le plus adéquat pour représenter le concept dans le document.

2.3 La pondération des concepts :

Dans cette étape nous utilisons la plateforme Terrier 3.5 afin de pondérer nos concepts obtenus à l'étape précédente avec le schéma classique Tf-IDf.

La figure qui suit montre une vue d'ensemble de notre approche d'extraction, les étapes seront détaillées en section suivante.

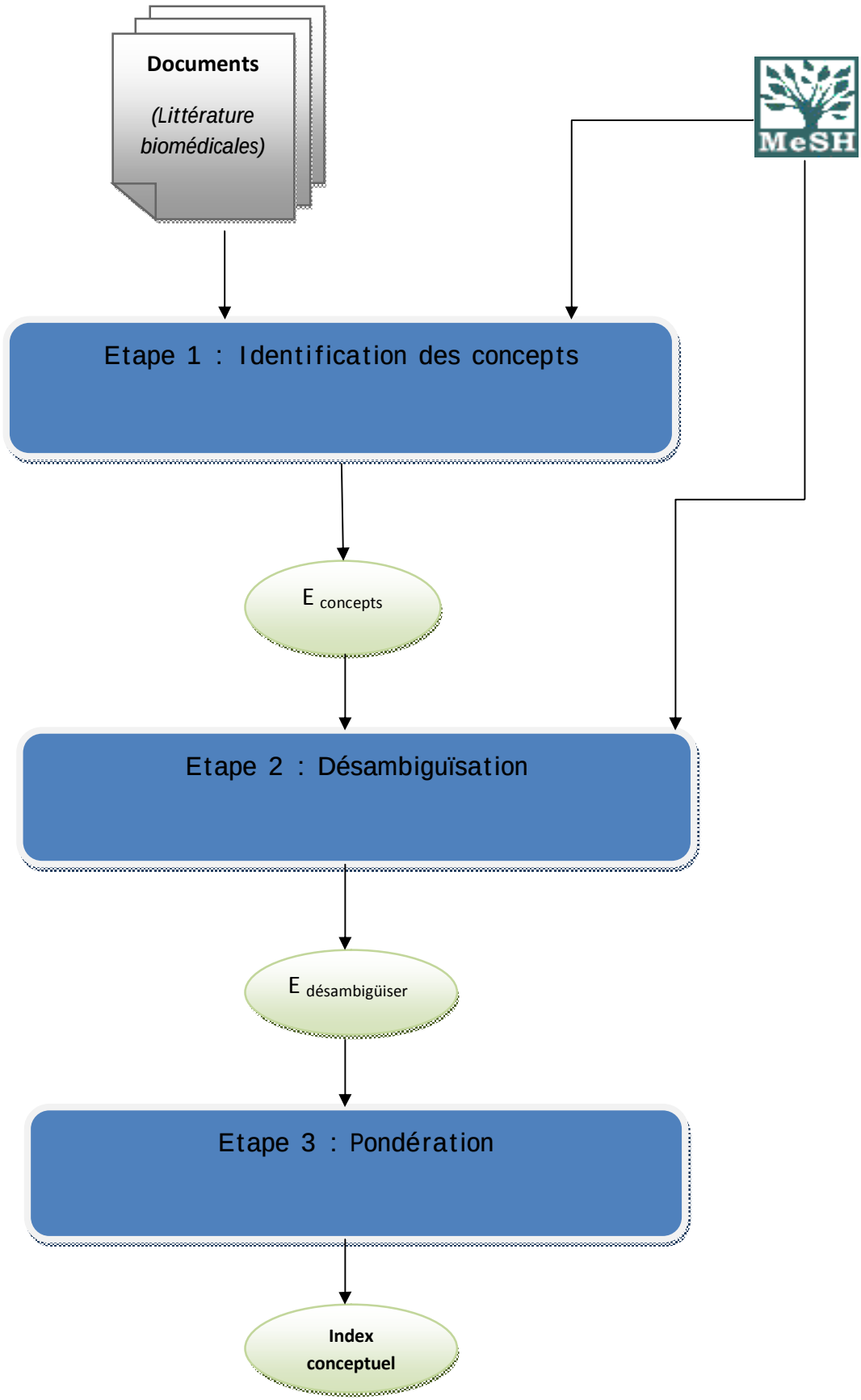


Figure III.1 : Vue d'ensemble de notre approche

III.3 Description détaillée de l'approche

3.1 Identification des concepts :

Le but de cette étape est d'identifier, en se basant sur MeSH, les concepts les plus représentatifs d'un document. Une étape préalable d'annotation syntaxique est d'abord prévue. L'objectif de l'annotation syntaxique (ou POS tagging) est d'identifier la partie du discours (Part Of Speech) de chaque entité syntaxique du document analysé. Une fois le document taggué, on extrait les termes. L'approche proposée est basée sur les trois phases suivantes :

- **1er Phase : Splitter le document en phrase**

Chaque document de la collection est découpé en phrases suivant les délimiteurs lexicaux (signes de la ponctuation, ; . ! ?). A la fin de cette phase, nous aurons identifié l'ensemble $E_{phrases}$ de toutes les phrases du document.

$E_{phrases} = \{p_1, p_2, \ldots, p_n\}$, où n est égale au nombre de phrases que le document contient.

- **2ème Phase : Analyser les phrases**

Pour chaque phrase p dans $E_{phrases}$, les mots qui composent p sont analysés. $P = \{w_1, w_2, \ldots, w_i\}$, *i* est égale au nombre de mots d'une phrase.

Pour chaque w_i de p, on cherche tous les concepts de MeSH qui commencent par w_i. L'ensemble de ces concepts constitue la liste $E_{candidats}$ des concepts candidats, par la suite cette liste sera ordonnée en taille décroissante de ses éléments.

Remarque :

- Si C_i est un concept préféré alors C_i est directement inséré dans $E_{candidats}$.
- Si C_i est un concept non préféré du concept C_i alors C_i est inséré dans $E_{candidats}$.

- **3ème Phase : identification des concepts**

Dans cette phase on procède à la projection de chaque élément de $E_{candidats}$ sur des expressions locales E_i de w_i. Une expression d'un mot w_i est la partie du texte qui commence par w_i. Sa taille est égale au nombre de mots qu'elle contient.

Si une expression locale s'apparie avec l'un des concepts candidats, elle est retenue et est insérée dans l'ensemble $E_{concepts}$. Sinon le concept suivant de la liste $E_{candidats}$ est examiné.

Soit C_n^i les concepts MeSH qui correspondent à w_i.

- Si C_n^i est un concept préféré alors C_n^i est directement inséré dans $E_{candidats}$.
- Si C_n^i est le concept non préféré du concept C_n^i alors c'est son concept C_n^i préféré qui est inséré dans $E_{candidats}$.

Algorithme de détection des concepts :

Entrée : document d

Sortie : $E_{concepts}$

Procédure : Soit w_i le prochain mot à analyser dans le document d.

Début :

1. Déterminer $E_{candidats} = \{C_1^i, C_2^i, \dots, C_n^i\}$ L'ensemble des concepts préféré MeSH commençant parle mot w_i en prenant en considération les synonymes de la balise ENTRY.
2. Ordonner $E_{candidats} = \{C_{(1)}^i, C_{(2)}^i, \dots, C_{(n)}^i\}$ où $(j)_{1\dots n}$ est une permutation d'indice telle que $|C_{(1)}^i| \geq |C_{(2)}^i| \geq \dots \geq |C_{(n)}^i|$, $|C_{(j)}^i|$ est la taille d'un terme d'entrée MeSH $C_{(j)}^i$.
3. Trouve ß faux ;
4. Pour chaque $C_{(j)}^i$ des $E_{candidats}$ et trouve = faux, faire :

   Calculer l'expression locale E_i de taille $|C_{(j)}^i|$:

   E_i ß $w_i, w_{i+1}, w_{i+2}, \dots, w_l$ telle que $|E_i| = |C_{(j)}^i|$ et $l = |C_{(j)}^i|$

   Si $E_i = C_{(j)}^i$ alors

   Trouve = vrai ;

   Fsi ;

   Fait ;

   Si trouve = vrai alors

   Mettre E_i dans l'ensemble $E_{concepts}$;

   Fsi ;

Fin.

3.2 Désambiguïsation des termes :

Dans notre travail on utilise la mesure de similarité de *Wu-Palmer* (Wu et al., 1994), afin de désambiguïser les termes obtenus dans l'ensemble $E_{concepts}$. Les mesures de similarité jouent un rôle important, en particulier dans le processus de désambiguïsation des termes. L'objectif des mesures de similarité sémantique est d'estimer la ressemblance entre les concepts (auxquels les termes des requêtes et documents sont rattachés). Un concept réfère à un sens particulier d'un terme donné. Pour cela la similarité est définie par rapport à la distance qui sépare deux concepts dans la hiérarchie de MeSH et également par leur position par rapport à la racine. La similarité entre c_1 et c_2 est :

$$Sim\left(C_1, C_2\right) = \frac{2 \times profondeur\,(C^*)}{profondeur\,C_1 + profondeur\,C_2 + 2 \times profondeur\,(C^*)}$$

Où :

$-\ C^*$ est le domaine le plus spécifique qui subsume $C_1\ et\ C_2$ dans la hiérarchie de MeSH.

$-\ profondeur\,(C^*)$ est le nombre d'arcs entre la racine de MeSH et le domaine *.

$-\ profondeur\,(C_{1/2})$ est le nombre d'arcs entre la racine de MeSH et le domaine $C_{1/2}$ en passant par le domaine C^*.

Soit $S_i = \{S_1^i, S_2^i, \dots, S_n^i\}$ l'ensemble de toutes les localisations obtenue pour un terme t_i.

Pour cela on associe à chaque $S_j^i \in S_i$ un score basé sur sa proximité sémantique avec les autres sens associés aux mots dans le document. Le terme S_j^i ayant le plus grand score est alors retenu comme concept adéquat pour le terme t_i.

Formellement :

$$\text{Score } (S_j^i) = \text{Arg max } \left(\sum_{l=1}^{l=m} \sum_{k=1}^{k=n} \quad Sim\left(C_{(j)}^{\ i}, C_{(k)}^{\ l}\right)\right)$$

Où m est le nombre de termes dans l'index d'un document, n représente le nombre des concepts associés à chaque terme t_l et $Sim\left(C_{(j)}^{\ i}, C_{(k)}^{\ l}\right)$ désigne la similarité entre les deux concepts $C_{(j)}^{\ i}$ et $C_{(k)}^{\ l}$.

Algorithme de désambiguïsation :

Entrée : $E_{concepts}$

Sortie : $E_{désambiguiser}$

Procédure : Soit C_i le prochain terme à désambiguïser.

Début :

1. Soit $S_l = \{S_1^i, S_2^i, \dots, S_n^i\}$ l'ensemble de toutes les localisations de t_i dans MeSH.
2. Score_max ß 0 ;
3. Pour chaque localisation $S_{(j)}^i \in S_l$ faire :
   - Score[j] ß 0 ;
   - Pour chaque terme $t_i \in CG_i$ faire :
     - Retrouver $S_l = \{S^1, S^2, \dots, S^p\}$ l'ensemble de tous les localisations du terme t_i dans MeSH ;
     - Pour chaque $S_{(l)}^k \in S_l$ faire :
       - Score[j] ß score[j]+ $Sim\left(S_l^{(j)}, S_l^{(k)}\right)$;
     - Fait ;
   - Fait ;
   - Si score[j] > score_max alors
     - Score_max ß score[j] ;
   - Fsi ;
4. Fait ;
5. Pour chaque score[j] /j=1..n faire :
   - Si score_max= score[j] alors
     - Prendre $S_{(i)}^j$ comme sens adéquat ;
   - Fsi ;
6. Fait ;

FIN.

A l'issue de cette étape tous les termes ambigus seront désambiguïsés, nous aurons ainsi identifié l'ensemble $E_{désambiguiser}$.

3.3 Pondération des concepts :

Une fois les concepts identifiés dans le document il s'agit de leur affecter un poids qui détermine leur importance dans le document. Nous utilisons pour notre cas le schéma de pondération classique tf * idf :

$$W(C_i) = Tf(C_i) \times IDf(C_i), \quad \text{où} \quad C_i \in E_{désambiguiser}$$

Où :

-Tf est le nombre d'occurrence du concept préféré C_i et ces synonymes apparaissent dans le document d,

-IDf est la fréquence documentaire inverse du concept C_i tel que :

$$IDf(C_i) = \log\left(\frac{nombre\ total\ de\ document\ dans\ la\ collection}{nombre\ de\ document\ contenant\ le\ concept}\right)$$

III.4 Illustration

Dans ce qui suit, nous illustrons par un exemple détaillé notre approche d'indexation conceptuelle basée sur l'extraction de concepts.

Etant donné le texte suivant (extrait du document 11712293, de la collection TREC Genomics 2004).

```
<DOC>
<DOCNO>11712293</DOCNO>
<TITLE> An appraisal of "supernormal" A-V conduction.</TITLE>
<ABSTRACT>
 - Certain temporal patterns of A-V and V-A transmission in
experimental preparations resemble phenomena attributed to
"supernormal" conduction in the clinic. Detailed study of the
properties of the A-V transmission system in such experiments
reveals alternative explanations in which supernormality is
clearly eliminated. By application of similar principles,
supernormality can be eliminated as a factor in most if not all of
the published examples. Three major categories can be discerned:
(1) occult 2:1 A-V block, in which an idioventricular beat
"retracts" an otherwise refractory barrier within the A-V node;
(2) alternation between dissociated intranodal transmission
pathways; and (3) "ventriculophasic" (vagal) depression of nodal
conductivity.
</ABSTRACT>
</DOC>
```

§ Etape 1 : Identification de concepts

- Phase 1 : Splitter le document en phrases

A partir du précédent document nous obtenons les neuf phrases suivantes :

1- *An appraisal of supernormal A-V conduction*

2- *- Certain temporal patterns of A-V and V-A transmission in experimental preparations resemble phenomena attributed to supernormal conduction in the clinic*

3- *Detailed study of the properties of the A-V transmission system in such experiments reveals alternative explanations in which supernormality is clearly eliminated*

4- *By application of similar principles*

5- *supernormality can be eliminated as a factor in most if not all of the published examples*

6- *Three major categories can be discerned: 1 occult 2:1 A-V block*

7- *in which an idioventricular beat retracts an otherwise refractory barrier within the A-V node*

8- *2 alternation between dissociated intranodal transmission pathways*

9- *and 3 ventriculophasic vagal depression of nodal conductivity*

- Phase 2 : Analyser les phrases et détection de concepts

N° phrase	Index	Correspondance	Concept préféré sélectionner
2	[*-, Certain, temporal, patterns, of, A-V, and, V-A, transmission, in, experimental, preparations, resemble, phenomena, attributed, to, supernormal, conduction, in, the, clinic*].	Preparations →prepar (terme non préféré)	Ritodine
4	[*By, application, of, similar, principles*]	Principles(lemme)→principlism(lemme du terme non préféré)	Principle-Based Ethics
5	[*supernormality, can, be, eliminated, as, a, factor, in, most, if, not, al,l of, the, published, examples*]	Published(lemme)→publishing(lemme d'un terme préféré)	publishing
7	[*in, which, an, idioventricular, beat, retracts, an, otherwise, refractory, barrier, within, the, A-V, node*]	A-V →A-V Node(debut du concept non préféré)	Atrioventicular Node
8	[*2, alternation, between, dissociated, intranodal, transmission, pathways*]	Dissociated(lemme)→dissociation(lemme d'un concept non préféré)	Dissociative Disorders
9	[*and, 3, ventriculophasic, vagal, depression, of, nodal, conductivity*]	depression→depression(concept préféré)	depression

Tableau: Etape d'identification des concepts

- **Phase 3 : Concepts identifiée**

$E_{concepts}$ = {(Ritodrine, D02.033.100.624.870; D02.033.755.624.870; D02.092.063.624.870; D02.092.471.683.870), (Principle-Based Ethics, K01.316.756; K01.752.256.830), (Publishing, L01.737), (Atrioventricular Node, A07.541.409.147), (Dissociative Disorders, F03.300), (Depression, F01.145.126.350)}.

Chaque concept qui est constitué d'un ensemble de termes est suivie par ça localisation dans l'arborescence MeSH.

§ Etape 2 : Désambiguïsation des concepts

On propose de désambiguïser le terme "principle-based ethics" lors de la recherche de ce terme dans l'arborescence MeSH on trouve deux localisations différentes:

principle-based ethics = (K01.316.756 ; K01.752.256.830)

On calcule le score de désambiguïsation associé à chacun des concepts de "principle-based ethics", on obtient les résultats suivant :

principle-based ethics|K01.316.756|1.53

principle-based ethics|K01.752.256.830|1.45

La première localisation K01.316.756 obtient le score le plus élevé, c'est donc le concept le plus adéquat du document.

III.5 Conclusion

Nous avons présenté dans ce présent chapitre la description détaillée de notre approche d'indexation de la littérature biomédicale basée sur l'utilisation du thésaurus MeSH. Cette approche permet l'extraction des concepts biomédicaux désambiguïsés.

Nous avons montré la faisabilité de notre approche en le déroulant sur un exemple détaillé.

Le chapitre suivant est consacré à son implémentation et à son évaluation.

CHAPITRE IV : Implémentation et tests

IV.1 Introduction :

Dans ce chapitre, nous allons d'abord présenter l'environnement technologique utilisé pour le développement de notre approche, donner quelques extraits des codes issus de son implémentation, puis présenter les résultats de son exécution.

L'approche est évaluée comparativement à l'approche de (DIHN, 2012) implémentée dans l'outil Cxtractor et comparativement à une approche classique basée mots-clés (telle que implémentée dans la plateforme Terrier 3.5).

IV.2 Environnement de développement :

Pour mener à bien notre travail, nous avons choisi comme environnement de développement Eclipse sous le système d'exploitation Windows 7.

- **Eclipse :** Eclipse est un environnement de développement (IDE) très puissant, extensible et intégré dont le but est une plate-forme modulaire pour permettre de réaliser des développements informatiques. Il a était développer par I.B.M. Dans notre application nous utilisant Eclipse GALILEO, Avec Eclipse, on travaille toujours au sein d'un projet, un projet est un ensemble de fichiers sources java. L'IDE Eclipse stocke l'information associée à un projet dans un dossier projet (projet folder), qui en plus de nos fichiers sources et les fichiers compilés, inclus toutes les ressources nécessaires à notre projet de développement, ainsi que deux fichiers de configurations (*.classpath* et *.projet*) utilisés par l'IDE. Tous nos projets sont regroupés dans un répertoire, *le workspace* Eclipse, qu'on doit choisir lors du lancement d'Eclipse.

Nous avons utilisé aussi pour le développement de notre application un étiqueteur syntaxique qui est :

- **Stanford POS tagger :** il s'agit d'un étiqueteur syntaxique développé par THE STANFORD NATURAL LANGUAGE PROCESSING GROUP. Il est utilisé pour identifier la partie du discours (part of speech) de chaque mot dans son contexte.

  Un exemple de code d'utilisation est présenté dans ce qui suit :

Exemple de code----------

```
//********** Initialisation du taggeur **********//
MaxentTagger tagger = new MaxentTagger("D:\\stanford-postagger\\models\\left3words-wsj-0-18.tagger");
//********** Tagguer une phrase **********//

.
.
.
if(ligne.startsWith("<TITLE>") || ligne.startsWith("<ABSTRACT>"))
text=sw.append(ligne).toString();
String tagged = tagger.tagString(text.trim());
st= new StringTokenizer(tagged.trim());
.
.
.
```

IV.3 Implémentation :

3.1 Programme pour splitter la collection en documents :

Un exemple du programme pour splitter la collection en document est donné dans ce qui suit :

Exemple de code----------

```
public class Spliter_Collection {
static File collection =new File( "C:\\trec04.txt");
//……

in=new BufferedReader(new FileReader(collection));
while ((ligne=in.readLine()) !=null)
{
      if(ligne.startsWith("<DOCNO>"))
      {
        //garder le DOCNO comme nom du document
          String id_doc=st.nextToken().trim();
//……
```

```
            while        ((ligne=in.readLine())        !=null        &&
            ligne.trim().startsWith("</ABSTRACT>")==false              &&
            ligne.trim().startsWith("</TITLE>")==false )
            {
                 text=sw.append(ligne).toString();
      }

}}
```

3.2 Programme d'identification des concepts :

Pour chaque document (requête) on commence par identifier les différents concepts extrait du thésaurus MESH. Un extrait de notre programme d'identification des concepts est donné dans ce qui suit :

Exemple de code----------

```
static File collection =new File( "C:\\collection\\trec04");
static File Mesh =new File( "C:\\mesh_de_dihn.txt");
static PrintWriter sortie;

public  static void main(String args[]) throws IOException

{ File[] document = collection.listFiles();
  for(int i=0; i<document.length; i++)
  {
      //parcours les phrases de notre texte et tokenizer en mots et les
      //ajouter dans la liste termes

      for(int j=0; j<phrases.length; j++)
      {
            ArrayList<String>termes=new ArrayList<String>();
            ArrayList<String>concepts=new ArrayList<String>();
            ArrayList<String>Keys_concepts=new
            //…
            …
           …
          in=new BufferedReader(new FileReader(phrases[j]));
          while ((ligne=in.readLine()) !=null){
           StringTokenizer st=new StringTokenizer(ligne);
            while(st.hasMoreTokens())
              termes.add(st.nextToken().trim());
              }
      }
```

```
//pour chaque mot dans termes si le mot commence par le terme entre MH
//alors le garder dans concept_pref
for(int k=0; k<termes.size();k++ ){
   while ((ligne=in.readLine()) !=null) {

      if(ligne.startWith("<MH>"))
      {StringTokenizer st=new StringTokenizer(ligne, ">");
            concept_pref=st.nextToken();

      collocs.add(concept_pref);
      preferes.add(concept_pref);
      keys.add(key);
      else
// si c'est le lemme de MH qui commence par le terme alors
         if(lemma.startsWith(termes.get(k)) {

          collocs.add(lemma.toString());
          preferes.add(concept_pref);
          keys.add(key);

      }
}
//si le terme commence par le terme qui ce trouve entre la balise ENTRY
//alors

     if(ligne.stratWith("<ENTRY>")){
      StringTokenizer st=new StringTokenizer(ligne, ">");
      colloc=st.nextToken();//…

      if(colloc.startsWith(termes.get(k))) {
      collocs.add(colloc);
      preferes.add(concept_pref);
      keys.add(key);
}else{
//……
//si c'est le lemme de ce qui ce trouve entre la balise de ENTRY sui est
egale au termes

      if(lemma2.startsWith(termes.get(k))) {

      collocs.add(lemma2.toString());
      preferes.add(concept_pref);
      keys.add(key);
//……
      }
   }
}
```

3.3 Programme de détection des localisations des concepts dans MESH :

Après l'identification des concepts pour chaque document, nous lui attribuant sa localisation dans l'arborescence MESH.

Exemple de code----------

```
static File desambiguisation =new File("C:\\Sortie");
//répertoire des résultats de détection de concepts
static File Mesh =new File("C:\\mesh.txt"); //MeSH avec localisation
public  static void main(String args[]) throws IOException
{ //récupérer les concepts identifié est les comparer au entrée MeSH
      while ((ligne11=in11.readLine()) !=null) {
      StringTokenizer st=new StringTokenizer(ligne11,";");
      String s=st.nextToken().trim();
//si le concept égale au concept MeSH récupéré sa localisation
      if(s.toLowerCase().equals(concepts.get(k).toLowerCase().trim()))
           {          num_concept.add(keys.get(k));
                      debut_concept.add(s.trim().toLowerCase());
                      localisation.add(st.nextToken().trim());
}
```

3.4 Programme de désambiguïsation des termes :

Un extrait du programme de désambiguïsation des termes qui utilise la similarité de Wu-Palmer est donnée dans ce qui suit

Exemple de code----------

```
//récupérer chaque deux termes avec localisations différentes
For (int j=0; j<keys_syns.size(); j++)
{
   for(int k=0; k<keys.size(); k++){

      if(syns.get(j).equals(num_concepts.get(k))==false) {
// tokenizer les localisations par rapport au point
         StringTokenizer str1=new StringTokenizer (tf2.get(j).trim(),".");
         StringTokenizer str2=new StringTokenizer (keys.get(k).trim(),".");
```

```
        if(str1.countTokens()>=str2.countTokens()){
        // si la 1er localisation est plus longue ou égale à la 2eme
                String key1= str1.nextToken().trim();
                String key2=str2.nextToken().trim();
                if(key1.trim().equals(key2.trim())){

                  n3=n3+1; // profondeur du subsumant

                 }
             n1++;
             n2++;
            }
          }
         else
           while(str1.hasMoreTokens()){
                n1++;// profondeur du concept 1

                }
           while(str2.hasMoreTokens()){

                n2++;// profondeur du concept 2
                  }
       //calcul de la similarité Wu-Palmer
              double sim=(2*n3)/(n1+n2+2*n3);
            ……

              som=som+sim;System.out.println("somme="+som);
          }
          }
----------------------------------------------------------------------
```

IV.4 Evaluation expérimentale :

L'objectif de l'évaluation expérimentale est d'étudier l'impact de notre approche d'indexation conceptuelle biomédical sur les performances de la recherche d'information. En pratique, il s'agit de mesurer la qualité des réponses de notre système à des requêtes utilisateur. Cette mesure se faisant par comparaison de nos résultats par rapport aux résultats obtenus par Cxtractor sur une collection de test et par une indexation classique.

4.1 Collection de test utilisé :

Pour mener ces expérimentations et afin de pouvoir comparée nos résultats à ceux de Cxtractor nous avons opté pour la même collection (sous collection de 1000 documents) que ce dernier utilise en l'occurrence la collection TREC Genomics 2004.

- **Description de la collection de documents**

La collection de TREC Genomics est un ensemble d'environ 4.6 millions de citations, extrait à partir de la base MEDLINE. Chaque document de la collection se compose principalement d'un titre et/ou un résumé qui constituent l'essentiel du contenu textuel du document.

- **Exemple de requêtes :**

Les requêtes sont collectées à partir des vrais besoins d'informations des biologistes. Un ensemble de 50 requêtes ont été créées et jugés par les utilisateurs. Chaque requête contient trois champs principaux :

- **ID** : identifiant de la requête,
- **TITLE** : besoin d'information bref ou requête courte,
- **NEED** : besoin d'information détaillé ou requête longue,
- **CONTEXT** : information supplémentaire sur la requête.

Dans nos expérimentations, nous avons considéré que les balises ID, TITLE et NEED. Un exemple de requête est donné dans ce qui suit :

```
<TOPIC>
<ID>1</ID>
<TITLE>Ferroportin-1 in humans</TITLE>
<NEED>Find articles about Ferroportin-1, an iron transporter, in
humans.</NEED>
<CONTEXT>Ferroportin1 (also known as SLC40A1; Ferroportin 1; FPN1;
HFE4; IREG1; Iron regulated gene 1; Iron-regulated transporter 1;
MTP1; SLC11A3; and Solute carrier family 11 (proton-coupled divalent
metal ion transporters), member 3) may play a role in iron
transport.</CONTEXT>
</TOPIC>
```

4.2 Protocole d'évaluation :

Dans ce cadre d'évaluation, nous avons utilisé les mesures d'évaluation suivantes :

– **la précision à** *X* premiers documents (dénotée *P@X*), est donc la proportion de documents pertinents par rapport aux *X* premiers documents renvoyés par le SRI. La précision à *X* mesure la satisfaction de l'utilisateur concernant les *X* premiers documents pertinents.

– **la MAP** (Mean Average Precision) correspond à la précision moyenne calculée sur l'ensemble des documents pertinents retournés. La MAP mesure la capacité du modèle d'appariement ou d'un SRI à pouvoir sélectionner les documents pertinents, en réponse à un ensemble de requêtes.

Afin d'évaluer notre approche, nous l'avons implémentée et intégrée dans la plateforme Terrier 3.5. L'évaluation est faite sur une sous collection TREC de 1000 documents, pour chaque requête les 1000 premiers documents restitués par le système sont examinés et les précisions P@x à différents points x (x= 1, 2, 3, 4, 5, 10, 15, 20, 30, 50, 100, 200, 500, 1000) ainsi que la précision moyenne MAP sont calculées.

Il s'agit ensuite de comparer principalement les résultats obtenus à partir de notre approche à ceux restitués par l'outil Cxtractor de (DIHN, 2012) et par une approche d'indexation classique.

4.3 Evaluation de l'approche d'extraction

Les résultats de l'évaluation de notre approche ainsi que ceux de Cxtractor et de Terrier classique sont présentés dans le tableau suivant :

Information	Indexation Classique Terrier 3.5 (TF_IDF)	Indexation conceptuelle	Cxtractor
Number of queries	50	46	48
Retrieved	26214	13584	13025
Relevant	1159	1084	1127
Relevant retrieved	974	766	802
Precision at 1 : Precision at 2 : Precision at 3 : Precision at 4 : Precision at 5 : Precision at 10 : Precision at 15 : Precision at 20 : Precision at 30: Precision at 50 : Precision at 100 : Precision at 200 : Precision at 500 : Precision at 1000 :	0.8800 0.8900 0.8933 0.8500 0.8280 0.7120 0.6120 0.5350 0.4173 0.2836 0.1612 0.0887 0.0386 0.0195	0.6957 0.6630 0.6377 0.5978 0.5913 0.4935 0.4290 0.3685 0.2870 0.2052 0.1243 0.0730 0.0331 0.0167	0.6875 0.6875 0.6528 0.6402 0.5958 0.4833 0.4236 0.3635 0.2910 0.2096 0.1279 0.0734 0.0334 0.0167
MAP:	0.5097	0.3150	0.3226

Figure IV.1 : Contenus des fichiers d'évaluations.

Le tableau suivant est exprimé sous forme de deux graphes dans ce qui suit :

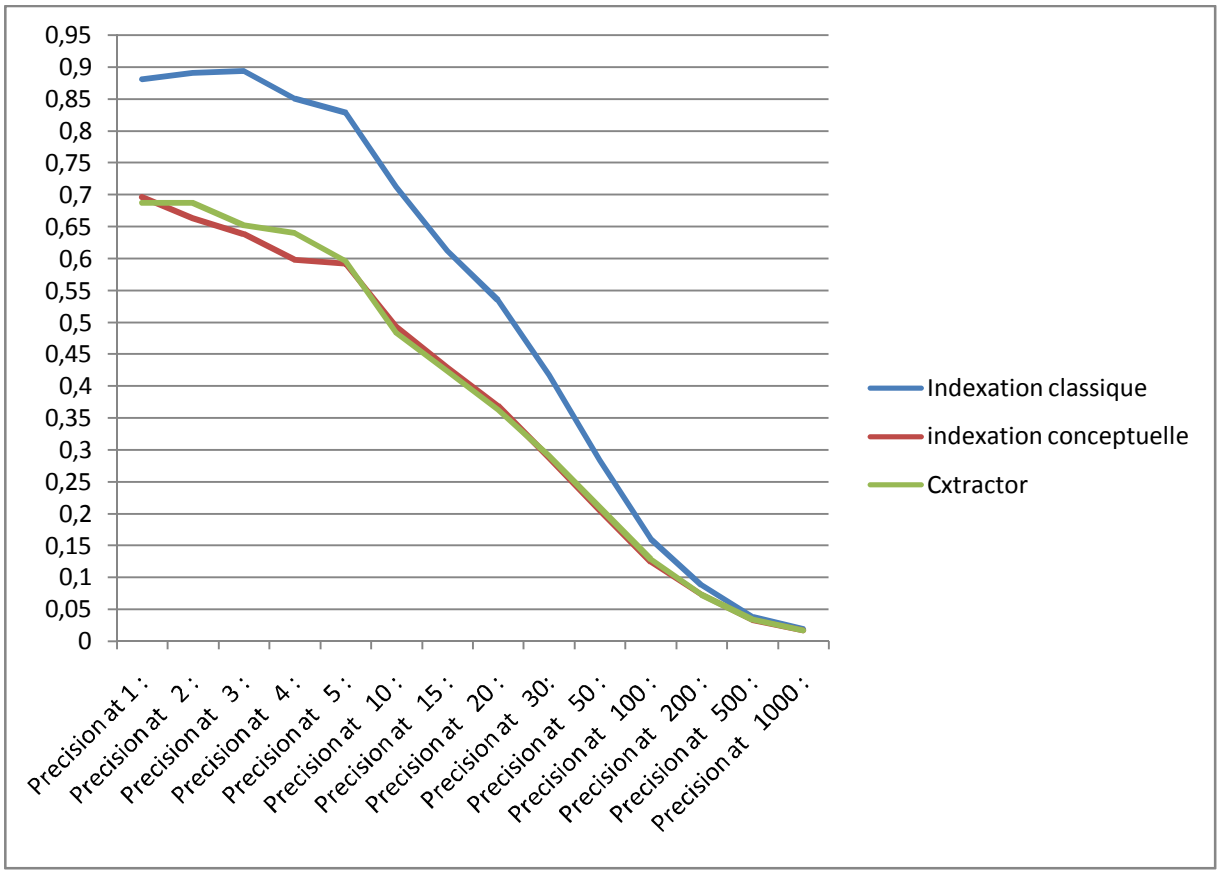


Figure IV.2 : Precision at x

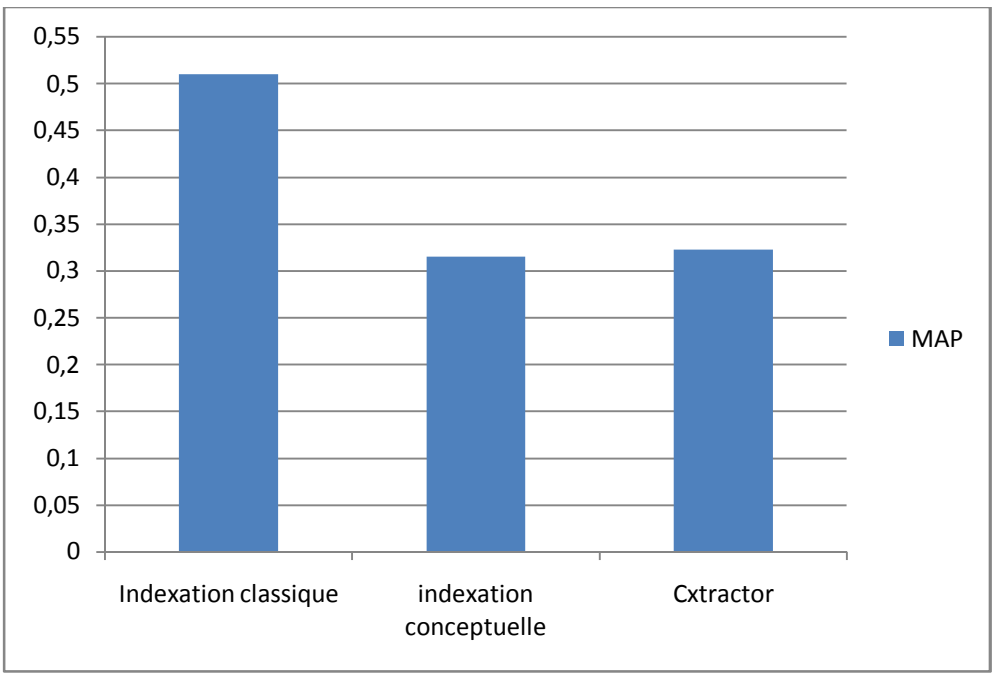


Figure IV.3 : MAP

Nous constatons que :

- L'indexation classique offre de meilleurs résultats que notre approche ainsi que celle implémenter dans Cxtractor.
- Notre approche et quasiment aussi performante que Cxtractor à partir des points de précisions 1 et 5. En deçà elle est légèrement moins performante.

L'indexation classique est meilleure car elle prend en considération tous les mots alors que notre indexation enlève les mots qui ne font pas partie du domaine biomédical.

Pour cela il aurait fallut qu'on intègre les mots simple qui ne font pas partie du domaine de la médecine.

Par ailleurs nos différences de résultat avec Cxtractor sont dûs à des erreurs de lemmatisation.

IV.5 Conclusion

Nous avons décrit au cours de ce chapitre l'implémentation et l'intégration de notre approche d'extraction de concepts à la plate forme Terrier 3.5. L'évaluation nous a permis de comparer notre approche à des approches existantes.

Les résultats ont montré que notre approche, bien que loin des performances de l'indexation classique, reste néanmoins concurrentielle par rapport à l'approche Cxtractor.

Conclusion générale

Le travail dans ce mémoire à présenté une approche d'indexation conceptuelle plus précisément à l'extraction de concepts basée sur l'utilisation d'une terminologie biomédicale qui est le thésaurus MeSH. Les concepts ainsi identifiés sont utilisés pour représenter les documents qui sont impliqués dans la phase de recherche afin d'ajuster la pertinence des documents par rapport à la requête.

Notre approche a été évaluée sur un petit corpus et comparée à une approche existante. Les résultats obtenus sont encourageants.

Dans nos travaux futurs, nous envisageons d'améliorer dans un premier temps cette approche en lui rajoutant les mots simples. Ensuite, utiliser d'autres ressources terminologiques. Et enfin, combiner plusieurs méthodes d'extraction de concepts, pour améliorer la précision de l'extraction de concepts afin de produire une meilleure représentation du document et de la requête.

Annexe I : Cxtractor

Cette annexe est consacrée à la présentation de la plateforme Cxtractor 1.0.3. Une plateforme pratique et efficace pour l'extraction de concepts dans les domaines biomédicaux.

§ **Présentation de Cxtractor :**

Cxtractor est un logiciel open source développé par (Duy, Dimh 2012) en java visant à intégrer les algorithmes d'extractions de concepts de l'auteur dans une plateforme générique, Cxtractor utilise Lesh afin d'extraire des concepts à partir des textes biomédicaux.

§ Structure de Cxtractor :

Cxtractor contient un ensemble de répertoire, comme le montre la figure ci-dessous

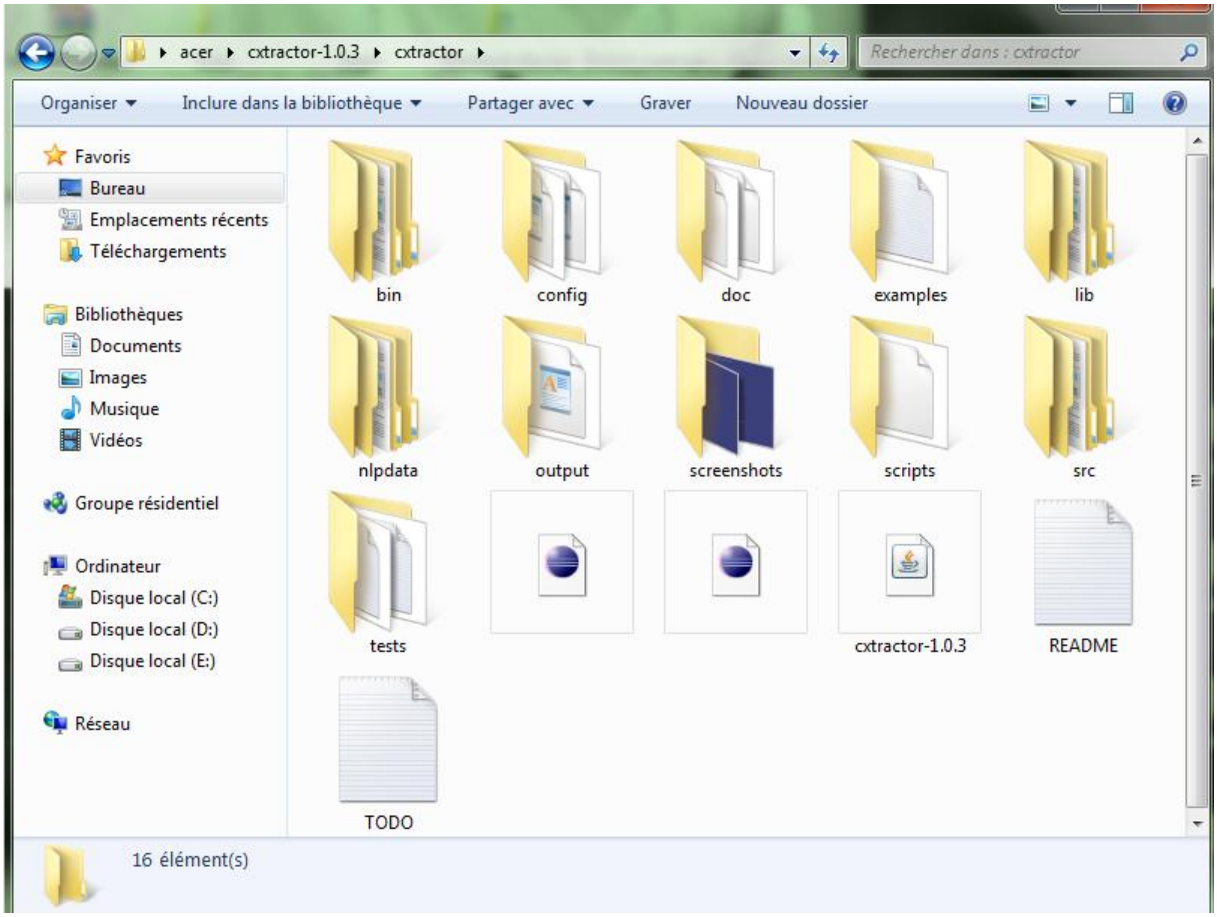


Figure : Contenue de Cxtracor

- bin\ : contient l'ensemble des scripts nécessaires pour exécuter Cxtractor.
- config\ : contient les fichiers de configuration de Cxtractor (le fichier settings properties sample contient la plupart des propriétés de configuration de cxtractor)
- doc\ : contient la documentation relative à cxtractor.
- examples\ : contient des exemples de document donnés en entrée pour extraction de concept.
- lib\ : contient les différentes librairies externes utilisées par cxtrator.

- nlpdata\ : contient les différentes ressources qu'utilise cxtractor pour extraire les concepts (Mesh, UMLS, Werdnet, snamed)
- output\ : contient les résultats lors de l'exécution d'extractions de concepts pour des documents en utilisant une des différentes ressources.
- screenstrats\ : contient quelques captures d'écran.
- scripts\ : contient un fichier en script shell.
- src\ : contient le code source java de cxtractor.
- tests\ : contient les collections qu'on donne pour l'extraction de concepts.

La liste complète des options pour lancer cxtractor[1] sur une ligne de commande est donnée comme suit :

Usage: java -jar cxtractor.jar [-r|--recursive] [-c|--clean] [-f|--file]
[-d|--folder] input [-e|--doctype documentType] [-o|--output output]
[-t|--terminology terminology] [-X|--cxMethod method]
[-w|--wModel weightingModel] [-v|--version]
Exemple d'usage:
cxtractor.jar -r -c -d tests -o output -X TerrierSpearmanExtractor
-r --recursive : Recursively processing
-c --clean : Clean all previous data
-h --help : Print this usage information
-f --file : Extracting concepts from a file
-t --terminology : Terminology used
-w --wModel : Weighting model (PL2 by default)
-X --cxMethod : Extraction method (MaxMatcherExtractor by default)
-d --folder : Extracting concepts from a directory
-e –doctype : Document type (file, trec, html)

[1] http://www.softpedia.com/progDownload/Cxtractor-Download-230918.html

-o --output : Output directory

-v --version : Version number

Les documents d'entrée peuvent avoir un des formats suivants : .txt, .html, ou les formats de TREC. (à configurer dans le fichier de configuration /config/settings.properties.sample). Lors de l'exécution, les paramètres de configuration sont chargés automatiquement.

Nous donnons ci-dessous un exemple de documents sous le format de TREC.

Chaque document TREC contient des balises particulières, par exemple :

– **DOC** représente le document,

– **DOCNO** représente l'identifiant unique du document,

– **TITLE** correspond au titre du document,

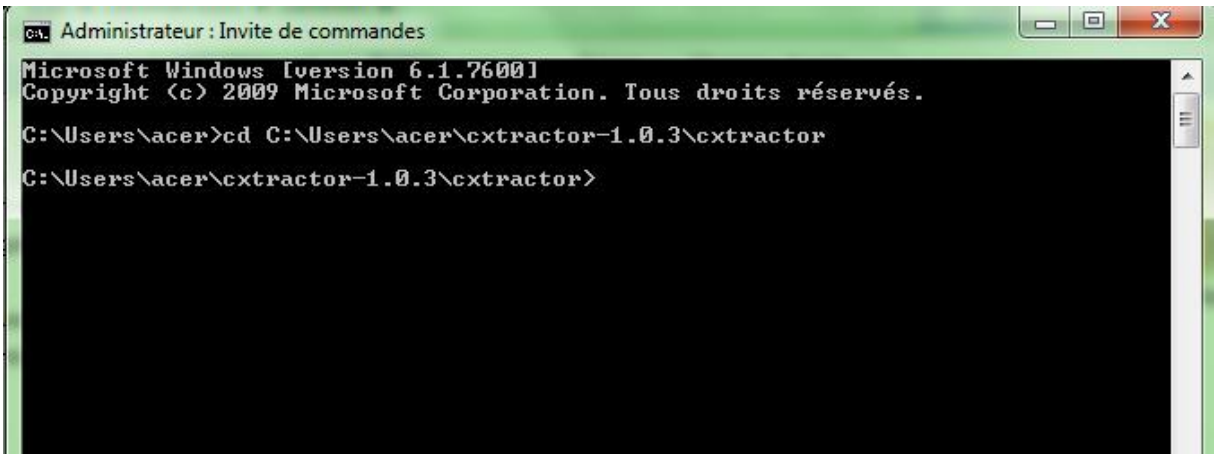
– **ABSTRACT** correspond au résumé du document.

<DOC>
<DOCNO>11096424</DOCNO>
<TITLE>- Prenatal radiation-induced limb defects mediated by Trp53-
dependent apoptosis in mice.</TITLE>
<ABSTRACT>- We reported previously that in utero radiation-induced apoptosis in the predigital regions of embryonic limb buds was responsible for digital defects in mice. To investigate the possible involvement of the Trp53 gene, the present study was conducted using embryonic C57BL/6J mice with different Trp53 status. Susceptibility to radiation-induced apoptosis in the predigital regions and digital defects depended on both Trp53 status and the radiation dose ; i.e., Trp53 wild-type (Trp53(+/+)) mice appeared to be the most sensitive, Trp53 heterozygous (Trp53(+/-)) mice were intermediate, and Trp53 knockout (Trp53(-/-)) mice were the most resistant. These results indicate that induction of apoptosis and digital defects by prenatal irradiation in the later period of organogenesis are mediated by the Trp53 gene. These findings suggest that the wild-type Trp53 gene may be an intrinsic genetic susceptibility factor that is responsible for certain congenital defects induced by prenatal irradiation.
</ABSTRACT>
</DOC>

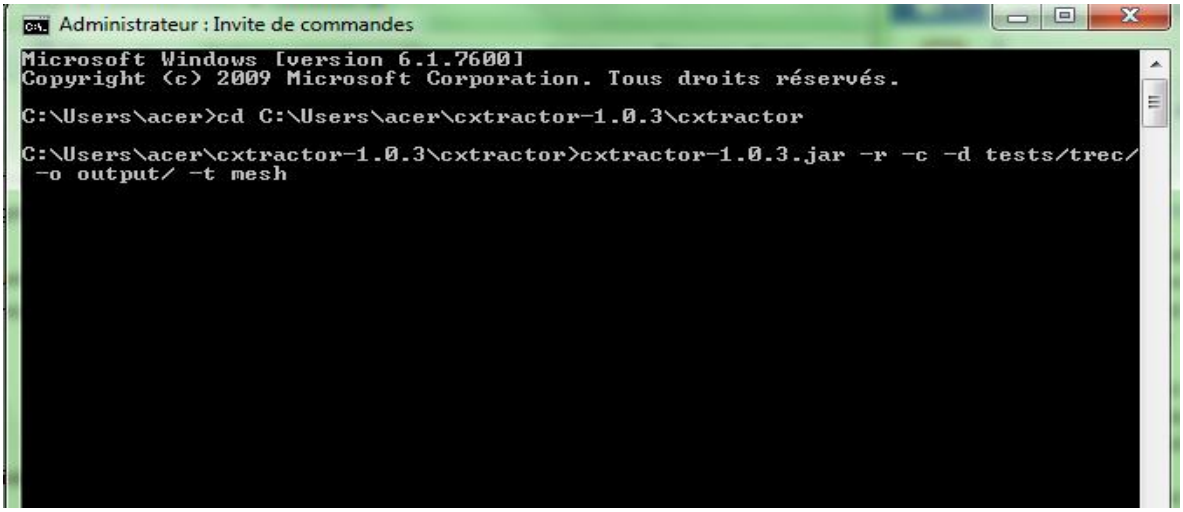
Afin de faciliter l'analyse et le traitement des concepts extraits, nous utilisons le format suivant pour sauvegarder les concepts candidats identifiés :

<DOCNO> DOC1
rank|CUI|concept name (preferred/non-preferred terms)|score
rank|CUI|concept name (preferred/non-preferred terms)|score
....
rank|CUI|concept name (preferred/non-preferred terms)|score
<DOCNO> DOC2
rank|CUI|concept name (preferred/non-preferred terms)|score
rank|CUI|concept name (preferred/non-preferred terms)|score
....

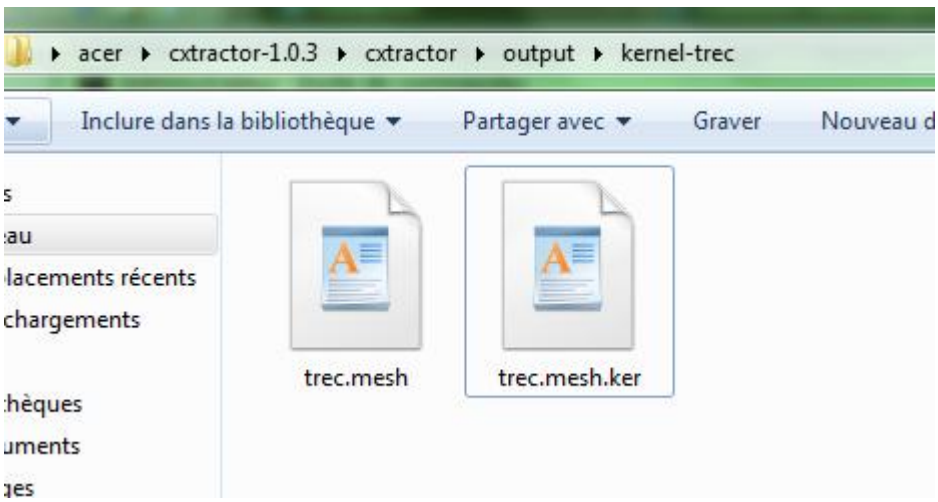
- **Exemple d'exécution :**



Rentrer dans le document cxtractor par l'invite de commende , lancé la commende *cxtractor.jar -r -c -d tests/trec/ -o output/ -t mesh* qui prendra les documents qui ce trouve dans le répertoire trec et utilisera le thésaurus MeSH pour l'extraction de concepts.



Par la suite on retrouve les résultats de l'extraction de concepts dans le répertoire output comme suit :



Exemple

```
<DOC>
<DOCNO>10928726</DOCNO>
<TITLE>A randomized controlled trial of Moderation-Oriented Cue
Exposure. </TITLE>
<ABSTRACT>OBJECTIVE: A randomized controlled trial was conducted to
examine the effectiveness of Moderation-Oriented Cue Exposure (MOCE)
in comparison to Behavioral Self-Control Training (BSCT). The main
hypothesis was that MOCE would be more effective than BSCT among a
sample of problem drinkers aiming at moderate drinking. A subsidiary
hypothesis was that MOCE would be relatively more effective than
BSCT among problem drinkers with higher levels of alcohol
dependence. METHOD: Clients (N = 91; 75% men) were randomly
allocated to either MOCE or BSCT. Treatment was delivered in weekly
sessions by two trained therapists, in a nested design in which
therapists switched to the alternative treatment modality
approximately halfway through the trial. Follow-up was carried out 6
months following posttreatment assessment, with 85% successful
contact. RESULTS: There was no evidence for the general superiority
of MOCE over BSCT. The subsidiary hypothesis was not confirmed. A
subsample of clients (n = 14) showing levels of dependence at
baseline above the commonly accepted cut-point for a moderation goal
(Severity of Alcohol Dependence Questionnaire [SADQ] 29) showed
outcomes at least as favorable as those below the cut-point. The
validity of self-reports of alcohol consumption and problems was
supported by significant relationships with liver function tests
(gamma-glutamyl transferase and alanine transferase). CONCLUSIONS:
These results provide no grounds for the replacement of BSCT by MOCE
in routine, moderation-oriented treatment practice. Assuming they
prefer it to abstinence and that it is not contra-indicated on other
grounds, there seems no reason why clients showing a higher level of
dependence (SADQ = 30-45) should not be offered a moderation goal.
</ABSTRACT>
</DOC>
```

Les résultats d'extraction de concepts pour ce document sont donnés comme suit :

```
<DOCNO>    10928726
0|C0001898|Alanine|24,9351|D12.125.042
1|C1096777|Randomized Controlled Trial|20,3211|V03.200.700
2|C0023901|Liver Function Tests|19,9158|E01.370.372.460
3|C0034394|Questionnaires|19,6674|E05.318.308.750
4|C0001948|Alcohol Drinking|16,9139|F01.145.317.269
5|C0025663|Methods|16,4256|E05.581
6|C0011546|Dependency (Psychology)|16,3984|F01.752.330
7|C0376287|Behavior Control|16,2367|E02.085
8|C0684271|Drinking|14,4327|G07.610.593.260.249
9|C0010439|Cues|13,3298|F02.463.425.234
10|C0018017|Goals|11,7591|F01.658.500
11|C0030705|Patients|10,4221|M01.643
12|C0025266|Men|9,5162|M01.390
13|C0039796|Therapeutics|8,0145|E02
14|C0040676|Transferases|7,7171|D08.811.913
15|C0001975|Alcohols|6,3566|C21.739.100.250
```

Annexe II : La plateforme de RI Terrier 3.5

Cette annexe est consacrée à la présentation de la plateforme de RI Terrier (version 3.5), une plateforme de haute performance et évolutive qui permet le développement à grande échelle de nouvelles applications de recherche d'information.

1. Présentation de Terrier :

Terrier, *TerabyteRetrIEveR* est moteur de recherche robuste et efficace, développé par le département informatique de l'université Glasgrow en Ecosse. Il est open source et entièrement écrit en java. Il est utilisé avec succès pour la recherche web, ad hoc et multilingue.

Comme tous moteurs de recherche, terrier permet :

- **L'indexation classique :** permet l'extraction des mots clés des documents appartenant à une collection et les stocker dans un index.
- **La recherche :** permet de retrouver des documents pertinents pour répondre aux requêtes formulées par l'utilisateur.
- **L'évaluation :** permet d'évaluer les résultats de la recherche.

2. Installation de Terrier :

Pour pouvoir utiliser Terrier 3.5, il est nécessaire de suivre les étapes suivantes :

- Installer une JRE (version 1.6.0). On peut télécharger la JRE ou la JDK sur le site de Java.
- Installer la version Java 1.6 ou supérieure.
- S'assurer que le chemin vers le dossier bin du java installé est dans la variable d'environnement PATH (vérifier l'exécution du java.exe et le compilateur javac.exe).
- Télécharger la copie de la version 3.5 de Terrier à partir du site de la plateforme de Terrier.
- Décompresser le contenu du fichier terrier-3.5.zip téléchargé

3. Structure de Terrier :

Terrier contient un ensemble de répertoire, ils sont structurés comme suit (Voir figure 1) :

- bin\ : contient l'ensemble des scripts nécessaires pour exécuter Terrier.
- doc\ : contient la documentation relative à Terrier.
- etc\ : contient les fichiers de configuration de Terrier (le fichier terrier.properties.sample contient la plupart des propriétés de configuration de Terrier).
- lib\ : contient les classes compilées de Terrier et les différentes librairies externes utilisées par Terrier.
- licenses\ : contient les informations sur la licence des différents composants de Terrier.
- share\ : contient la liste des mots vides (stopword-list.txt) et des exemples de documents à tester sur Terrier.
- scr\ : contient le code source java de Terrier.
- var\ : contient deux dossiers. Le premier est créé après une indexation, le second est créé après une recherche.

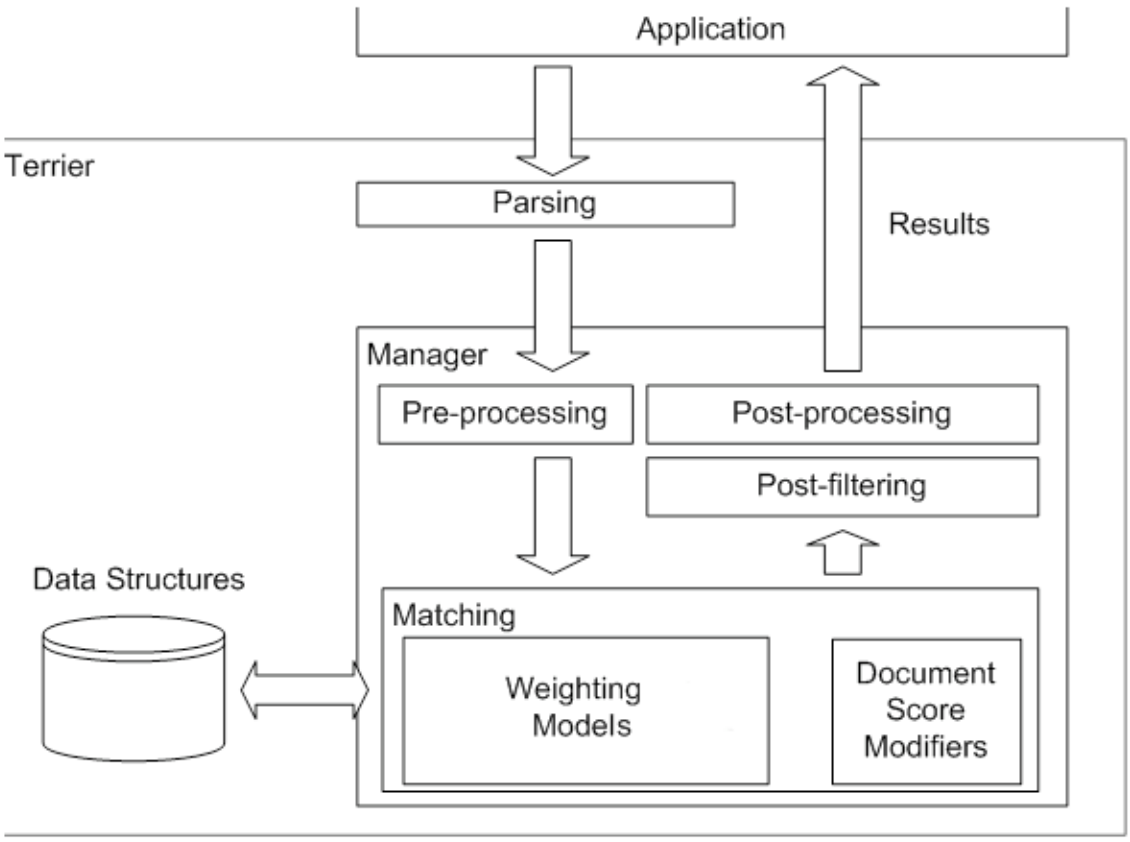
- index\ : contient les structures de données après indexation (fichier inverse, fichier lexicon, index direct, index documents).
- result\ : contient les résultats de la recherche et l'évaluation.

4. Composants de Terrier :

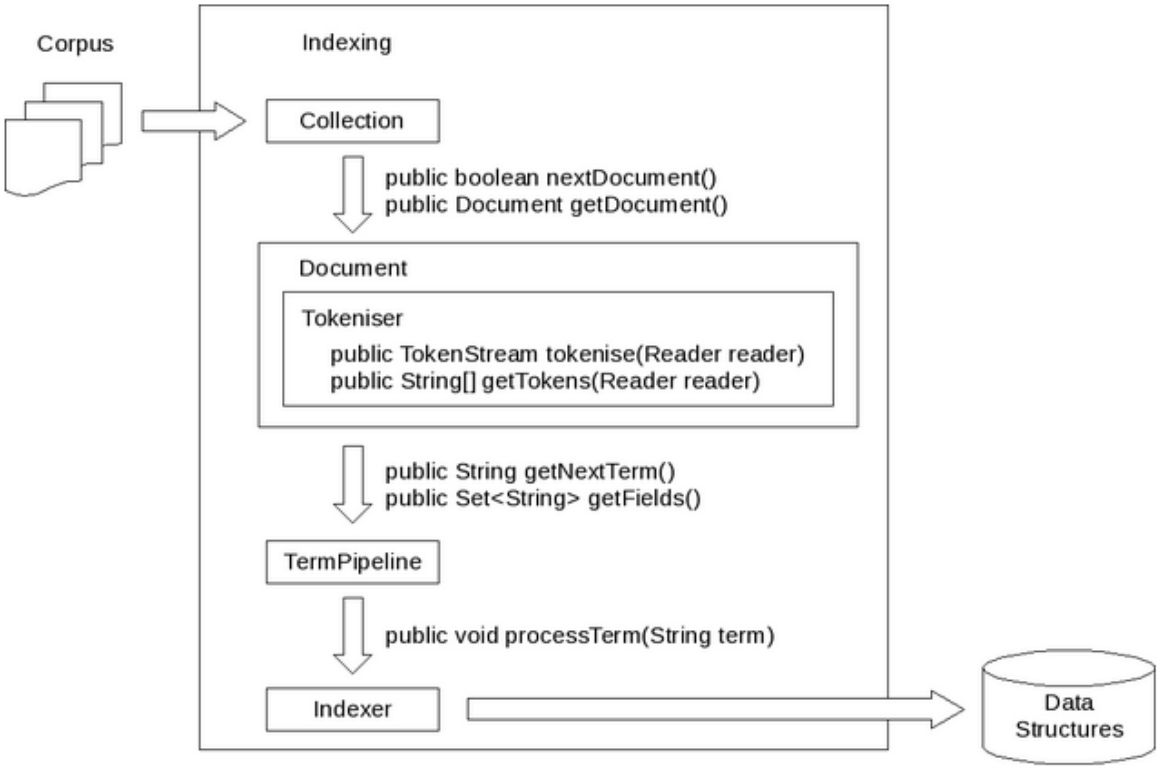
§ L'API d'indexation :

Le processus d'indexation de Terrier est divisé en quatre procédures :

1. Parcourir l'ensemble du corpus et envoyer chaque document à l'étape suivante.
2. Parser chaque document reçu et extraire les différents termes.
3. traiter tous les termes extraits via le composant TermPipeline.
4. Construction de l'index.



La figure 3 représente un aperçu de l'interaction des différents composants impliqués dans le processus d'indexation de Terrier.



4.1.1. Collection :

C'est une interface qui se trouve dans le package org.terrier.indexing (…\src\core\terrier\indexing). Ce composant permet de Splitter une collection (corpus) en documents. Il fait appel pour cela à plusieurs méthodes :

- public Document getDocument() ;
- public booleannextDocument() ;
- public String getDocid() ;
- public booleanendOfCollection() ;

Plusieurs Classes implémentent l'interface collection selon le format du document :

SimpleFileCollection : PDF, TXT, HTML, … ,TrecCollection, SimpleXMLCollection : XML, … etc.

4.1.2. Document :

C'est une interface qui se trouve dans le package org.terrier.indexing (…\src\core\org\terrier\indexing). Ce composant permet de parcourir les documents et d'en extraire les termes en utilisant *Tokeniser.*

Les méthodes utilisées sont :

- public String getNextTerm() ;
- public booleanendOfDocument() ;

Plusieurs Parsers sont disponibles selon le format du document : HTMLDocument, FileDocument, MSExcelDocument…etc.

4.1.3. TermPipiline :

C'est une interface qui se trouve dans le package org.terrier.terms (…\src\core\org\terrier\terms). Ce composant permet de traiter les termes extraits du document :

- Elimine les mots vides (Stopwords)
- Lemmatise les termes selon la langue : pour l'anglais l'algorithme de lemmatisation utilisé est l'lagorithme de Porter (PorterStemmer).

4.1.4. Indexer :

Ce composant s'occupe de la gestion du processus d'indexation, notamment la construction de l'index et son écriture dans la structure de données appropriée.

Il existe deux types d'indexeur dans Terrier :Basicindexer, Blockindexer.

4.1.5. Les structures d'index :

Après indexation, les termes sont stockés en structures de données, le tableau 1 présente le contenu de ces différentes structures d'index :

Structure d'index	Contenu
Lexicon (Information du chaque terme de la collection)	• Terme • Id Terme • Nombre de documents qui contiennent le terme • Fréquence du terme dans la collection • Offset terme dans le fichier inverse
Inverted Index (Fichier inverse)	• Id Terme • Id document • Fréquence du terme dans le document • #Filds (#of fields bits)
Direct index (Index)	• Id Terme • Fréquence Terme • #Filds (#of fields bits)
Document Index	• Id document • Longueur document • Byte offset dans Direct Index
Propriétés Index	• Index sur les propriétés d'indexation (Exemple : Nombre de tokens dans l'index, …)

Tableau 1 : Présentation des différentes structures d'index

4.2. L'API de recherche :

Après le processus d'indexation, le processus de recherche est réalisé.

4.2.1. Query :

C'est une Class abstraite qui se trouve dans le package org.terrier.querying.parser (…\src\core\org\terrier\querying\parser). Un objet Query est créé pour chaque requête.

Terrier supporte trois modèles de requêtes :

- SingletermQuery : c'est un modèle de requête avec un seul terme.

- MUltiTermQuery : c'est un modèle de requête avec plusieurs termes.
- FieldQuery : c'est un modèle de requête qualifié par un champ (exemple : dans le titre du document).

4.2.2. Manager :

C'est un composant chargé de la gestion de la recherche. Il utilise à son tour quatre composants :

- **Pre-processing :** Il permet d'appliquer le Tokeniser et TermePipeline sur la requête.
- **Matching** : Il permet de déterminer les documents qui correspondent à la requête en initialisant le WeightingModels et DocumentScoreModifiers.

  - WeightingModels : Il assigne un score pour chaque terme de la requête dans le document, c'est le principe de la pondération. On trouve plusieurs modèles de pondération dans le package org.terrier.matching.models dont : TF_IDF, BM25…etc.

  - DocumentScoreModifiers : Il modifie le score d'un document en fonction du langage de la requête (ou expansion de la requête).
- **Post-filtring** : Il permet de filtrer les documents pertinents pour la requête.
- **Post-processing :** Il permet de reclasser les documents pertinents s'il y a une expansion de la recherche.

4.2.3. Set-Results :

Ce composant se charge de retourner les documents selon leur degré de pertinence.

5. Les applications de Terrier :

Terrier est livré avec trois applications :

- Desktop Terrier.
- Trec (Batch) Terrier.
- Interactive Terrier.

5.1. Desktop Terrier :

C'est une interface graphique pour l'indexation et la recherche de documents pertinents. Desktop Terrier ne permet pas d'évaluer les résultats de la recherche.

Pour exécuter Desktop Terrier sous Windows : double cliquer sur terrier-3.5\bin\desktop_terrier.bat.

5.2. Trec (Batch) Terrier :

TREC Terrier est conçu principalement pour l'indexation, la recherche et l'évaluation des résultats sur des Collections TREC. Un document de format TREC est délimité par les balises (tags) <DOC></DOC>.

<DOC>

<DOCNO> id doc </DOC>

Contenu du document…

</DOC>

Le trec (Batch) Terrier est exécuté en utilisant l'invite de commandes. C'est l'application le plus recommandée pour effectuer des travaux complexes de l'indexation et de la recherche.

5.2.1. Exécution de TREC Terrier sous Windows :

Pour démarrer, il faut spécifier dans l'invite de commandes le chemin vers le dossier bin de Terrier : *cd*<Path vers… \terrier-3.5\bin>.

Ø Processus d'indexation :

1. Supprimer le contenu du dossier index qui se trouve dans le dossier var de terrier (s'il est plein).
2. Initialisation de Terrier pour une nouvelle indexation en spécifiant le chemin vers la collection TREC à indexer avec la commande trec_setup :
   Trec_setup<path de la collection à indexer>
   Cela va générer trois fichiers dans le dossier etc :
   - **collection_spec** : qui contient les chemins vers les documents à indexer.
   - **terrier-log.xml** : qui utilise la bibliothèque Log4j.

- **terrier.propeties** : utiliser pour la configuration des propriétés de Terrier.

3. Modifier les propriétés de l'indexation de Terrier, et cela selon les besoins. En ajoutant ou en modifiant les propriétés du fichier *terrier.proprties* en consultant les propriétés possibles qui se trouvent dans le fichier *terrier.properties.sample*.

Par exemple :

- Si l'id document est supérieur à 20 caractères, définir la taille max des id (200 par exemple) : *indexer.meta.forward.keylens = 200*
- Modifier la taille max d'un terme dans le document : *max.term.length = 300* (taille max 300 caractères)
- Indiquer à l'indexeur de supprimer les mots vides et de ne pas lemmatiser : *termpipelines = Stopwords*
- Indiquer à l'indexeur de ne pas supprimer les mots vides et de limmatiser : *termpipelines = WeakPrterStemmer*

4. Après avoir configuré le fichier terrier.properties, lancer l'indexation à l'aide de la commande : *trec_terrier –i*

   Cela va générer dans le dossier …\terrier-3.5\var\index les différents fichiers binaires correspondants aux structures d'index. Le tableau 2 présente les différentes commandes utilisées pour afficher les structures d'index dans l'invite de commandes.

Fichier structure d'index	Commande
.data.lexicon (Lexicon)	trec_terrier --printlexicon
data.docid (Document Index)	trec_terrier --printdocid
data.inverted (Inverted Index)	trec_terrier --printinverted
data.direct (Direct Index)	trec_terrier --printdirect
data.meta	trec_terrier --printmeta
data.properties	trec_terrier --printstats

Tableau 2 : Les différentes commandes pour afficher les structures d'index

Ø Processus de recherche :

1. Ajouter le propriété trec.topics dans le fichier terrier.properties qui spécifie le chemin vers le fichier texte contenant les requêtes :
   trec.topics=<Path vers le fichier txt contenant les requêtes>
2. Modifier les propriétés de la recherche de Terrier, selon les besoins (même procédé que l'indexation).
   Par exemple :
   - Indiquer au Perser d'analyser une requête simple par ligne :
     trec.topics.parser=SingleLineTRECQuery
   - Configuration du modèle de pondération :
     Trec.model= TF_IDF (utiliser le modèle TF_IDF).
3. Lancer la recherche à l'aide de la commande : trec_terrier –r
   Le résultat de la recherche sera stocké dans le fichier x (.res), dans …\terrier-3.5\var\result\x.res
   Format du fichier (.res) : num_requête Q0 id_Doc Rang Score Modèle-Pondération.

Ø Processus d'évaluation :

1. Spécifier dans le fichier terrier.properties le chemin vers le Rel_Ass qui contient les documents pertinents pour chaque requête.
   trec.qrels= <path vers fichiers Rel_Ass.qrels>
2. Lancer l'évaluation à l'aide de la commande : *trec_terrier –e*
   Le fichier évaluation.x (.eval) sera dans : …\terrier-3.5\var\result\x.eval.

5.3. Interactive Terrier :

Interactive Terrier est une interface graphique pour la recherche de documents pertinents. Cette application permet une recherche interactive, c'est un moyen facile de tester Terrier.

Pour l'exécuter sous Windows : double cliquer sur

…terrier-3.5\bin\interactive_terrier.bat

6. **Compilation de Terrier** :

Terrier est extensible. En effet, on peut modifier son code source et lui intégrer de nouveaux modules (classes). Pour ce faire, on doit recompiler terrier.

Pour compiler Terrier, on a besoin d'utiliser le Ant de l'environnement IDE Eclipse et le fichier build.xml de Terrier-3.5.

Annexe III : Stanford POS Tagger

Stanford Part-of-Speech Tagger est un étiqueteur morphosyntaxique et un lemmatiseur développé par The Stanford Natural LanguageProcessing Group. Il est distribué librement à des fins d'évaluation, de recherche ou d'enseignement. Pour l'étiquetage, il implémente une méthode probabiliste (arbres de décision) nécessitant une phase d'entraînement ; Il est donc possible de développer une version spécifique selon la langue pour laquelle on souhaite l'utiliser. Une version dédiée à l'Anglais est ainsi disponible sur la page d'accueil du Stanford POS Tagger : seul le fichier de paramètres varie : le moteur probabiliste reste inchangé. La liste des catégories grammaticales utilisées par Stanford POS Tagger pour l'Anglais est présentée dans le tableau ci-dessous. Cette liste correspond aux jeux d'étiquetage Treebank Tag set.

Catégorie	Signification
CC	Conjonction
CD	Nombre
DT	Determinant (the, a, all, and, both, etc…)
EX	There
FW	Mot ou expression étrangère
IN	Preposition (across, after, as, for, in, etc…)
JJ	Adjectif
LS	Reference
MD	Auxiliares(con, should, may, would, will, might)
NN	Nom
NNP	Nom proper
NNPG*	Nom de groupe (société, association, etc…)
NNPL*	Nom de lieu
NNP	Nom de personne

NNS	Pluriels
PDT	All, both, that, this
POS	‘s possessif
PRP	Pronom personnel
PRP$	Pronompossessif
RB	Adverbe
RBR	Adverbe de comparaison (better, etc…)
SPUNC’	Ponctuation forte
TO	Infinitif to
UH	Interjection
UNK	Mot inconnu
VB	Verbe
VBD	Verbe au passé
VBG	Participe présent
VBN	Participe passé
VBP	Auxiliaires (be, do, have, is, does, has)
WDT	That, whatever, which
WP	Whadya, what, who, whom, adjectifsinterrogatifsrelatifs
WP$	Whose
WPUNC’	Ponctuation faible
WRB	How, when, whence, whenever, where, whereby, wherever, why
ZTRM’	Fin de phrase

Liste des catégories grammaticales

Bibliographie

Aronson, A. R. (2006). MetaMap: Mapping Text to the UMLS Metathesaurus.

Boubeker, F AZZOUG, W. (2011). Indexation sémantique de documents textuels.

BAZIZ, M. (2005). INDEXATION CONCEPTUELLE GUIDEE PAR ONTOLOGIE POUR LA RECHERCHE D'INFORMATION. *INSTITUT DE RECHERCHE EN INFORMATIQUE DE TOULOUSE* .

Boubekeur, F. (2008). Contribution à la définition de modèles de recherche d'information flexibles basés sur les CP-Nets. *Thèse de Doctorat* .

Collier, N. N.-i. (2000). Extracting the names of gene products with a hidden Markov model. *18th international conference on Computational linguistics* , 201-207.

Dan Shen, J. Z.-L. (2003). *Effective adaptation of a Hidden Markov Model-based Named Entity Recognizer For Biomedical Domain.*

DINH, B.-D. (2012). Accès à l'information biomédicale : vers une approche d'indexation et de recherche d'information conceptuelle basée sur la fusion de ressources termino-ontologiques. *Doctorat de l'université de Toulouse 3 Paul Sabatier* .

Frantzi, K. A. (2000). Automatic recognition of multi-word terms : the C-value/NC-value method. *International Journal on Digital Libraries* , 115–130.

Fukuda, K. T. (1998). Toward Information Extraction : Identifying protein names from biological papers. *Proceedings of Pacific Symposium on Biocomputing* , 707–718.

Gaizauskas, R. D. (2003). Protein structures and information extraction from biological texts : the PASTA system. *Bioinformatics,* , 135–143.

Krauthammer, M. R. (2000). Using BLAST for identifying gene and protein names in journal articles. *Gene* , 245-252.

Narayanaswamy M., R. K.-S. (2003). A Biological Named Entity Recognizer. *Pacific Symposium on Biocomputing* , 427–438.

Xiaohua Zhoo, X. a. (2006). *MaxMatcher : Biological concept extraction using approximate dictionary lookup.*

Zhou Xiaohua, X. Z. (2006). MaxMatcher: Biological Concept Extraction Using Approximate Dictionary Lookup. *College of Information Science & Technology, Drexel University 3141 Chestnut Street, Philadelphia, PA 19104* , 1-5.

Rijsbergen, C. (1979). *Information retrieval, Second edition.* Butterworths.

Salton, G. (1989). *Automatic text processing : the transformation, analysis,*

and retrieval of information by computer. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Baziz, M. (2005). *Indexation conceptuelle guidée par ontologie pour la recherche d'information*. Thèse de doctorat, Université Paul Sabatier, Toulouse, France.

Boubekeur, F. (2008). *Contribution à la définition de modèles flexibles de recherche d'information basés sur les CP-Nets*. Thèse de doctorat, Université Paul Sabatier.

Singhal, A. et Pereira, F. (1999). Document expansion for speech retrieval. *In SIGIR'99 Conference on Research and Development in Information Retrieval*, pages 34–41, New York, NY, USA. ACM.

Maron, M. E. et Kuhns, J. L. (1960). On Relevance, Probabilistic Indexing and Information Retrieval. *J. ACM*, 7:216–244.

Cleverdon, C. (1967). The Cranfield tests on index language devices. *In Aslib Proceedings*, pages 173–194.

Resnik, P. (1999). Semantic Similarity in a Taxonomy : An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. *J. Artif. Intell. Res. (JAIR)*, 11:95–130.

Leacock, C., Towell, G. et Voorhees, E. (1993). Corpus-based statistical sense resolution. *In Proceedings of the workshop on Human Language Technology*, HLT '93, pages 260–265. Association for Computational Linguistics.

Jiang, J. et Conrath, D. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. *In International Conference on Research in Computational Linguistics (ROC)*, pages 19–33.

Lin, D. (1998). An information-theoretic definition of similarity. *In Proceedings of the Fifteenth International Conference on Machine Learning*, ICML '98, pages 296–304, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

Agirre, E. et Rigau, G. (1996). Word sense disambiguation using conceptual density. *In International Conference on Computational Linguistics (COLING)*, pages 16–22.

Fellbaum, C., éditeur (1998). *WordNet : An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, illustrated edition édition.

Mihalcea, R., Tarau, P. et Figa, E. (2004). PageRank on semantic networks with application to word sense disambiguation. *In International Conference on Computational Linguistics (COLING)*, pages 1126–1132.

Khan, L., Leod, D. M. et Hovy, E. (2004). Retrieval effectiveness of an ontology-based model for information selection. *The VLDB Journal*, 13:71–
85.

Voorhees, E. M. (1993). Using WordNet to Disambiguate Word Senses for Text Retrieval. *In SIGIR*, pages 171–180.

Bourigault, D., Aussenac-Gilles, N. et Charlet, J. (2004). Construction de ressources terminologiques ou ontologiques à partir de textes Un cadre unificateur pour trois études de cas. *Revue d'Intelligence Artificielle*, pages 87–110.

Foskett, D. J. (1997). Readings in information retrieval. chapitre Thesaurus, pages 111–134. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge acquisition*, 5:199–220.