République Algérienne Démocratique et Populaire
Ministère de L'Enseignement Supérieur et de la Recherche Scientifique
Université Mouloud Mammeri de Tizi-Ouzou
Faculté de Génie Electrique et d'Informatique
Département d'Informatique

Mémoire

de fin d'études

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

Spécialité : Systèmes Informatiques

Thème :

Indexation sémantique de documents textuels

Proposé et dirigé par :

M^me^ F.AMIROUCHE

Réalisé par :

M^elle^ DOUFENE Lynda

Promotion: 2011 / 2012

Je dédie ce modeste travail,

A mes chers parents, à mon frère Samy et à ma sœur Sabrina.

Remerciements

Je tiens tout particulièrement à remercier ma promotrice M^me^ AMIROUCHE Fatiha pour avoir accepté de m'encadrer, pour sa disponibilité, sa compréhension et ses conseils.

Je remercie également M^elle^ AZZOUG Wassila étudiante en Magister pour son aide précieuse et pour sa gentillesse.

Mes remerciements s'adressent aussi aux membres du jury, pour avoir accepté de juger ce travail.

Table des Matières

Chapitre II : Indexation sémantique en RI.

Chapitre III : Notre approche d'indexation sémantique.

Liste des figures et tableaux

Chapitre I : La recherche d'information (RI).

Chapitre II : Indexation sémantique en RI.

Chapitre III : Notre approche d'indexation sémantique.

Chapitre IV : Implémentation et tests.

Introduction générale

La recherche d'information (RI), est une branche de l'informatique qui s'intéresse à l'acquisition, l'organisation, le stockage et la recherche des informations. Pratiquement, il s'agit d'ensemble de procédures et techniques permettant de sélectionner, parmi un ensemble de documents, les informations pertinentes en réponse à un besoin en information exprimé par l'utilisateur à travers une requête.

Le développement rapide des technologies de l'information durant ces dernières années a eu pour conséquence une prolifération de sources d'informations hétérogènes. Il devient de plus en plus difficile pour un système de recherche d'information (SRI) de retrouver précisément ce que les utilisateurs recherchent dans cette masse de données. Le problème qui se pose actuellement n'est plus tant la disponibilité de l'information mais la capacité d'accès et de sélection de l'information répondant aux besoins précis d'un utilisateur, à partir des représentations qu'il perçoit. Le développement d'outils de recherche efficaces, permettant notamment à l'utilisateur de n'avoir accès qu'à l'information qu'il juge pertinente, devient une nécessité absolue.

Pour améliorer la qualité des SRI, plusieurs approches ont été proposées. Les premières approches qualifiées de classiques, se basent sur une recherche par mots clés, les documents (respectivement requêtes) sont représentés comme des sacs de mots clés souvent pondérés, et la pertinence d'un document vis-à-vis d'une requête est souvent estimée en s'appuyant sur les fréquences d'apparition des mots de la requête dans ces mêmes documents. Ces approches ne tiennent pas compte des sens des mots, ce qui pose problème vu la richesse et l'ambiguïté de langue. Pour pallier ces problèmes, d'autres approches se basant sur les sens des mots pour représenter les documents et les requêtes sont apparues à travers l'indexation sémantique.

Dans le cadre de ce mémoire de Master notre objectif est d'une part, d'implémenter une nouvelle approche d'indexation sémantique, et d'autre part, d'intégrer le module qui en résulte à la plateforme de RI Terrier.

Notre mémoire s'articule autour de quatre chapitres comme suit :

Dans le premier chapitre nous présentons les différents aspects liés à la recherche d'information et aux SRI. Nous nous y intéressons en particulier au processus de RI et aux principaux modèles de RI.

Le deuxième chapitre sera consacré à la présentation des approches d'indexation sémantique. Nous évoquons l'importance de l'indexation sémantique et nous présentons les travaux les plus connus basés sur la désambiguïsation des sens des mots.

Nous présentons dans le troisième chapitre la partie contribution de ce mémoire. La description de notre approche d'indexation sémantique basée sur l'utilisation conjointe de WordNet et du Stanford POS Tagger pour la désambiguïsation des mots et l'intégration de cette approche à la plateforme de RI Terrier 3.5, constituant ainsi une nouvelle plateforme appelée Sem-Terrier.

Le quatrième chapitre présente quelques aspects liés à l'implémentation de notre approche et son intégration dans Terrier 3.5, ainsi que les expérimentations et résultats de notre système résultant Sem-Terrier. Enfin, nous concluons ce mémoire et présentons les perspectives envisagées pour la continuité de ce travail.

Chapitre I

La Recherche d'information (RI)

I.1. Introduction :

Les systèmes de recherche d'information (SRI), servent d'interface entre une source (collection) contenant des quantités considérables de documents et des utilisateurs cherchant, via des requêtes, des informations susceptibles de se trouver dans cette collection. Les SRI intègrent un ensemble de techniques permettant de sélectionner ces informations. Ces techniques peuvent être résumées en quatre fonctions, qui sont le stockage de l'information, l'organisation de ces informations, la recherche d'informations en réponse à des requêtes utilisateurs et la restitution des informations pertinentes pour ces requêtes. La dernière fonction est celle qui est visible pour l'utilisateur.

Ce chapitre traite des concepts, techniques et modèles étudiés dans le domaine de la recherche d'information. Il est organisé comme suit :
Dans la première section on présente les notions de base de la Recherche d'Information (RI), la deuxième section est consacrée à la description du processus de RI, dans laquelle sont définis les notions d'indexation, de reformulation de requêtes et d'appariement document-requête, la troisième section introduit les principaux modèles de RI, ainsi que les outils et méthodes d'évaluation d'un SRI.

I.2. Notions de base de la RI :

I.2.1 Définition d'un SRI :

Un système de recherche d'information (SRI) est un système informatique qui permet de retrouver, parmi une collection de documents préalablement stockés, les documents qui répondent à un besoin en information d'un utilisateur exprimé sous forme de requête. Pour cela, un SRI met en œuvre un ensemble de processus de sélection des documents pertinents pour la requête.

Plusieurs concepts clés s'articulent autour de cette définition :

- **Document** : Un document peut être un texte, un morceau de texte, une page Web, une image, une bande vidéo, etc. On appelle document toute unité ou granule documentaire qui peut constituer une réponse à une requête utilisateur.
- **La collection de documents :** la collection de documents (ou fond documentaire, corpus) constitue l'ensemble des informations exploitables et accessibles. Elle est constituée d'un ensemble de documents. Dans le cas général et dans un souci d'optimalité, la base constitue des représentations simplifiées mais suffisantes pour ces documents. Ces représentations sont étudiées de telle sorte que la gestion (ajout suppression d'un document) ou l'interrogation (recherche) de la base se font dans les meilleures conditions de coût [Baziz, 05].
- **Besoin en information** : Le besoin en information est souvent assimilé au besoin de l'utilisateur. Trois types de besoins utilisateur ont été définis par [Ingwersen,92]:
  a) *Besoin vérificatif* : l'utilisateur cherche à vérifier le texte avec les données connues. Il recherche une donnée particulière, et sait souvent comment y accéder (par exemple chercher un document ayant une adresse connue sur le web).
  b) *Besoin thématique connu* : l'utilisateur cherche à clarifier, à revoir ou à trouver de nouvelles informations dans un sujet et un domaine connus (il est possible que le besoin s'affine au cours de la recherche).
  c) *Besoin thématique inconnu*: l'utilisateur cherche de nouveaux concepts ou de nouvelles relations hors des sujets ou domaines qui lui sont familiers.

- **Requête :** C'est l'expression du besoin de l'utilisateur. Elle est l'interface entre le SRI et l'utilisateur. Une requête peut être soit un ensemble de mots clés, ou exprimée en langage naturel, booléen ou graphique.
- **Pertinence** : La pertinence est une notion fondamentale en RI. On peut la définir comme le degré de correspondance entre un document et une requête. De façon générale, dans le document pertinent, l'utilisateur doit pouvoir trouver les informations dont il a besoin. C'est sur cette notion de pertinence que le système doit juger si un document doit être donné à l'utilisateur comme réponse [GIL, 06].

I.2.2. Processus de recherche d'information :

L'objectif fondamental d'un processus de RI est de sélectionner les documents "les plus proches" du besoin en information de l'utilisateur décrit par une requête. Pour cela, le système de recherche regroupe un ensemble de méthodes et procédures permettant la gestion des collections de documents stockés sous forme d'une représentation intermédiaire permettant de refléter aussi fidèlement que possible leurs contenus. L'interrogation de la collection de documents à l'aide d'une requête nécessite la représentation de cette dernière sous une forme unifiée compatible avec celles des documents [Zemirli, 08]. Ces fonctionnalités sont représentées à travers le processus global de la RI, communément nommé processus en U et schématiquement illustré par la figure I.1.

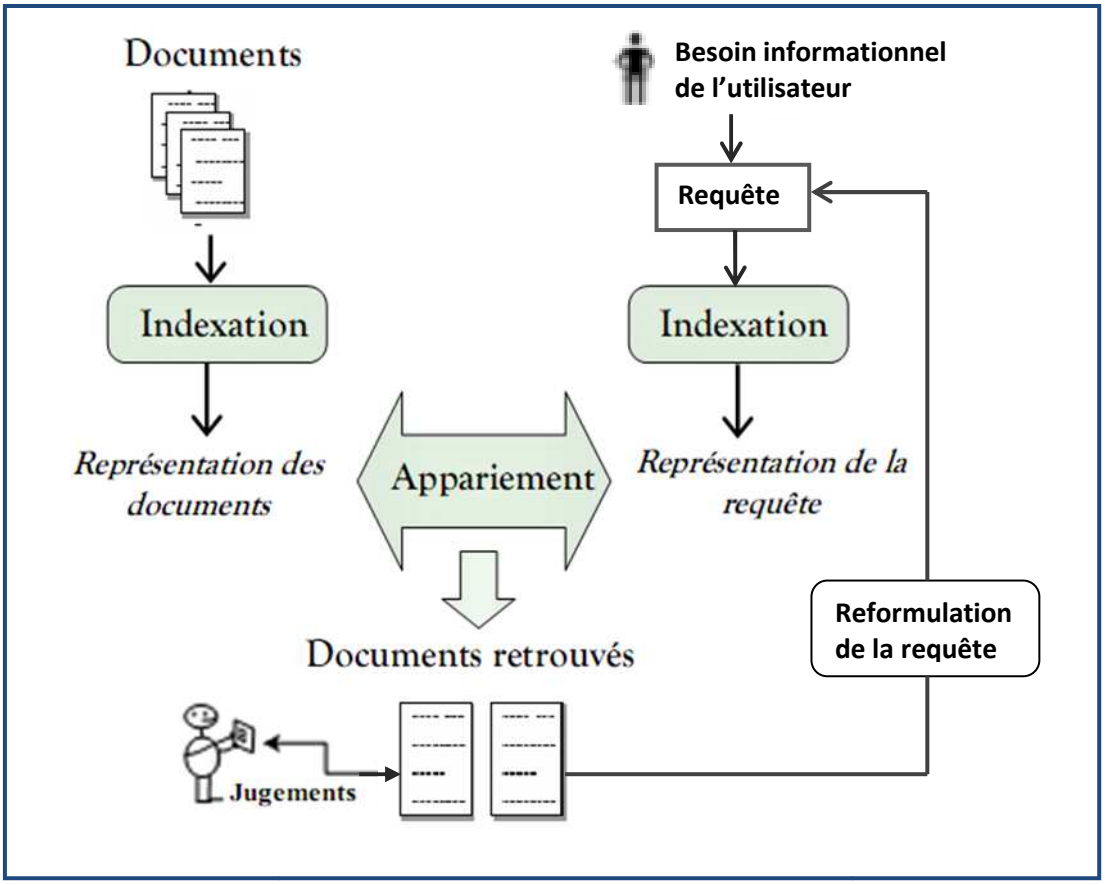


Figure I.1 : Processus général de recherche d'information

Le déroulement de ce processus induit deux principales phases : l'indexation et l'appariement requête/document :

- **L'indexation :** Elle consiste à déterminer et à extraire les termes représentatifs du contenu d'un document ou d'une requête.
- **L'appariement** : il représente le processus du noyau d'un SRI. Il comprend la fonction de décision fondamentale qui permet d'associer à une requête, l'ensemble des documents pertinents à restituer.

I.3. L'indexation :

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. Une bonne indexation doit permettre de retrouver tous les documents pertinents au besoin de l'utilisateur et pas (ou peu) de documents non pertinents pour celui-ci.

En phase d'indexation, le document (ou la requête) est analysé(e) et les mots clés caractérisant son contenu informationnel, sont extraits. Un mot clé peut-être soit un mot simple ou un groupe de mots. Les mots-clés descriptifs du contenu sémantique d'un document sont dits termes d'indexation (appelés aussi descripteurs). L'ensemble de tous les termes d'indexation constitue le langage d'indexation [Amirouche, 08]. Ce langage peut-être libre ou contrôlé :

- **Langage d'indexation libre** : Ce langage est construit à partir des termes extraits du document analysé.
- **Langage d'indexation contrôlé** : Ce langage est construit à partir d'un ensemble de termes préalablement définis et organisés généralement dans un thésaurus[1]. Lorsqu'un document est analysé, on ne garde que les mots clés qui appartiennent à ce thésaurus.

I.3.1. Mode d'indexation :

L'indexation peut être effectuée selon trois modes différents : manuelle, automatique ou semi-automatique [Baziz, 05], [Savoy, 03].

- **L'indexation manuelle :** Chaque document est analysé par un spécialiste du domaine ou par un documentaliste, qui se charge de construire l'ensemble des termes représentants le contenu du document. L'indexation manuelle est difficilement réalisable dans le cas de collections volumineuses. Elle présente également un aspect subjectif. En effet, des termes différents peuvent être utilisés par deux documentalistes différents pour représenter un même document, et un même indexeur peut à deux moments différents, utiliser deux termes distincts pour représenter le même concept.
- **L'indexation automatique :** c'est un processus complètement automatisé qui se charge d'extraire les termes caractéristiques du document. L'intérêt d'une telle approche réside dans sa capacité à traiter les textes nettement plus rapidement que l'approche précédente, et de ce fait, elle est particulièrement adaptée aux corpus volumineux. L'indexation automatique est l'approche la plus étudiée en RI, nous la détaillons en section suivante.

[1] C'est un vocabulaire contrôlé et dynamique de termes, obéissant à des règles terminologiques propres et reliés entre eux par des relations sémantiques.

- **L'indexation semi-automatique :** appelée aussi indexation supervisée, est une combinaison des deux approches d'indexation précédentes. dans ce mode, un premier processus automatique permet d'extraire les termes du document. Cependant, le choix final reste au spécialiste du domaine ou au documentaliste pour établir les relations entre les mots clés et choisir les termes significatifs.

I.3.2. L'indexation automatique :

L'indexation automatique regroupe un ensemble de traitements automatisés sur un document. Ces traitement se résument à :

- L'identification des termes d'indexation.
- La normalisation des termes.
- La pondération des termes l'index.

I.3.2.1. L'identification des termes d'indexation :

L'objectif est de trouver les mots qui représentent au mieux le contenu d'un document, ce traitement comprend :

- **L'extraction automatique des termes :**

Cette opération consiste à travers une analyse lexicale à transformer un document en un ensemble de termes. Un terme est un groupe de caractères constituant un mot significatif. Durant cette opération, les espaces, les ponctuations, les caractères spéciaux et la mise en page sont supprimés [Champclaux, 09].

A travers les travaux de la RI, nous pouvons recenser les différents types de descripteurs (termes) utilisés [Said l'hadj, 09]:

- Les mots simples non vides.
- Les lemmes ou les racines des mots simples,
- Les n-grammes qui sont une représentation originale d'un texte en séquences de N caractères consécutifs.
- Les groupes de mots : un groupe de mots est souvent plus riche sémantiquement que les mots qui le composent pris isolément. Cet argument a conduit à considérer les groupes de mots comme unités de base dans le langage d'indexation.

- Les contextes c'est-à-dire les termes n'apparaissant pas explicitement dans le texte du document mais ayant un lien sémantique et/ou de cooccurrence avec les mots du document.
- Les concepts qui sont des expressions contenant un ou plusieurs mots. Ils peuvent être écrits de manière libre par l'utilisateur ou, ce qui est souvent le cas, choisis dans une liste de concepts (vocabulaire contrôlé : thésaurus, ontologie, hiérarchie de concepts, …etc.).

- **L'élimination des mots vides :**

Un des problèmes majeurs de l'indexation consiste à extraire les termes significatifs et à éviter les mots vides (pronoms personnels, prépositions,...). Les mots vides peuvent aussi être des mots athématiques (les mots qui peuvent se retrouver dans n'importe quel document parce qu'ils exposent le sujet mais ne le traitent pas, comme par exemple : contenir, appartenir, …). On distingue deux techniques pour éliminer les mots vides :

- L'utilisation d'une liste de mots vides (aussi appelée anti-dictionnaire) : quand un mot est rencontré dans un texte à indexer, s'il apparaît dans l'anti-dictionnaire, il n'est pas considéré comme terme d'index [Salton & al., 88].
- L'utilisation de l'information sur les fréquences d'occurrences des mots : Le principe consiste à éliminer des mots dépassant une certaine fréquence d'occurrences ou ayant une fréquence d'occurrence quasi nulle.

I.3.2.2. La normalisation des termes :

Ce traitement consiste à retrouver, pour un mot donné sa forme normalisée (généralement le masculin pour les noms, l'infinitif pour les verbes, le masculin singulier pour les adjectifs…etc.). Le traitement associé à la normalisation repose sur deux procédures : la troncature et la lemmatisation.

- **La troncature :** est utilisée pour éliminer les variations morphologiques ou les variantes orthographiques des mots-clés. Elle consiste à couper le mot à partir d'un rang précis, afin d'obtenir son radical. Pour la langue française, la troncature à sept caractères est adoptée.

- **La lemmatisation** : est un processus morphologique permettant de regrouper les variantes d'un mot. En effet, on remarque que beaucoup de mots ont des formes différentes, mais leur sens reste le même ou très similaire et notamment dans le cas des mots conjugués. Ces mots ont la même racine (lemme). Ainsi, on arrive à éliminer les terminaisons des mots, et garder seulement la racine, on a donc une forme identique pour tous ces mots.

I.3.2.3. La pondération des termes :

La pondération est l'une des fonctions fondamentales en RI. Elle consiste à associer un poids d'importance (ou valeur de représentativité) w_{ij} à chaque terme t_i d'un document d_j [Amirouche, 08]. De manière générale, la majorité des formules de pondération des termes est construite par combinaison de deux facteurs. Un facteur de pondération locale *tf* (term frequency), quantifiant la représentativité locale d'un terme dans le document, et un second facteur de pondération globale idf (inverted document frequency), mesurant la représentativité globale du terme vis-à-vis de la collection des documents [Tebri, 04]. Plusieurs formules de pondération existent, dont :

- La formule de pondération tf * idf :

$$w_{ij} = \frac{tf_{ij}}{df_i} = tf_{ij} \times \frac{1}{df_i} = tf_{ij} \times idf_i$$

Où *:*

w_{ij} : est le poids du terme t_i dans le document d_j.

tf_{ij} : est la fréquence d'occurrence du terme t_i dans le document d_j ($f(t_i,d_j)$).

df_i : est la fréquence documentaire du terme t_i (fréquence d'un terme t_i dans toute la collection).

Idf_i : est la fréquence documentaire inverse, définie classiquement par:

$$Idf_i = \log (N / N_i)$$

Tel que *:* N *:* est le nombre de documents de la collection

N_i : le nombre de documents contenant le terme *ti*.

La mesure tf * idf donne une bonne approximation de l'importance du terme dans le document, particulièrement dans les corpus de documents de tailles homogènes.

- La formule de pondération BM25 :

[Robertson et al., 1997] proposent d'intégrer la taille des documents à la formule de pondération de la façon suivante :

$$w_{ij} = \frac{tf_{ij}(k_1 + 1)}{k_1\left[(1 - b) + b\frac{dl_j}{\Delta l}\right] + tf_{ij}}$$

Où:

w_{ij} : est le poids du terme t_i dans le document d_j.

K_1 : constante qui permet de contrôler l'influence de la fréquence du terme t_i dans le document d_j. Sa valeur dépend de la longueur des documents dans la collection. Le plus souvent, sa valeur est fixée à 1,2.

b : constante qui permet de contrôler l'effet de la longueur du document. Sa valeur la plus souvent utilisée est : 0,75.

dlj : est la longueur du document di.

Δl : est la longueur moyenne des documents dans la collection entière.

I.4. Reformulation de Requêtes :

L'utilisateur exprime son besoin en information sous forme d'une requête afin de trouver des résultats qui l'intéressent. Cependant, le SRI renvoi parfois des résultats qui ne lui conviennent pas. Pour cela, une étape de reformulation de la requête est souvent utilisée dans l'objectif de retrouver plus de documents pertinents. Ce processus permet de générer une requête plus adéquate que celle initialement formulée par l'utilisateur.

La reformulation de la requête consiste à modifier la requête de l'utilisateur par ajout de termes significatifs et/ou suppression de termes inutiles. Les termes rajoutés proviennent soit des documents de la collection, on parle alors de réinjection de pertinence (relevance feedback), ou sont issus de ressources externes telles que les ontologies ou les thesaurus.

I.5. L'appariement document-requête :

La comparaison entre le document et la requête revient à calculer un score, supposé représenter la pertinence du document vis-à-vis de la requête. Cette valeur est calculée à partir d'une fonction ou d'une mesure de similarité notée *RSV (Q,d)* (Retrieval Status Value), où *Q* est une requête et *d* un document. Cette mesure tient compte du poids des termes dans les documents déterminés en fonction d'analyses statistiques et probabilistes.

La fonction d'appariement est très étroitement liée aux opérations d'indexation et de pondération des termes de la requête et des documents du corpus. D'une façon générale, l'appariement document-requête et le modèle d'indexation permettent de caractériser et d'identifier un modèle de recherche d'information [Mammeri ,09] .
La fonction d'appariement permet ensuite d'ordonner les documents renvoyés à l'utilisateur.

Il existe deux types d'appariement [Zemirli, 04] :

- Appariement exact :

Le résultat est une liste de documents respectant exactement la requête spécifiée avec des critères précis. Les documents retournés ne sont pas triés.

- Appariement approché :

Le résultat est une liste de documents sensés être pertinents pour la requête. Les documents retournés sont triés selon un ordre de mesure. Cet ordre reflète le degré de pertinence document/requête.

I.6. Les modèles de recherche d'information :

Le modèle joue un rôle central en RI. C'est le modèle qui détermine le comportement clé d'un système de RI. Ainsi, si c'est l'indexation qui choisit les termes pour représenter le contenu d'un document ou d'une requête, c'est au modèle de leur donner une interprétation.
Étant donné un ensemble de termes pondérés issus de l'indexation, le modèle remplit les deux rôles suivants [Mammeri ,09] :

- Créer une représentation interne pour un document ou pour une requête basée sur ces termes.
- Définir une méthode de comparaison entre une représentation de document et une représentation de requête afin de déterminer leur degré de correspondance (ou similarité).

Les modèles de RI se déclinent en trois grandes catégories qui sont :

- Les modèles ensemblistes.
- Les modèles algébriques.
- Les modèles probabilistes.

La figure I.2 présente une classification des différents modèles de RI [Baeza-Yates & al., 99].

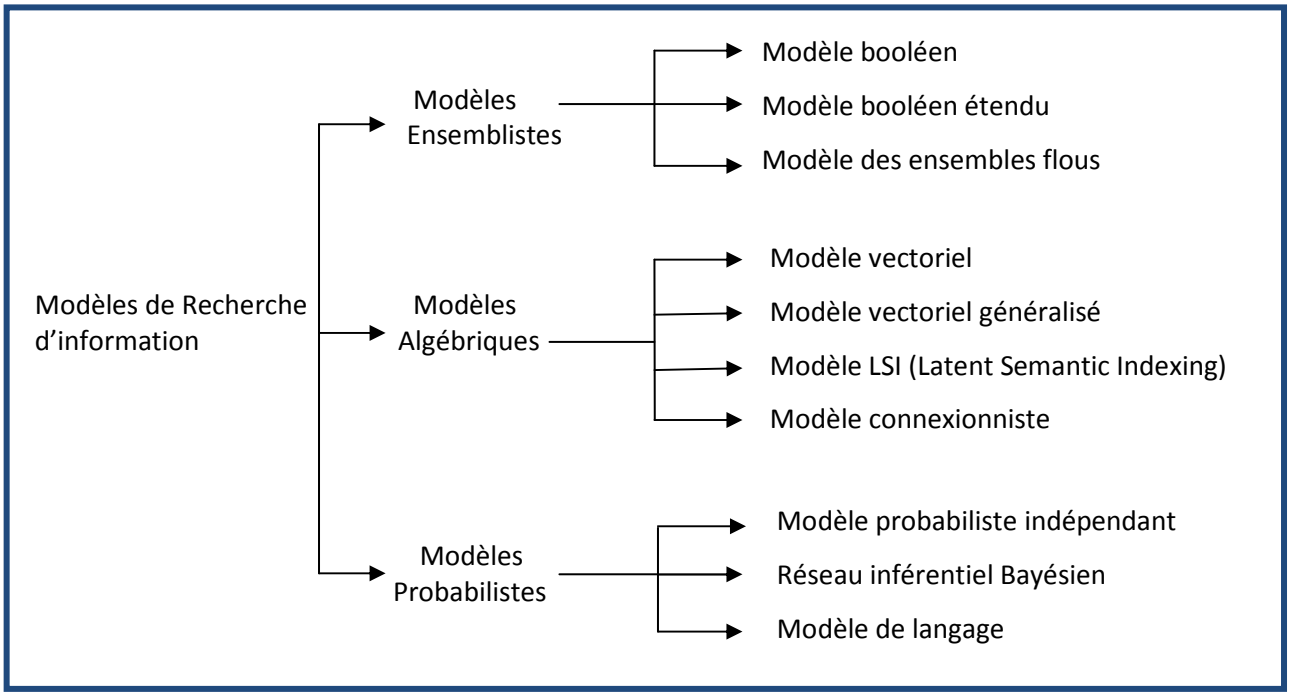


Figure I.2 : Taxonomie des modèles de RI

I.6.1. Les modèles reposant sur la théorie des ensembles :

I.6.1.1. Le modèle booléen de base :

Le modèle booléen est le plus ancien de tous les modèles de RI conventionnels. Dans ce modèle chaque document *d* est représenté comme une conjonction logique de termes (non pondérés) qui le composent, par exemple :

$$d = \{t_1, t_2, \ldots, t_n\}.$$

Une requête est représentée par une expression logique quelconque de termes utilisant les opérateurs booléens (AND, OR et NOT). Par exemple :

$$q = (t_1 \text{ and } t_2) \text{ ou } (t_3 \text{ et } (\text{not } t_4)).$$

La correspondance $RSV(d_j,Q_k)$ entre une requête Q_k et un document d_j, (tel que q_i est un terme de la requête Q_k) est déterminée comme suit :

$RSV(d_j,q_i)$ = 1 si $q_i \in d_j$; 0 sinon,

$RSV(d_j, q_1 \wedge q_2)$ = 1 si $RSV(d_j,q_1) = 1$ et $RSV(d_j,q_2) = 1$; 0 sinon,

$RSV(d_j, q_1 \vee q_2)$ =1 si $RSV(d_j,q_1) = 1$ ou $RSV(d_j,q_2) = 1$; 0 sinon,

$RSV(d_j, \neg q_i)$ =1 si $q_i \notin d_j$; 0 sinon,

Ce modèle se caractérise par sa simplicité et son caractère intuitif. Il est reconnu pour sa force à faire une recherche très restrictive et obtenir, pour un utilisateur expérimenté, une information exacte et spécifique.

Son inconvénient majeur est que les documents pertinents dont la représentation ne correspond qu'approximativement à la requête ne sont pas sélectionnés, les résultats sont ainsi classés dans deux catégories : l'ensemble des documents pertinents et l'ensemble des documents non pertinents. Pour remédier à ces inconvénients, le modèle booléen étendu a été développé.

I.6.1.2. Le modèle booléen étendu :

Le modèle booléen étendu appelé aussi modèle PNorm, a été introduit en 1983 par Salton et al [Salton & al, 83]. Son principe de base est de compléter le modèle de base, en affectant des poids à chaque terme du document et de la requête. Ainsi, les documents restitués par le SRI sont ordonnés.

La similarité entre un document d_j et une requête Q_k décrite sous une forme conjonctive ou disjonctive est donnée comme suit :

Opérateur OR : $$RSV(d_j, Q_k) = \left(\frac{\sum_{i=1}^{n} q_{ik}^{p} \; d_{ij}^{p}}{\sum_{i=1}^{n} q_{ik}^{p}}\right)^{\frac{1}{p}}$$

Opérateur AND: $$RSV(d_j, Q_k) = 1 - \left(\frac{\sum_{i=1}^{n} q_{ik}^{p} \left(1 - d_{ij}^{p}\right)}{\sum_{i=1}^{n} q_{ik}^{p}}\right)^{\frac{1}{p}}$$

Où :

n est le nombre de terme dans d_i (t_1,…….t_n).

d_{ij} est le poids du terme t_i dans le document d_j, avec $1 \leq i \leq n$ et $0 \leq d_{ij} \leq 1$.

q_{ik} est le poids du terme t_i dans la requête Q_k.

p est une constante $0 \leq P \leq \infty$.

Lorsque p=1, il n'y a aucune distinction entre les deux connecteurs AND et OR. Par conséquent, on se ramène au modèle booléen.

I.6.2. Les modèles algébriques :

I.6.2.2.Le modèle vectoriel :

Plutôt que de chercher une correspondance "logique" entre documents et requêtes, le modèle vectoriel [Salton,71] aborde le problème de la recherche d'information de manière algébrique en s'appuyant sur la théorie mathématique des espaces vectoriels et plus généralement de l'algèbre linéaire.

Dans ce modèle, les documents et les requêtes sont représentés sous forme de vecteurs, dans l'espace vectoriel engendré par tous les termes d'index $(t_1, t_2, t_3, \ldots, t_N)$. Les coordonnées d'un vecteur document (ou requête) sont les poids indiquant l'importance du terme correspondent par rapport au document (ou à la requête). De manière formelle, un document d_i (ou requête Q) est un vecteur dans un espace vectoriel de dimension N, il (elle) est représenté(e) comme suit :

$$d_j = \begin{Bmatrix} w_{1,j} \\ w_{2,j} \\ \vdots \\ w_{n,j} \end{Bmatrix} \text{ pour } j=1,..,m \quad \text{et} \quad Q = \begin{Bmatrix} w_{1,Q} \\ w_{2,Q} \\ \vdots \\ w_{n,Q} \end{Bmatrix}$$

Où :

$w_{i,j}$ représente le poids du terme t_i dans le document d_j.

$w_{i,Q}$ représente le poids du terme t_i dans la requête. Ce poids peut être soit une forme de tf * idf (pondération utilisée pour les documents), soit un poids attribué manuellement par l'utilisateur.

n est le nombre de termes d'indexation de la collection de documents.

m est le nombre de documents dans la collection.

La pertinence du document d_i pour la requête Q est définie par des mesures de similarité dans l'espace vectoriel. La fonction de similarité évalue la correspondance entre les deux vecteurs (document et requête) et cherche à retrouver les vecteurs documents qui s'approchent le plus du vecteur requête. La figure I.3 montre un exemple d'une représentation du modèle vectoriel.

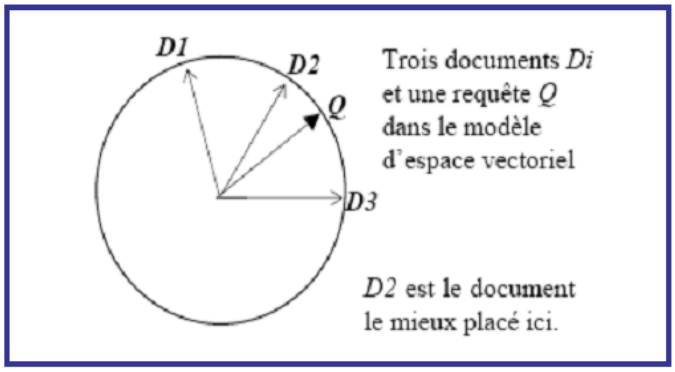


Figure I.3: Exemple d'une représentation du modèle

Une des plus simples mesures de similarité est celle du produit scalaire :

$$RSV(Q, d_j) = \sum_{i=1}^{n} w_{i,Q} \times w_{i,j}$$

Plusieurs autres fonctions de similarité ont été proposées. Nous citons les fonctions les plus répandues : les mesures de Cosinus, Jaccard et Dice.

Mesure de cosinus : $$RSV(Q, d_j) = \frac{\sum_{i=1}^{N} w_{i,Q} \times w_{i,j}}{\sqrt{\sum_{i=1}^{N} w_{i,Q}^2} \times \sqrt{\sum_{i=1}^{N} w_{i,j}^2}}$$

Mesure de Jaccard : $$RSV(Q, d_j) = \frac{\sum_{i=1}^{n} w_{i,Q} \times w_{i,j}}{\sum_{i=1}^{n} w_{i,Q}^2 + \sum_{i=1}^{n} w_{i,j}^2 - \sum_{i=1}^{n} w_{i,Q} \times w_{i,j}}$$

Mesure de Dice : $$RSV(Q, d_j) = \frac{2 \times \sum_{i=1}^{n} w_{i,Q} \times w_{i,j}}{\sum_{i=1}^{n} w_{i,Q}^2 + \sum_{i=1}^{n} w_{i,j}^2}$$

Le modèle vectoriel permet de pallier à l'un des inconvénients majeurs du modèle booléen, en permettant de trier les documents répondant à une requête. Les documents dans le modèle vectoriel, sont en effet restitués dans un ordre décroissant de leur degré de similarité avec la requête. Plus le degré de similarité d'un document est élevé, plus le document ressemble à la requête, et donc pertinent pour l'utilisateur. Cependant, ce modèle ne considère pas les éventuels liens qui peuvent exister entre les termes, chacun des termes est considéré comme indépendant des autres. Le modèle vectoriel généralisé (Generalized Vector Space Model) [Wong et al, 85] permet cependant de résoudre le problème d'indépendance des termes.

I.6.3. Les modèles reposant sur les probabilités :

I.6.3.1. Le modèle probabiliste :

Le premier modèle probabiliste a été proposé par Maron et Kuhns [Maron & al, 60] au début des années 1960. Son principe de base consiste à retrouver des documents qui ont une forte probabilité d'être pertinents, et une faible probabilité d'être non pertinents.

Etant donné une requête utilisateur notée *Q* et un document *d*, le modèle probabiliste tente d'estimer la probabilité que le document *d* soit un document pertinent (respectivement non pertinents). Un document est alors sélectionné si la probabilité qu'il soit pertinent à *Q*, notée *P(d/R)*, est supérieure à la probabilité qu'il soit non pertinent à *Q*, notée *P (d/ NR).* Le score d'appariement entre le document *d* et la requête *Q*, noté *RSV (d,Q),* est donné par [Robertson, 94b]:

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

Il existe plusieurs méthodes pour l'estimation de ces différentes probabilités. La plus connue est celle du modèle BIR(Binary Independance Retrieval), où, on considère que la variable document $d(t_1 = x_1, t_2 = x_2, .. , t_n = x_n)$ est représenté par un ensemble d'événements qui dénotent la présence ($x_i = 1$) ou l'absence ($x_i = 0$) d'un terme dans un document. En supposant que ces événements soient indépendants (d'où l'appellation BIR), les probabilités de pertinence (resp. de non pertinence) d'un document, notées P (d/R) (resp. P(d/NR)), sont données par :

$$P(d/R) = \prod_i P(t_i = x_i/R)$$

$$P(d/NR) = \prod_i P(t_i = x_i/NR)$$

t_i est le $i^{ème}$ terme utilisé pour décrire le document d, et x_i vaut 0 quand le terme est absent, et vaut 1 quand il est présent dans le document. Ces probabilités sont estimées comme suit :

$$P(d/R) = \prod_i P(t_i = x_i/R) = \prod_i P(t_i = 1/R)^{x_i} \times P(t_i = 0/R)^{1-x_i}$$

$$P(d/NR) = \prod_i P(t_i = x_i/NR) = \prod_i P(t_i = 1/NR)^{x_i} \times P(t_i = 0/NR)^{1-x_i}$$

En posant P (t_i = 1/R) par p_i, et P(t_i = 0/NR) par q_i, RSV(d,Q) peut s'écrire, après transformation, comme suit :

$$RSV(d,Q) = \frac{{p_i}^{x_i}\ (1-p_i)^{1-x_i}}{{q_i}^{x_i}\ (1-q_i)^{1-x_i}}$$

Et en passant par la fonction log :

$$RSV(d,Q) = \sum_{i\ ;\ x_i=1} log\ \frac{p_i\ (1-q_i)}{q_i(1-p_i)}$$

Si on suppose que les ensembles R (documents pertinents) et NR (documents non pertinents) soient connus, alors on peut aisément estimer les probabilités p_i et q_i, en utilisant les proportions définies dans le tableau I.1 comme suit [Amirouche, 08]:

$$p_i = \frac{r_i}{n} \quad \text{et} \quad q_i = \frac{R_i - r_i}{N-n}$$

Nombre de documents pertinents contenant t_i $\boldsymbol{r_i}$	Nombre de documents pertinents ne contenant pas t_i $\boldsymbol{n - r_i}$	Nombre de documents pertinents $\boldsymbol{n}$
Nombre de documents non pertinents contenant t_i $\boldsymbol{R_i - r_i}$	Nombre de documents non pertinents ne contenant pas t_i $\boldsymbol{N - R_i - n + r_i}$	Nombre de documents non pertinents $\boldsymbol{N - n}$
Nombre de documents contenant t_i $\boldsymbol{R_i}$	Nombre de documents ne contenant pas t_i $\boldsymbol{N - R_i}$	Nombre d'échantillons $\boldsymbol{N}$

Tableau I.1 : Distribution de probabilités de pertinence des termes d'un corpus

Ainsi la RSV se réduit à:

$$RSV(d,Q) = \sum_{i\ ;\ x_i=1} log\ \frac{r_i\ (N - R - n + r_i)}{(n - r_i)(R_i - r_i)}$$

Un des inconvénients de ce modèle est l'impossibilité d'estimer ses paramètres si des collections de testes ne sont pas disponibles.

I.6.3.2. Le modèle de langue :

Les SRI utilisant les modèles de langages suivent une approche différente des autres modèles. En effet, dans la plupart des modèles, on cherche à comparer une représentation de la requête de l'utilisateur avec une représentation du document recherché, pour évaluer la pertinence de celui-ci. Ici, on part de l'observation que, l'utilisateur crée la requête à partir d'une représentation hypothétique qu'il se fait du document recherché. La requête est donc, inférée par l'utilisateur à partir des documents voulus [Fellag, 06]. Ce modèle considère alors que la pertinence d'un document pour une requête est en rapport avec la probabilité que la requête puisse être générée par le document [Ponte & al., 98] [Boughanem & al., 04].

La probabilité que la requête soit inférée par le document $P(Q/d)$ est estimée par :

$$P(Q/d) = \prod_{i=1}^{n} P(t_i/d)$$

Où n est le nombre de termes dans la requête

t_i est un terme de la requête pour $i=1,2,\ldots,n$

$P(t_i/d)$ peut être estimé en se basant sur l'estimation maximale de vraisemblance (maximum likelihood estimation). Elle est donnée par :

$$P(t_i/d) = \frac{tf(t_i/d)}{\sum_t tf(t/d)}$$

Ou : $tf(t_i/d)$ est la fréquence du terme t_i dans le document d.

Dans ce type d'estimation lorsqu'un terme de la requête est absent du document, on a systématiquement RSV(d,Q) = 0. Afin de pallier cet inconvénient, des technique de lissage ont été développées comme : le lissage de Laplace, le lissage de Good-Turing ou le lissage de Backoff [Boughanem & al., 04]. Elles consistent à assigner des probabilités non nulles aux termes, qui n'apparaissent pas dans les documents.

I.7. Evaluation des systèmes de recherche d'information :

L'évaluation d'un SRI constitue une étape importante dans l'élaboration d'un modèle de RI, puisqu'elle permet de paramétrer le modèle, et de fournir des éléments de comparaison entre modèles. L'évaluation nécessite alors, la définition d'un ensemble de mesures et méthodes d'évaluation et bases de test, assurant l'objectivité de l'évaluation [Fellag, 06].

L'évaluation d'un SRI, peut être appréhendée selon deux aspects : un *aspect efficience* et un *aspect efficacité*. L'aspect efficience dépend de l'évaluation cognitive de l'utilisateur, tels que la facilité d'utilisation du système, rapidité d'accès, temps de réponse à une requête, présentation des résultats, …etc. Alors que l'aspect efficacité concerne la capacité du système, à sélectionner le maximum de documents pertinents et un minimum de documents non pertinents. Nous nous intéressons dans cette section, à présenter l'aspect efficacité.

Pour mesurer la qualité des résultats retrouvés pas le SRI, différentes métriques sont utilisées, qui sont définies dans ce qui suit :

I.7.1.Les mesures de Rappel/Précision :

Les mesures de rappel et précision permettent d'évaluer la capacité d'un SRI à répondre aux deux objectifs principaux qui sont : (1) retrouver tous les documents pertinents et (2) rejeter tous les documents non pertinents. Afin de représenter ces deux mesures, nous introduisons (Figure I.4) le partitionnement de l'ensemble des documents restitués (noté B) par le SRI en deux sous ensembles : un sous ensemble de documents pertinents et un sous ensemble de document non pertinents.

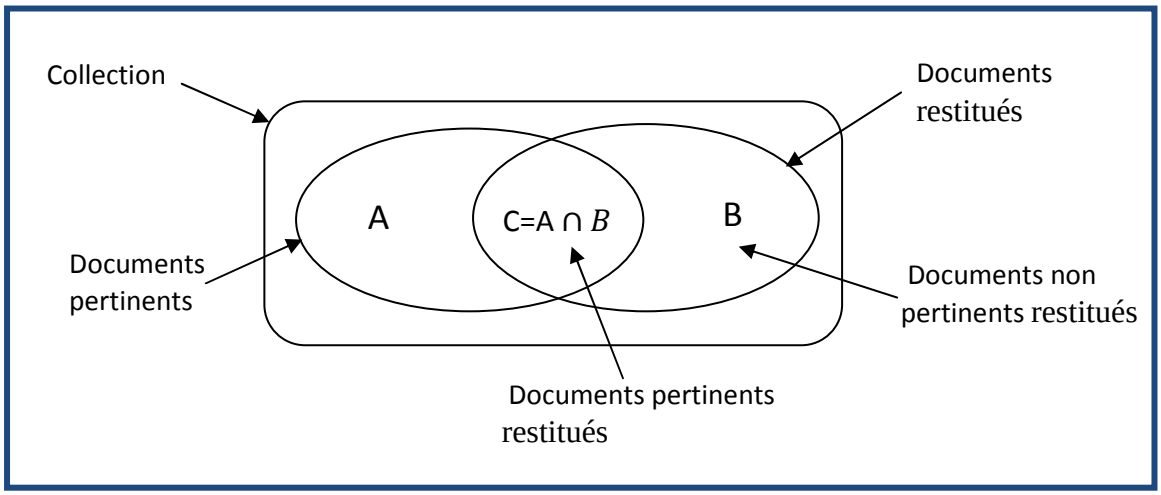


Figure I.4 : Partition de la collection pour une requête

- **Taux de précision** : la précision mesure la capacité du système à rejeter tous les documents non pertinents à une requête. Le taux de précision est la proportion de documents pertinents restitués par le système relativement à l'ensemble des documents restitués par le système. Il est exprimé par :

$$Taux\ de\ précision = \frac{|C|}{|B|}$$

- **Taux de rappel** : le rappel mesure la capacité du système à retrouver tous les documents pertinents répondant à une requête. Le taux de rappel est la proportion de documents pertinents restitués par le système relativement à l'ensemble des documents pertinents contenus dans la collection. Il est exprimé par :

$$Taux\ de\ rappel = \frac{|C|}{|A|}$$

Dans le cas d'un système idéal, le taux de précision est égal au taux de rappel, c'est-à-dire, tous les documents pertinents sont restitués (C = A ∩ B = A = B).

Des mesures complémentaires au rappel et à la précision, respectivement le silence et le bruit ont été définies comme suit :

- Le silence = $\frac{\text{nombre de documents pertinents non restitués}}{\text{nombre de documents pertinents}}$

- Le bruit = $\frac{\text{nombre de documents non pertinents restitués}}{\text{nombre total de documents restitués}}$

I.7.2.Mesures combinées de rappel et précision :

- **Mesure harmonique :**

La mesure harmonique H est une fonction qui combine les valeurs de rappel et de précision en une seule valeur incluse dans l'intervalle [0,1] [Shaw et al, 97].

$$H(j) = \frac{2}{\frac{1}{R(j)} + \frac{1}{P(j)}}$$

Où : *R (j)* et *P (j)* représentent respectivement le rappel et la précision du j ème document renvoyé par le système.

Cette mesure égale à 0 lorsqu'aucun document pertinent n'est restitué et égale à 1 lorsque tous les documents restitués sont pertinents. On peut constater que la fonction *H* prend des valeurs élevées quand les valeurs de rappel et de précision sont élevées.

- **Mesure d'évaluation « E »**

Cette mesure a été proposée par Van Rijsbergen [Rijsbergen, 79]. Son but est de permettre à l'utilisateur de spécifier laquelle des valeurs de précision et de rappel est plus intéressante. La mesure est ainsi définie par :

$$E(j) = 1 - \frac{1 + b^2}{\frac{b^2}{R(j)} + \frac{1}{P(j)}}$$

La variable *b* est un paramètre de l'utilisateur qui permet de spécifier l'importance du rappel et précision. Si *b=1*, *E(j)* va prendre la valeur du complément de la mesure harmonique *H(j).*

Si *(b < 1),* l'utilisateur privilège le rappel et si *(b > 1),* il privilège la précision.

I.7.3. collections de test :

La collection de test est constituée d'une collection de documents donnée, d'un ensemble de requêtes avec les jugements de pertinence associés. Ces dernières vont permettre d'évaluer la qualité des réponses fournies par les systèmes à évaluer. Pour évaluer un SRI, il suffira alors de lui soumettre les requêtes, et de comparer les réponses qu'il fournit par rapport aux réponses-types (jugements de pertinence). En mesurant l'écart entre la réponse du système et la réponse-type, on obtiendra une mesure de qualité sur les performances du système documentaire.

I.7.4. Les compagnes d'évaluation :

L'histoire de l'évaluation des SRI a connu la naissance de grandes campagnes d'évaluation destinées à stimuler et favoriser l'émergence de nouveaux SRI. Ces compagnes sont menées régulièrement depuis plusieurs années, les plus connues sont :

- Le programme américain TREC[2] (Text Retrieval Evaluation Conference) entamé en 1992, initié par le NIST (National Institute of Standards and Technology) aux USA. Il fournit une plate-forme comportant des collections de tests, des tâches spécifiques et des protocoles d'évaluation pour chaque tâche, pour l'évaluation et la comparaison d'expérimentations sur des collections volumineuses de textes. Les collections TREC représentent aujourd'hui un référentiel incontournable en RI.
- Le programme CLEF[3] (Cross Language Evaluation Forum). CLEF était à l'origine, la tâche multilingue de TREC avant de devenir depuis 2000, un programme à part entière supportant la majorité des langues européennes.

[2] TREC : http ://trec.nist.gov
[3] CLEF http ://www.clef-campaign.org/ accédés le 20-05-2009

I.8. Conclusion :

Nous avons présenté dans ce chapitre les concepts de base de la recherche d'information traditionnelle à travers l'étude du processus de RI et des modèles de recherche, Enfin nous avons décrit l'évaluation des systèmes de recherche à travers l'introduction des mesures d'évaluation. Le but est de montrer que les systèmes de recherche d'information sont conçus à l'origine pour retrouver les documents pertinents traitant une requête donnée.

La majorité des SRI présentés (basés sur la théorie des ensembles, algébriques et probabilistes) se contentent de chercher les documents qui contiennent les mêmes mots que ceux de la requête. Ceci est évidemment insuffisant dans la mesure où l'utilisateur du système ne connaît pas à priori le vocabulaire que l'auteur a utilisé dans le document.

Pour remédier cette lacune, de nouvelles approches d'indexation appelées « approches sémantiques» ont été développées. Elles utilisent principalement des techniques de désambiguïsation des sens des mots. Le chapitre suivant est dédié à ces nouvelles approches de RI.

Chapitre II

Indexation sémantique en RI

II.1. Introduction :

Ces dernières années, beaucoup de travaux ont souligné l'insuffisance de l'indexation classique [Robertson & al., 97] [Woods, 97] [Khan, 00] [Guarino & al., 01], qui se base sur des mots clés pour représenter les entités textuelles (documents et requêtes) et ne prend pas en considération les liens sémantiques qui peuvent exister entre eux. Pour pallier les problèmes liés à l'indexation classique, des travaux tentant d'incorporer l'information sémantique dans le processus de RI sont apparus à travers **l'indexation sémantique**. L'indexation sémantique s'intéresse principalement à la représentation des documents et requêtes par les sens des mots qu'ils contiennent plutôt que par les mots eux mêmes.

L'objectif du présent chapitre est de présenter les principales approches d'indexation sémantique. Nous commençons par présenter la problématique de l'indexation classique basée mots clés dans la section (II.2). Une présentation des différentes ressources sémantiques dans la section (II.3). La section (II.4) décrit l'approche d'indexation conceptuelle. La section (II.5) est dédiée à la présentation des approches d'indexation sémantique basées sur la désambiguïsation.

II.2. Problématique :

Dans les SRI classiques, documents et requêtes sont représentés comme des listes de mots clés, généralement pondérés. L'appariement document-requête est lexical et se base sur la présence ou l'absence d'un mot de la requête dans le document. Ces systèmes posent problème vu la complexité de la langue. La variation linguistique, caractéristique du langage naturel et présente dans les requêtes utilisateurs et dans les documents, dégrade l'efficacité des systèmes de RI en termes de précision et de rappel [Prié, 00]. D'un côté, l'ambiguïté lexicale (ou variation sémantique et syntaxique) touche la précision. Et d'un autre côté, la disparité des mots (ou variation lexicale) touche le rappel.

Ainsi, dans une indexation classique, où les documents et requêtes sont représentés par des mots clés issus de leurs contenus, on se retrouve confrontés à deux problèmes : l'ambiguïté des mots et leur disparité [Amirouche, 08] :

1) L'ambiguïté des mots (ambiguïté lexicale) :

Se rapporte à des mots lexicalement identiques et portant des sens différents. En effet, deux mots peuvent s'écrire exactement de la même marnière sans pour autant avoir le même sens.

L'ambiguïté lexicale est généralement divisée en deux types [Krovetz, 97; Krovetz et al, 92] : l'ambiguïté sémantique et l'ambiguïté syntaxique.

- *L'ambiguïté sémantique* : se rapporte à des différences dans la signification, et est décomposée en homonymie (sens très différent par exemple le mot « jaguar» peut avoir le sens « animal » ou « constructeur automobile ») et polysémie (sens plus proches, distinction plus difficile).
- L'ambiguïté syntaxique : se rapporte à des différences dans la catégorie syntaxique. Par exemple, le nom «mouse» est différent du verbe «mouse».

Le problème d'ambiguïté implique que des documents non pertinents, contenant les mêmes mots que la requête sont retrouvés. Par exemple (figure II.1), dans une recherche sur Google, à l'aide du mot clé avocat (métier) des documents contenant le mot avocat sont retournés mais ils ne traitent pas tous le métier d'avocat.

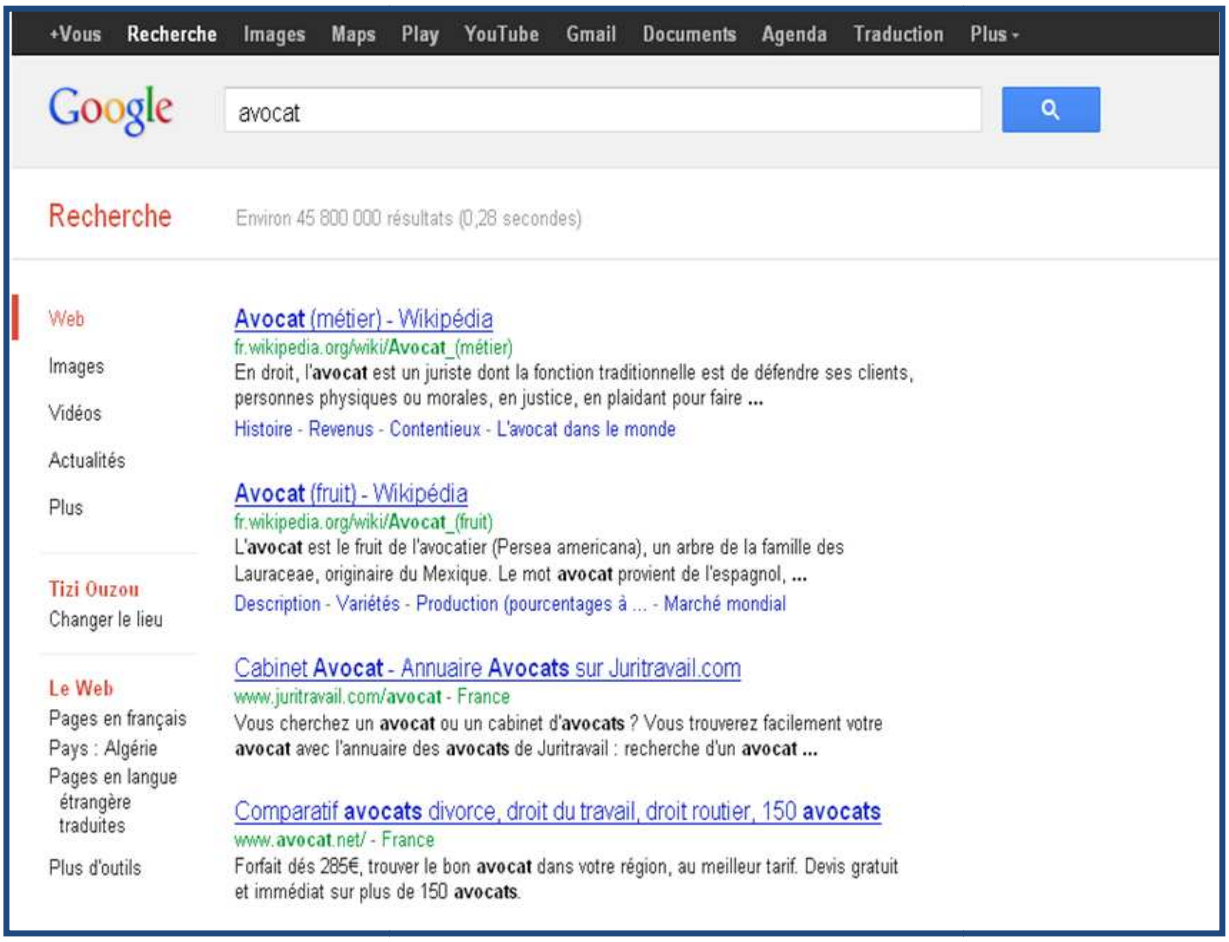


Figure II.1 : Résultat de la recherche du mot « avocat » sur Google[1]

2) La disparité des mots (Word Mismatch) **:**

A l'inverse de l'ambiguïté lexicale, la disparité des mots se réfère à des mots lexicalement différents mais portant un même sens (synonymie). Ceci implique que des documents, pourtant pertinents, ne partagent pas de mots avec la requête, ne sont pas retrouvés. Par exemple : automobile, voiture, véhicule… Cette situation pose un problème à la RI quand l'utilisateur soumet au système une requête contenant le mot «véhicule », mais que le système ne trouve pas les documents qui contiennent le mot «voiture ».

[1] www.google.com

Les travaux du domaine ont d'abord adressé ces problèmes séparément en apportant des solutions spécifiques, tel que l'utilisation d'expressions ou de mots composés pour résoudre le problème lié à l'ambiguïté lexicale, et l'expansion de requête à l'aide de mots synonymes d'un thésaurus, pour résoudre la disparité des mots [Salton et al.., 83]. Puis une solution globale s'est dégagée pour remédier à ces deux problèmes, à travers **l'indexation sémantique**.

L'indexation sémantique s'intéresse à deux principaux points : d'abord retrouver le sens correct de chaque mot dans le document (respectivement de la requête), ensuite représenter ce document (respectivement cette requête) [Amirouche ,08] :

- Pour retrouver le sens correct d'un mot ambigu dans son contexte d'utilisation dans le document (ou la requête), l'indexation sémantique se base sur des techniques dites de désambiguïsation des sens des mots ou WSD (Word Sense Disambiguation). Il existe deux principales approches de désambiguïsation du sens: les approches endogènes s'appuyant sur la collection pour désambiguïser un mot, alors que les approches exogènes s'appuient sur des ressources externes (exemple WordNet) pour désambiguïser un mot [Audibert, 03].
- Pour la représentation sémantique des documents et requêtes, l'indexation sémantique s'intéresse à leur représentation en se basant sur les sens des mots qu'ils contiennent

  Dans ce contexte, deux principales approches de représentation existent:

  - *La représentation basée sur les sens* : les mots sont désambiguïsés et ce sont les sens correspondants calculés, qui sont finalement utilisés comme termes d'indexation.
  - *La représentation combinée mots-clés/sens* : les sens sont utilisés conjointement avec les mots clés qu'ils représentent. Un terme d'indexation est alors représenté par le couple (mot-clé, sens associé).

II.3. Les différentes ressources sémantiques :

Avant de décrire les différentes ressources sémantiques qui existes nous définissons d'abord la notion *de concept en RI.*

Selon le dictionnaire de l'académie française, "Le concept regroupe les objets qu'il définit en une même catégorie appelée « classe » ". De façon générale, le terme concept est souvent utilisé comme se référant à toute notion, de l'idée au lexème, en passant par l'entité et la catégorie [Baziz, 05].

Les concepts sont le plus souvent prédéfinis et pré-ordonnés dans des structures conceptuelles telles les ontologies. Ces structures sont souvent construites manuellement par des spécialistes du domaine qu'elles couvrent. Un concept, correspond à un nœud de cette structure tel que définis dans la ressource WordNet (dans WordNet un nœud est appelé aussi Sysnet).

II.3.1. Les réseaux sémantiques :

Les réseaux sémantiques sont très utilisés dans les travaux sur la compréhension et le traitement des langages. Quillian [Quillian, 68] définit un réseau sémantique comme étant «un format de représentation permettant de mémoriser le sens des mots, pour rendre possible leur utilisation à la manière de l'être humain». L'idée de base est que la signification d'un mot dépend des autres mots qui co-occurrent avec lui. Ainsi, la signification d'un concept est liée au réseau sémantique auquel il fait partie et de ses relations avec les autres concepts du réseau. Dans un réseau sémantique, un concept est représenté par un nœud. Chaque relation entre deux concepts est représentée par un arc étiqueté qui relie les nœuds associés (concepts). un réseau sémantique est assimilé à un graphe orienté G=[S, R] dont les sommets S sont les concepts et les relations R sont les relations sémantiques entre les concepts (voir figure II.2).

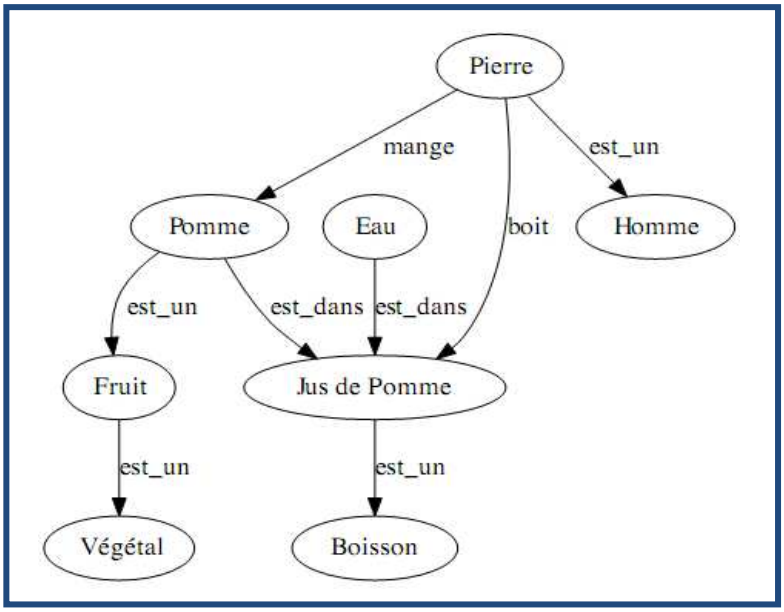


Figure II.2 : Exemple de Représentation de Réseau Sémantique.

II.3.2. Les taxonomies :

La taxonomie est un réseau sémantique où l'unique relation est une relation hiérarchique, transitive et non réflexive. Souvent, la relation hiérarchique «est_un» (en anglais, «is_a») est utilisée et on parle de classification. Les concepts sont appelés des taxons [Chahine, 11].

II.3.3. Le thésaurus :

Un thésaurus est un vocabulaire contrôlé. Il rassemble un ensemble de termes structurés choisis pour leur capacité à décrire un domaine. Ces termes sont utilisés pour décrire d'une manière précise le contenu des documents. Ils sont sélectionnés et normalisés pour l'indexation et le classement des documents. Dans un thésaurus, les termes dénotent les concepts d'un domaine particulier. Ces concepts sont reliés entre eux par des relations sémantiques : liens hiérarchiques (généralisation et spécialisation), synonymie, et relation d'association (relations de proximité sémantique, proche-de, relié-à, etc.)[Zargayouna, 05]. Chaque concept possède un terme descripteur qui permet de le nommer facilement.

II.3.4. Les ontologies :

La définition la plus citée présente une ontologie comme étant « une spécification explicite et formelle d'une conceptualisation partagée» [Gruber, 93]. 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.

De façon générale, le mot "ontologie" est utilisé abusivement pour renvoyer à des structures lexicales et sémantiques variées. Par exemple, les dictionnaires, les thesaurus de la communauté de l'informatique linguistique, WordNet, MeSH, etc.

Une ontologie peut être utilisée par un SRI, pour accéder à des ressources (des documents, des informations). L'apport de l'ontologie peut être appréhendé (Figure II.3) à trois niveaux [Aussenac, 02]:

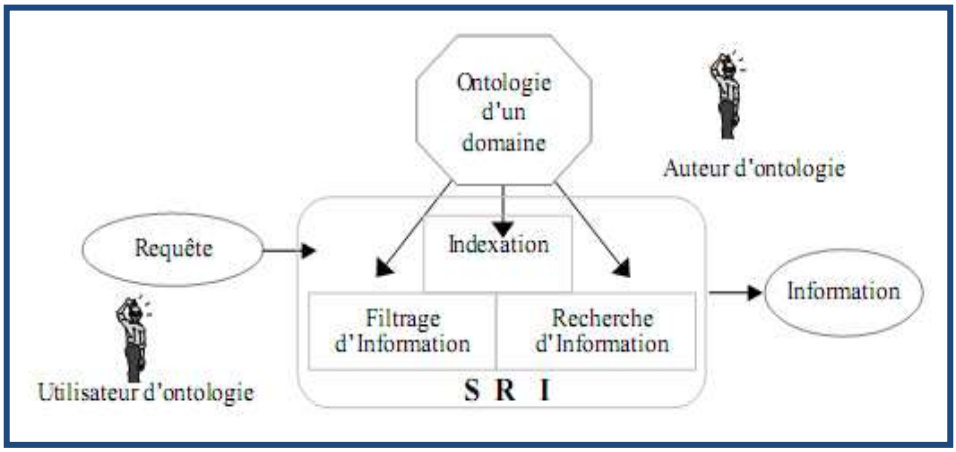


Figure II.3 : L'ontologie greffée au processus de recherche d'information

- Au niveau du processus d'indexation : en s'associant à des techniques de traitement automatique du langage naturel (processus de désambiguïsation), les termes du document (ou requête) seront reliés à des concepts de l'ontologie. Si cette étape s'est passée convenablement (processus de désambiguïsation performant), la recherche sera plus aisée par la suite.
- Au niveau du processus de filtrage d'information.
- Au niveau de la reformulation des requêtes, pour améliorer les requêtes utilisateurs.

II.3.5. WordNet :

WordNet est une base de données lexicale, développée depuis 1985 à l'université de Princeton par une équipe de psycholinguistes et de linguistes. Son but est de répertorier, classifier et mettre en relation, de diverses manières, le contenu sémantique et lexical de la langue anglaise[1]. Wordnet couvre la grande majorité des noms, verbes, adjectifs et adverbes de la langue anglaise, qu'elle structure en un réseau de nœuds et de liens.

- **Les nœuds** sont constitués par des groupes de termes synonymes appelés synsets (provenant de synonym set). Un synset représente un concept (ou sens), Il comporte la liste des synonymes exprimant un même concept, la définition du concept (gloss) et éventuellement des exemples d'usage.
- **Les liens** représentent les relations entre les synsets, par exemple la relation is-a (hyperonyme/ hyponyme) X est un hyponyme de Y si X est un type de (is a) Y. (Voir Annexe 1 pour plus de détails).

II.4. L'indexation conceptuelle :

L'indexation conceptuelle se base sur des entités conceptuelles (concepts) tirés de ressources (ontologies, taxonomies…etc) pour indexer les documents et requêtes. Elle peut être vue comme une généralisation de l'indexation sémantique, dans la mesure où les concepts aussi véhiculent des sens [Mallak, 11].

Woods [Woods, 97] propose une approche d'indexation conceptuelle qui consiste à extraire automatiquement des mots et des expressions de textes à l'aide d'une ontologie, puis les organise en une structure hiérarchique (taxonomie, réseau sémantique). Le système est capable de transformer automatiquement les syntagmes pour qu'ils correspondent à des concepts du réseau sémantique.

Exemple 1 : Dans la phrase «la maison est repeinte en rose», le syntagme «repeinte en rose» va être associé au concept présent dans le réseau «changement de couleur» et sera utilisé pour l'indexation. Il s'agit bien d'indexation conceptuelle, car ce ne sont plus les termes désambiguïsés qui indexent le document et la requête, mais bien les concepts du réseau [Chahine, 11].

Exemple 2 : Etant donnée l'information que voiture est un genre-de automobile et que lavage est un genre-de Nettoyage, le système peut automatiquement déterminer que lavage de voiture est un genre-de nettoyage d'automobile [Amirouche, 08] (voir Figure II.4).

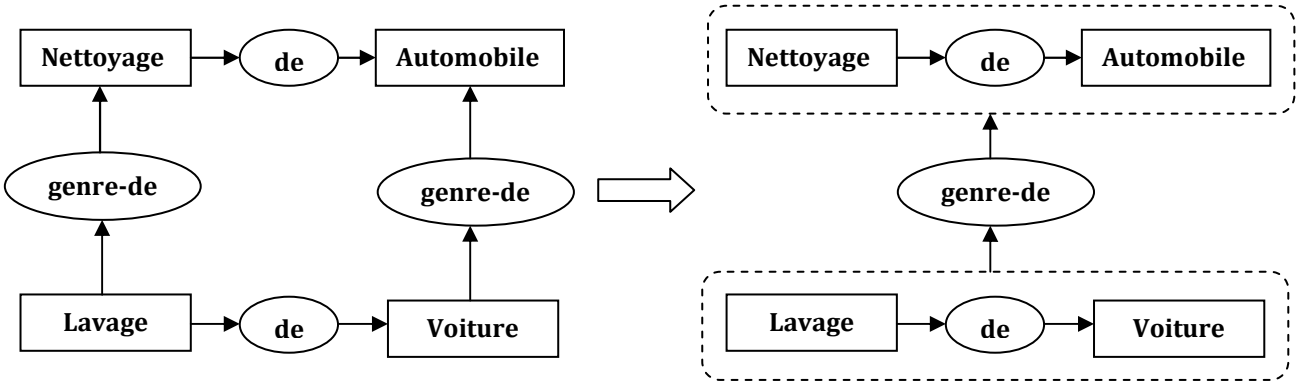


Figure II.4 : Exemple de taxonomie conceptuelle.

II.5. L'indexation sémantique basée sur la désambiguïsation :

L'indexation sémantique a depuis longtemps été pressentie comme un moyen qui permettrait d'améliorer les performances de la RI. Elle se base sur les sens des mots pour indexer les documents et les requêtes. Pour retrouver les sens corrects des mots dans un document (respectivement dans une requête), l'indexation sémantique a recours aux techniques de désambiguïsation automatique des sens des mots.

II.5.1. Les approches d'indexation sémantique :

Nous pouvons distinguer deux types d'approches d'indexation sémantique [Amirouche, 08] : Les approches d'indexation sémantique basées sur la WSD endogène (WSD basée sur le corpus) et les approches d'indexation sémantique basées sur la WSD exogène (WSD basée sur les ressources linguistiques externes).

II.5.1.1. Les approches d'Indexation sémantique basée sur la désambiguïsation endogène :

Ces approches ont recours aux corpus pour la désambiguïsation des mots des documents (et des requêtes). Le principe général consiste à désambiguïser un mot ambigu à l'aide de calculs statistiques sur les instances de ce mot dans le corpus et les différents contextes dans lequel il est employé. Les documents et requêtes sont ainsi indexés en utilisant les sens retrouvés. L'approche de Weiss [Weiss, 73] est un exemple de ce genre d'approches.

- **Weiss [Weiss, 73]** :

Weiss est l'un des premiers à s'intéresser à l'apprentissage automatique de règles de désambiguïsation à partir de corpus. Il a manuellement construit deux types de règles permettant la désambiguïsation. Il s'agit de règles générales de contexte et de règles de modèle.

- Une règle générale de contexte établit qu'une occurrence de mot ambigu a un certain sens si un mot particulier apparaît *près* de cette occurrence du mot ambigu. Par exemple, si le mot « print » apparaît près du mot « type » alors son sens est probablement lié à l'impression.
- Une règle de modèle établit qu'une occurrence d'un mot ambigu a un certain sens si un mot particulier apparaît *à un endroit spécifique* relatif à cette occurrence. Par exemple, si le mot « of » apparaît juste après le mot « type », alors le sens de cette occurrence est probablement « variety of ».

Weiss a constaté que les règles de modèle étaient plus efficaces pour déterminer le sens que les règles de contexte. L'exactitude de son désambiguïseur était de l'ordre de 90%, mais le nombre de tests était très réduit.

II.5.1.2. Les approches d'Indexation sémantique basée sur la désambiguïsation exogène :

Contrairement aux approches précédentes, ces approches s'appuient sur une ressource linguistique externe (exemple WordNet) pour désambiguïser un mot ambigu. Le principe de base des approches d'indexation sémantique basées sur la WSD exogène est décrit comme suit :

1) Extraire l'ensemble des termes descripteurs du document, par une approche classique d'indexation.
2) Désambiguïser chaque terme descripteur. Pour ce faire, il s'agit de retrouver tous les sens du terme à partir de la ressource externe puis à associer un score à chacun de ses sens sur la base de : Soit la distance sémantique de ce sens aux différents sens associés aux autres termes dans le document [Baziz, 05] ou dans le contexte global [Amirouche, 11]. Soit le degré de recouvrement entre d'une part, le contexte local de ce mot et d'autre part le voisinage (hood) [Voorhees, 93] de ce sens ou la définition de ce sens (ensemble de synonymes) [Katz et al., 98] dans la ressource linguistique utilisée. Le sens qui maximise le score est alors retenu comme sens adéquat du terme d'indexation.
3) La représentation des textes indexés se fait soit à partir des seuls sens (ou concepts) identifiés lors de l'étape de désambiguïsation, soit à partir d'une combinaison des mots-clés et sens corrects associés.

Nous allons citer dans ce qui suit quelques approches basées sur la WSD exogène :

- **Voorhees [Voorhees, 93]** :

  Dans l'approche d'indexation de Voorhees, les textes à indexer sont analysés phrase par phrase. La phrase représente alors le contexte local de chacun de ses mots. Chaque mot non vide rencontré dans la phrase, est projeté sur l'ontologie WordNet afin de retrouver tous ses sens (synsets) possibles (un mot ambigu correspond à plusieurs synsets dans Wordnet). Par la suite, pour déterminer le synset (concept) adéquat pour un mot ambigu dans une phrase, chaque synset de ce mot est classé en se basant sur le nombre de mots co-occurrents entre un *voisinage* (appelé aussi hood) de ce synset et le contexte local du mot ambigu correspondant. Le synset le mieux classé est alors considéré comme sens adéquat du mot ambigu.

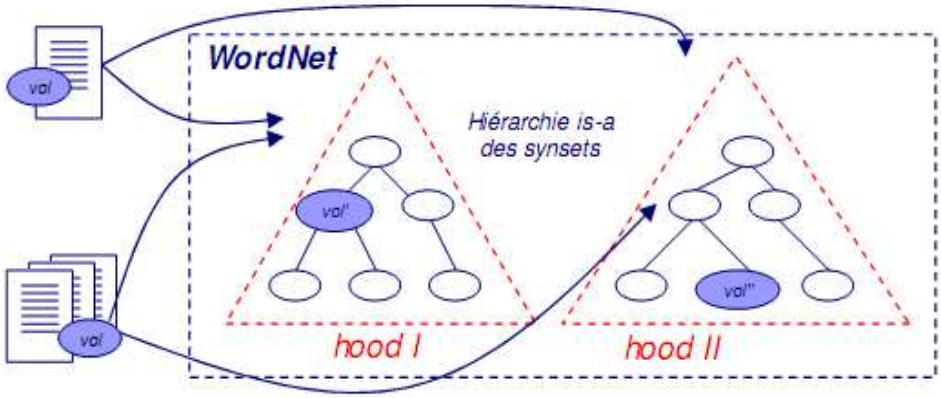


Figure II.5 : schéma de l'utilisation des hood dans WordNet

Voorhees définit le voisinage (hood) d'un synset S comme suit :

« *En considérant l'ensemble des synsets comme les sommets et les relations d'Hyperonymie et d'Hyponymie dans WordNet comme les arcs dirigés d'un graphe, le voisinage d'un synset donné S, serait alors le plus grand sous-graphe connexe qui contient S, qui contient seulement les descendants d'un ancêtre de S et ne contient aucun synset ayant un descendant qui inclut une autre instance d'un membre (un mot) de S* ».

Par exemple, à partir du fragment de la structure de WordNet donnée par la figure II.6, le voisinage du premier sens de "*house*" inclurait les termes: *housing, lodging, apartment, flat, cabin, bungalow, cottage, gatehouse.*

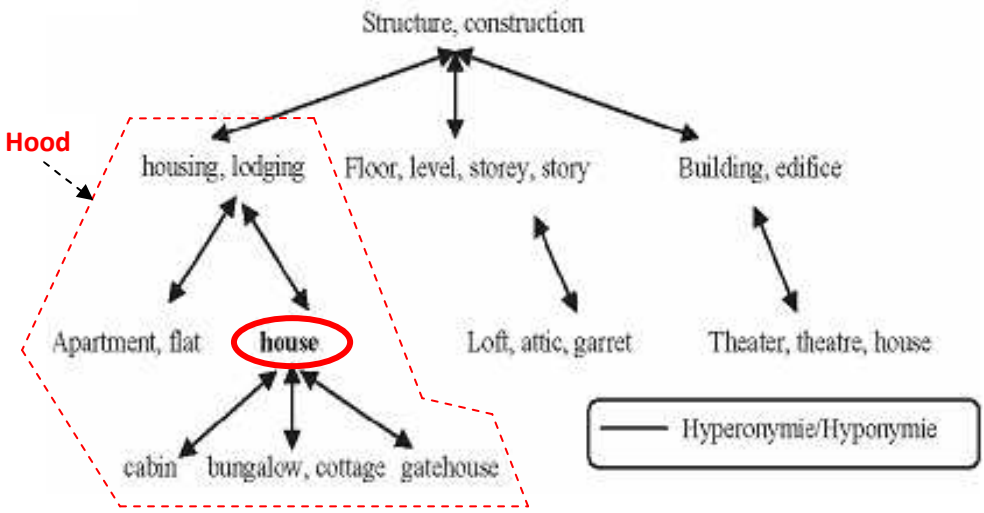


Figure II.6 : Exemple de hood (voisinage de mot) selon Voorhees.

Les termes "structure" et "construction" (situés en haut de la hiérarchie), ne seraient pas inclus puisque un des descendants de leur synset contient un autre sens du terme "house".

Voorhees a soumis cette approche pour expérimentation sur une collection de test désambiguïsée manuellement (les requêtes et les documents) et l'a comparée aux performances du même système sur la même collection non désambiguïsée.

Elle a constaté une dégradation dans la performance de son système pour la plupart des requêtes. Elle suppose que l'exploitation des deux relations de WordNet n'est pas suffisante pour retrouver les sens corrects des mots.

- **Katz et al [Katz et al, 98] :**

La méthode de Katz et al est similaire à celle de Voorhees. Elle est basée sur WordNet et sur le contexte local pour désambiguïser les mots dans le texte. Pour chaque mot non vide rencontré, on recherche dans WordNet tous les synsets qui lui correspondent. Le contexte local d'un mot est défini comme étant la liste ordonnée des mots démarrant du mot utile le plus proche du voisinage gauche ou droit jusqu'au mot cible, par exemple dans le texte "... *the jury had been* ***charged*** *to investigate reports of irregularities in the primary*...", le contexte local droit de charged est "*X to investigate*". Son contexte local gauche est "*the jury has been X*".

L'hypothèse de Katz et al est que des mots utilisés dans le même contexte local (appelés sélecteurs), ont souvent des sens proches. Les sélecteurs des mots d'entrée sont extraits des contextes locaux gauche et droit, puis l'ensemble S de tous les sélecteurs obtenus est comparé avec les synsets de WordNet. Le synset qui a le plus de mots en commun avec l'ensemble S est sélectionné comme sens adéquat du mot cible.

La figure II.7 illustre le processus d'extraction de sélecteurs du mot "charged" dans une phrase donnée. La désambiguïsation du mot "charged" se fait dans deux contextes différents. En comparant les sélecteurs des mots d'entrée avec les synsets de WordNet, le sens de "charged" est associé au sens 3 de WordNet ("charge, bill") dans le premier contexte et au sens 4 de WordNet ("appoint, charge") dans le deuxième contexte.

Les fréquences globales des Sélecteurs sont alors comme suit:

SELECTOR	FREQUENCY
Vessel	2
Equipped	1
Billed	1
Charged	1
…□	…□

SELECTOR	FREQUENCY
Appointed	52
Assigned	28
Established	20
Hired	16
…□	…□

Quelques sens de "charge" comme ils apparaissent dans WordNet:

Sens 1	Sens 2	Sens 3	Sens 4	Sens 5	…
Charge, brear down	Charge, accuse	Charge, bill	Appoint, charge	charge	…

Figure II.7 : Désambiguïsation de termes selon l'algorithme de Katz

Katz et al ont intégré leur algorithme de désambiguïsation au processus de recherche d'information SMART [Salton, 71]. Leur conclusion est que leur algorithme n'améliore pas les performances du système de RI.

- **Mihalcea et Moldovan [Mihalcea & al., 00] :**

  La méthode de Mihalcea et Moldovan se base sur l'information contextuelle, et sur l'identification des sens des mots à partir de WordNet. Un nouveau mot est désambiguïsé en tenant compte de sa relation avec les mots du corpus qui sont déjà désambiguïsés. Ce processus itératif leur permet d'identifier dans le corpus d'origine les mots qui peuvent être désambiguïsées avec une grande précision. Ils arrivent ainsi à désambiguïser 55% des mots (noms et verbes) avec une précision de 92%. La collection de test qu'ils ont utilisée est Cranfield, composée de 1400 documents SGML du domaine de l'aérodynamique.

- **Baziz [Baziz, 05] :**

  L'approche d'indexation de Baziz, comprend deux étapes principales : La première consiste à identifier les différents termes d'indexation du document (respectivement de la requête) en se basant sur WordNet. Pour cela, avant d'éliminer les mots vides du document (respectivement de la requête) la méthode tente de détecter les termes formés par des mots uniques ou des groupes de mots qui correspondent à des entrées dans WordNet. La deuxième étape consiste à désambiguïser les termes identifiés. Ces derniers sont d'abord projetés sur l'ontologie WordNet afin de retrouver tous les concepts (ou sens) correspondants dans l'ontologie. Si un terme d'indexation est associé à plus d'un concept (sens) dans WordNet alors ce terme est ambigu, il faut le désambiguïser. Pour ce faire, On assigne un score basé sur une mesure de similarité pour chacun de ses sens possibles (appelés concepts candidats). Le score d'un concept candidat est obtenu en sommant les valeurs de similarité qu'il a avec les autres concepts candidats (correspondant aux différents sens des autres termes du document). Le concept candidat ayant le plus haut score est alors retenu comme sens adéquat du terme d'indexation.

Baziz a évalué son approche d'indexation sémantique avec la collection de test MuchMore et avec la campagne CLEF 2004. Les résultats rapportés montrent que l'utilisation des sens (concepts de WordNet) seuls pour représenter les documents ne permet pas d'améliorer les résultats comparativement à la méthode classique basée sur les mots clés. Néanmoins, la combinaison de l'indexation classique et de l'indexation sémantique apporte une nette amélioration de la précision.

- **Amirouche [Amirouche, 11] :**

Dans une approche similaire à celle de Baziz, Amirouche extrait d'abord l'ensemble des termes d'indexation en se basant sur WordNet. A l'issue de cette étape, trois listes sont construites : la liste des collocations, la liste des mots simples ayant des entrées correspondantes dans WordNet, et la liste des mots simples n'ayant pas d'entrées dans WordNet (dits des mots orphelins). Puis chaque mot simple non vide identifié dans WordNet est projeté sur l'ontologie WordNet afin de retrouver tous ses sens (concepts) possibles. Si le mot simple possède plusieurs sens alors il faut le désambiguïser.

L'hypothèse de Amirouche est que parmi les différents sens possibles d'un terme donné, le plus adéquat est celui qui a le plus de liens avec les autres sens (concepts) des mots du contexte global de ce terme. Le contexte global d'un mot représente toutes les phrases qui contiennent ce mot dans le document. L'approche de désambiguïsation consiste alors à associer à chaque sens d'un mot non vide, un score basé sur sa proximité sémantique avec les autres sens associés aux mots de son contexte global. Le concept ayant le plus grand score est alors retenu comme sens adéquat du mot.

Amirouche a soumis cette approche pour expérimentation sur la collection de test TIME et a comparé les résultats obtenus à une approche d'indexation classique en utilisant la même collection de test. Les résultats obtenus montrent que l'approche d'indexation sémantique améliore la précision par rapport à une indexation classique.

II.6. Conclusion :

Nous avons consacré ce chapitre à l'état de l'art sur l'indexation sémantique en RI.
Nous avons mis l'accent sur l'intérêt de l'indexation sémantique comme solution au problème de la variation lexicale et sémantique en RI. L'indexation sémantique se base sur les sens des mots pour représenter les documents et les requêtes et se fonde sur des techniques de désambiguïsation des sens des mots (WSD).
Nous avons passé en revue les différentes approches d'indexation sémantique. Bien que les résultats des travaux étudiés divergent sur certains points, ils convergent tous vers une même conclusion, à savoir : la nécessité d'une désambiguïsation. En particulier, dans [Mihalcea & al., 00], [Baziz , 05], [Amirouche, 11] il a été montré que l'indexation par des synsets (concepts) de WordNet, peut réellement améliorer l'efficacité de la RI.

Le chapitre suivant est dédié à la présentation de notre approche d'indexation sémantique et de son intégration à la plateforme de RI Terrier.

Chapitre III

Notre approche d'indexation sémantique

III.1. Introduction :

Après avoir présenté un état de l'art sur les différents travaux liés à l'indexation sémantique, nous avons montré que l'indexation sémantique a pour objectif de représenter les documents et requêtes par les sens des mots (ou les concepts) plutôt que par les mots d'indexation eux même. L'intérêt d'une telle approche est de lever l'ambiguïté des mots et de résoudre le problème de disparité des termes qui existe dans une approche d'indexation classique.

L'objectif du présent chapitre est de présenter notre approche d'indexation sémantique (cette approche a été proposée par [Amirouche, 11]). En effet, elle se base sur WordNet pour l'extraction des différents termes d'index, et leur désambiguïsation. Cette approche est ensuite intégrée à la plate forme de RI Terrier[1], constituant ainsi une extension de la plate forme à la prise en charge de la RI sémantique.

Ce chapitre est organisé autour de deux principales sections : la première est dédiée à la présentation de notre approche d'indexation sémantique, la seconde est consacrée à l'intégration de cette approche dans la plate forme Terrier.

[1] http://terrier.org/docs

III.2. Description de l'approche:

III.2.1. Définitions et notations :

- *Une collocation* désigne une association habituelle De deux ou plusieurs mots reliés par des soulignés pour constituer un mot composé existant dans la langue.
- *Le contexte local* d'un mot désigne la phrase qui contient le mot. On note le contexte local d'un mot m_i par CL_i.
- *Le contexte global* d'un mot m_i dans un document d, est l'union de tous ses contextes locaux. On note le contexte global du mot m_i dans le document d par CG_i.
- *Un mot simple* désigne un mot non vide ayant une entrée dans WordNet.
- *Un mot orphelin* désigne un mot simple n'ayant pas d'entrée dans Wordnet.
- *Un concept* réfère à un synset (sens) particulier d'un mot.
- L'expression locale d'un mot w_i, est la chaîne de caractères commençant par le mot w_i et se terminant à la ponctuation suivante. La taille d'une expression locale est le nombre de mots qui la composent.
- *L'ensemble* $E_{Collocations}$ contient toutes les collocations d'un document *d*.
- *L'ensemble* $E_{simples}$ contient tous les mots simples d'un document *d.*
- *L'ensemble* $E_{Orphelins}$ contient tous les mots orphelins d'un document *d.*
- *L'ensemble* $E_{désambiguïsés}$ contient tous les concepts sélectionnés après désambiguïsation des mots simples du document *d*.
- *L'ensemble* $E_{Concepts}$ est l'union entre les deux ensembles $E_{Collocations}$ et $E_{désambiguïsés}$.

III.2.2. Description sommaire:

Notre approche tente de représenter de manière précise le document par un noyau sémantique composé de termes pondérés qui décrivent au mieux son contenu. Le processus d'indexation s'effectue principalement en trois étapes:

1) L'identification des termes d'index :

Le but de cette étape est d'identifier en se basant sur WordNet l'ensemble des termes représentatifs du document, ainsi que leurs catégories syntaxiques respectives. La catégorie syntaxique des termes du document est définie par un POS-Tagger (en l'occurrence le *Stanford POS tagger*[2] dans notre cas). A l'issue de cette étape, trois ensembles sont construits : l'ensemble $E_{Collocations}$, l'ensemble $E_{simples}$, et l'ensemble $E_{Orphelins}$.

2) La désambiguïsation des termes :

Un terme ambigu est associé à plusieurs synsets (sens) correspondants dans WordNet, le but de cette étape est de sélectionner le sens (concept) adéquat du terme ambigu dans le document.

3) La pondération des termes :

Dans cette étape on propose un nouveau modèle de pondération qui est une variante du modèle TF_IDF. Cette pondération s'applique aux termes de l'ensemble $E_{Concepts}$. Pour les termes qui n'ont pas d'entrées dans WordNet (l'ensemble $E_{Orphelins}$) on utilise le modèle classique TF_IDF.

La figure III.1 montre une vue d'ensemble de notre approche d'indexation. Les étapes seront détaillées en section suivante.

[2] http://nlp.stanford.edu/software/tagger.shtml

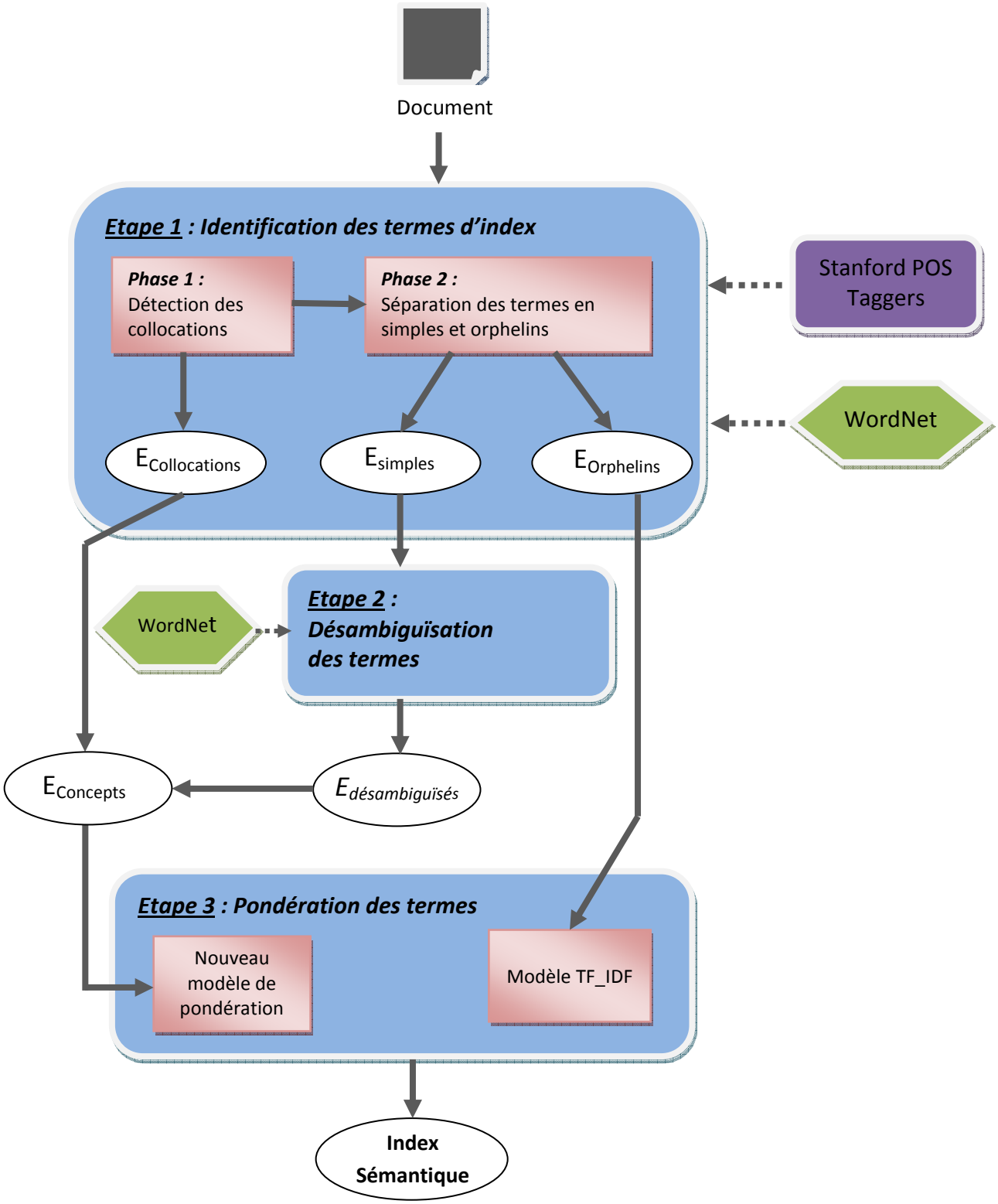


Figure III.1 : Vue d'ensemble de notre approche d'indexation sémantique.

III.2.3. Description détaillée:

III.2.3.1. Identification des termes d'index :

Le but de cette étape est d'identifier, en se basant sur WordNet, l'ensemble des termes représentatifs du document. Mais avant d'extraire les différents termes d'index du document, ce dernier est d'abord syntaxiquement annoté. L'objectif de l'annotation syntaxique (ou *POS*[3] *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 d'index.

La technique que nous proposons pour extraire les termes d'index est basée sur une analyse mot par mot du document. Elle est formellement décrite dans ce qui suit :

- ***Phase 1 :*** **Détection des collocations :**

Pour chaque occurrence d'un mot w_i à analyser, on commence par extraire l'ensemble C_i de toutes les collocations qui commencent par le mot w_i, et cela à partir d'une liste (préalablement construite) contenant toutes les collocations existantes dans WordNet. Par la suite on ordonne C_i par tailles décroissantes de ses éléments, puis on projette chaque élément de C_i sur des expressions locales E_i de w_i. Si une expression locale s'apparie avec une collocation, elle est retenue et insérée dans l'ensemble $E_{Collocations}$. Si aucune collocation de C_i ne s'apparie avec une expression locale de w_i alors w_i est passé à la phase 2.

- ***Phase 2 :*** **Séparation des termes en simples et orphelins:**

Cette phase vérifie d'abord si w_i est un mot vide, si c'est le cas il est ignoré. Par ailleurs, s'il possède une entrée dans WordNet, il est inséré dans l'ensemble $E_{simples}$, sinon w_i est inséré dans l'ensemble $E_{Orphelins.}$

Dans le tableau suivant on décrit l'algorithme de détection des termes d'index :

[3] Part Of Speech (partie du discours ou catégorie syntaxique)

Algorithme de détection des termes :

Entrée *: document d taggé.*

Sortie *:* $E_{Collocations}$, $E_{simples}$, $E_{Orphelins}$.

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

Début :

Déterminer $C_i = \{C^1_i, C^2_i, \ldots\ldots, C^n_i\}$ *l'ensemble des collocations commençant par le mot* w_i *;*

Ordonner C_i *comme suit* $C_i = \{C^{(1)}_i, C^{(2)}_i, \ldots\ldots, C^{(n)}_i\}$ *où* $(j)_{1\ldots n}$ *est une permutation d'indices telle que* $|C^{(1)}_i| \geq |C^{(2)}_i| \geq \ldots \geq |C^{(n)}_i|$*, où* $|C^{(j)}_i|$ *est la taille de la collocation* $C^{(j)}_i$ *;*

trouve ← *faux ;*

Pour chaque $C^{(j)}_i$ *dans* C_i *et trouve = faux faire :*

Calculer l'expression locale E_i *de taille* $|C^{(j)}_i|$ *:*

$E_i \leftarrow w_i \_ w_{i+1} \_ w_{i+2} \_ \ldots\ldots\ldots \_ w_{i+l}$, *tel que* $|E_i| = |C^{(j)}_i|$ *et* $l=(i+|E_i|)-1$ *;*

Si $E_i = C^{(j)}_i$ *alors*

trouve = vrai ;

fsi ;

fait ;

si trouve = vrai alors

mettre E_i *dans l'ensemble* $E_{Collocations}$ *;*

fsi ;

sinon

si w_i *mot non vide alors*

si w_i *possède une entrée dans WordNet alors*

mettre w_i *dans* $E_{simples}$ *;*

fsi ;

sinon mettre w_i *dans* $E_{Orphelins}$ *;*

fsi;

Fin.

A l'issue de cette étape trois ensembles de termes sont identifiés : l'ensemble des collocations $E_{Collocations}$, l'ensemble des termes simples $E_{simples}$ et l'ensemble des termes orphelins $E_{Orphelins}$. Enfin, nous calculons la fréquence de chaque terme dans *d,* éliminons les termes redondants (doublons) et gardons la catégorie syntaxique de chaque mot simple pour l'utiliser dans l'étape de désambiguïsation, on obtient alors les ensembles suivants :

$E_{Collocations}$= {(t_1, fq_1),….., (t_c, fq_c) / $t_i \in E_{Collocations}$, fq_i est le nombre d'occurrence de t_i dans *d.*},

$E_{simples}$ = {(t_1, pos_1, fq_1),……,(t_s, pos_s,fqs_s) / $t_i \in E_{simples}$, pos_i est la catégorie syntaxique du terme t_i, fqi est le nombre d'occurrence du couplé (t_i, pos_i) dans *d* },

$E_{Orphelins}$ = {(t_1, fq_1),…..,(t_o, fq_o) / $t_i \in E_{Orphelins}$, fq_i est le nombre d'occurrence de t_i dans *d* }

L'ensemble T(d) des termes représentatifs du document *d* est alors défini par :

$T(d) = E_{Collocations} \cup E_{simples} \cup E_{Orphelins}$.

III.2.3.2. Désambiguïsation des termes :

L'objectif de la désambiguïsation est de sélectionner parmi les sens possibles d'un terme ambigu, celui qui est sensé le définir au mieux dans le document. La désambiguïsation concernera uniquement les mots simples (l'ensemble $E_{simples}$), vu que les collocations sont des expressions quasiment désambiguïsées (la majorité des collocations est associée à un seul synset dans WordNet).

L'approche consiste à retrouver pour chaque terme simple $t_i \in E_{simples}$ tous les sens qui lui sont associés dans WordNet, puis à le désambiguïser dans son **contexte global** dans le document. Pour cela, nous posons l'hypothèse que chaque terme contribue à la définition de la sémantique du document *d* avec seulement un seul sens (même si cela est quelque peu erroné, puisqu'un terme peut avoir différents sens dans un même document, mais nous considérons ici le seul sens dominant).

En pratique, chaque terme $t_i \in E_{simples}$ peut avoir un certain nombre de sens correspondant à des synsets de WordNet. Soit $S_i = \{S^{(1)}{}_i, S^{(2)}{}_i, \ldots\ldots, S^{(n)}{}_i\}$ l'ensemble de tous les synsets associés au terme t_i. L'objectif est alors de sélectionner parmi ces synsets (sens) le sens correct dans le document. Pour cela, on associe à chaque synset (concept) $S_i^{(j)} \in S_i$, un score basé sur sa proximité sémantique avec les autres sens associés aux mots de son contexte global. Le concept $S_i^{(j)}$ ayant le plus grand score est alors retenu comme sens adéquat pour le terme t_i dans d.

Formellement :

$$S_i^{(j)} = Arg\ max \left(\sum_{\substack{1 \le l \le m \\ l \ne i}} \sum_{1 \le k \le p} Sim\left(S_i^{(j)}, S_l^{(k)}\right) \right)$$

Où :

m est le nombre de termes dans le contexte global de t_i.

p est le nombre de synsets associés à chaque terme t_l, tel que $t_l \in$ contexte global de t_i.

$Sim\ (S_i^{(j)}, S_l^{(k)})$ est la similarité sémantique entre les deux concepts $S_i^{(j)}$ et $S_l^{(k)}$ (cette mesure est détaillée en section III.2.4).

Dans le tableau suivant on décrit l'algorithme de désambiguïsation utilisé :

Algorithme de désambiguïsation

Entrée *: $E_{Simples}$.*

Sortie *: $E_{désambiguïsés}$.*

Procédure *: Soit t_i le prochain terme à désambiguïser.*

Début :

Récupérer le contexte global CG_i du terme t_i

Retrouver $S_i = \{S^{(1)}{}_i, S^{(2)}{}_i, \ldots\ldots, S^{(n)}{}_i\}$ l'ensemble de tous les synsets de t_i dans WordNet.

score_max ← 0 ;

Pour chaque synset $S_i^{(j)} \in S_i$ faire :

score [j] ← 0;

Pour chaque terme $t_l \in CG_i$ faire :

Retrouver $S_l = \{S^{(1)}{}_l, S^{(2)}{}_l, \ldots\ldots, S^{(p)}{}_l\}$ l'ensemble de tous les synsets de t_l dans WordNet.

Pour chaque synset $S_l^{(k)} \in S_l$ faire :

score [j] ← score [j] + Sim $(S_i^{(j)}, S_l^{(k)})$;

Fait ;

fait ;

Si score [j] > score_max alors

score_max ← score[j] ;

fsi ;

fait ;

Pour chaque score [j] /(j=1..n) faire :

Si score_max = score[j] alors

Prendre $S_i^{(j)}$ comme sens adéquat;

fsi ;

fait ;

Fin.

A l'issue de cette étape tous les termes ambigus seront désambiguïsés, nous aurons ainsi identifié l'ensemble des concepts associés aux mots simples (soit l'ensemble $E_{désambiguïsés}$), puis nous construisons l'ensemble $E_{Concepts}$ formé par l'union entre l'ensemble $E_{Collocations}$ et $E_{désambiguïsés}$.

III.2.3.3. Pondération des termes :

Une fois les termes extraits du document et désambiguïsés, il s'agit de leur affecter un poids qui détermine leur importance dans le document. Nous utilisons deux schémas de pondération pour pondérer les termes :

1) Pour l'ensemble des concepts, nous avons défini une formule de pondération comme variante de tf*idf qui prend en compte en plus du facteur de pondération local tf et le facteur de pondération global Idf, la proximité sémantique entre le concept à pondérer et les autres concepts de l'index.
2) Pour l'ensemble des mots orphelins, nous avons opté pour un schéma de pondération classique tf * idf, puisque les proximités sémantiques sont inexistantes.

Ainsi :

$$\begin{cases} W(t_i) = \alpha * tf(t_i) * idf(t_i) + \beta \sum_{l \neq i} Sim(t_i, t_l), \text{ pour } t_i \in E_{Concepts} \\ W(t_i) = tf(t_i) * idf(t_i) \text{ , pour } t_i \in E_{Orphelins} \end{cases}$$

Où :

tf est le nombre d'occurrence du terme dans le document d

idf est la fréquence documentaire inverse du terme, tel que :

$$idf(terme) = \log\left(\frac{nombre\ total\ de\ documents\ dans\ le\ corpus}{\text{nombre de documents contenant le terme}}\right)$$

$Sim(t_i, t_l)$ est la similarité sémantique entre les deux concepts t_i, t_l ($t_i , t_l \in E_{Concepts}$).

α et β sont des facteurs de pondération qui permettent de balancer le produit tf*idf par rapport à la proximité sémantique. Ces facteurs sont fixés expérimentalement.

III.2.4. Mesure de similarité :

Les mesures de similarité entre concepts sont utilisées dans l'étape de désambiguïsation des termes simples et dans l'étape de pondération des concepts. Plusieurs mesures de proximité sémantique ont été proposées dans la littérature utilisant des structures de réseaux sémantiques ou hiérarchiques. Ces mesures sont soit basées sur le chemin (path based measures) entre les deux concepts à comparer telles que définies par exemple dans [Rada et al. ,89] [Leacock et al., 94] [Jiang et al., 97], sur la notion de contenu d'information (Information Content ou IC) telle que définie par Wu et Palmer [Wu et al., 94], Resnik [Resnik, 99] et Lesk [Lesk, 86], ou sur une combinaison du chemin et du contenu d'information [Lin, 98].

Dons notre approche nous avons opté pour la mesure de similarité Resnik.

- **La mesure de Resnik :**

Resnik [Resnik, 99] a introduit la notion de Contenu d'Information (Information Content ou IC) des concepts en utilisant le sous-ensemble correspondant à la hiérarchie est-un (is-a) de WordNet. L'idée principale derrière cette mesure est que deux concepts sont sémantiquement liés ou proches, proportionnellement à la quantité d'information qu'ils partagent. La quantité d'information est déterminée par le contenu d'information du plus spécifique concept (noeud de la hiérarchie) qui subsume les deux concepts à comparer qu'il appelle lcs (pour Least Common Subsumer) [Baziz, 05]. Elle est définie comme suit :

$$Sim\_{resnik}(C_1, C_2) = IC(lcs(C_1, C_2))$$

Le contenu d'information (IC) d'un concept est estimé en calculant sa fréquence dans un large corpus. Il est défini comme le négatif du log de sa probabilité :

$$IC(concept) = -\log(P(concept))$$

La fréquence d'un concept dans la hiérarchie, inclut la fréquence de tous ses descendants puisque une occurrence ajoutée à un concept est aussi ajoutée aux concepts qui le subsument. Par conséquent, les concepts qui se trouvent dans la partie supérieure de la hiérarchie vont avoir les plus grandes fréquences que ceux qui se trouvent dans le niveau le plus spécifique (en bas de la hiérarchie). Ce qui justifie le moins (-) du log affecté par Resnik pour favoriser les concepts spécifiques qui se trouvent en bas de la hiérarchie.

Example (extrait de [Baziz, 05])**:**

Dans la hiérarchie de la Figure III.2, le plus spécifique nœud qui subsume les nœuds *Dime* et *Credit card* est : $lcs(dime, credit\ card) = medium\ of\ exchange$

La similarité selon Resnik entre les deux concepts est alors :

$$Sim\_{resnik}(dime, credit\ card) = IC(medium\ of\ exchange)$$

$$= -\log\left(P(medium\ of\ exchange)\right)$$

Sachant que $P(medium\ of\ exchange)$ représente le nombre d'occurrence du concept *medium of exchange* dans un corpus d'apprentissage.

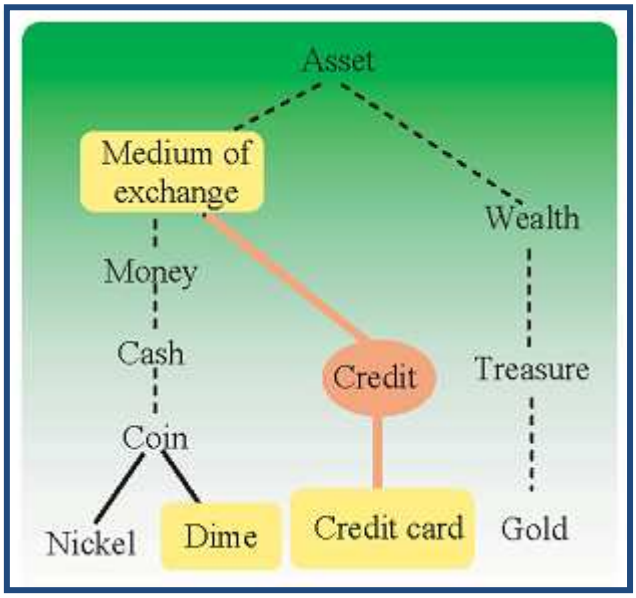


Figure III.2 : Extrait de la taxonomie de WordNet. Les lignes représentent le lien IS-A; les lignes discontinues indiquent que des nœuds intermédiaires ont été omis.

III.2.5. Illustration :

Dans le paragraphe suivant, nous illustrons par un exemple notre approche d'indexation sémantique. Nous nous focalisons en particulier sur la désambiguïsation puisque de sa précision dépend en grande partie la précision de l'indexation.

Etant donné le texte suivant (extrait du document 049.txt de la collection TIME) :

" MOROCCO DISCARDING THE EGGSHELLS A YOUNG MONARCH IN A SHAKY NEW COUNTRY CAN DO WORSE THAN CHOOSE CHARLES DE GAULLE AS A MODEL TO RULE BY . MOROCCO'S HASSAN II IS JUST SUCH A KING . LIKE LE GRAND CHARLES, HASSAN CONSIDERS HIMSELF HIS COUNTRY'S INDISPENSABLE MAN, AND HE MAY BE RIGHT . LIKE DE GAULLE HE CHOSE THE DEVICE OF A POPULAR REFERENDUM WHEN HE DECIDED TO ADOPT A CONSTITUTION (TIME, DEC 28); HIS SMASHING VICTORY WON HASSAN THE RARE ESTEEM OF HIS IDOL IN PARIS . EMPLOYING SOME GAULLIST FIRMNESS, HASSAN HAS NOW FIRED THE THREE MEMBERS OF HIS CABINET WHO REPRESENTED THE POWERFUL ISTIQLAL PARTY, FILLED THEIR JOBS WITH OFFICIALS STOUTLY LOYAL TO THE THRONE . SINCE ISTIQLAL WAS THE PARTY LARGELY RESPONSIBLE FOR ORGANIZING HASSAN'S SUCCESSFUL REFERENDUM, THERE WERE THOSE WHO THOUGHT HASSAN WAS A BIT UNGRATEFUL . BUT ISTIQLAL LEADERS WERE PRESSING FOR CLOSE ECONOMIC AND DIPLOMATIC TIES WITH CAIRO, BASED ON A COMMON ISLAMIC HERITAGE, AND DEMANDING THAT HASSAN PURSUE MOROCCO'S CLAIMS TO SPANISH SAHARA, MAURITANIA AND PART OF ALGERIA'S SAHARA . "

- **Identification des termes d'index :**

L'étape identification des termes d'index retourne les trois ensembles : $E_{Collocations}$, $E_{simples}$, $E_{Orphelins}$ suivants :

$E_{Collocations}$ = {(charles_de_gaulle , 1), (de_gaulle, 1), (spanish_sahara, 1)}

$E_{simples}$ = { (**morocco, N, 3**), (discard, V, 1), (eggshell, N, 1), (young, Adj, 1), (monarch, N, 1), (shaky, Adj, 1), (new, Adj, 1), (country, N, 2), (worse, Adj, 1), (choose, V, 2), (model, N, 1), (rule, V, 1), (king, N, 1), (le, N, 1), (grand, Adj, 1), (charles, N, 1), (consider, V, 1), (indispensable, Adj, 1),(man, N, 1), (device, N, 1), (popular, Adj, 1), (referendum, N, 2), (decide, V, 1), (adopt , V, 1), (constitution, N, 1), (time, N, 1), (28, N, 1), (smashing , N, 1), (victory, N, 1), (win, V, 1), (rare, Adj, 1), (esteem, N, 1), (idol, N, 1), (paris, N, 1), (employ, V, 1), (cat, N, 1), (firmness, N, 1), (fire, V, 1), (member, N, 1), (cabinet, N, 1), (represent, V, 1), (powerful, Adj, 1), (party, N, 2), (fill, V, 1), (job, N, 1), (official, N, 1), (stoutly, Adv, 1), (loyal, Adj, 1), (throne, N, 1), (largely, Adv, 1), (responsible, Adj, 1), (organize, V, 1), (successful, Adj, 1), (think, V, 1), (bit, N, 1), (ungrateful, Adj, 1), (leaders, N, 1), (press, V, 1), (close, Adj, 1), (economic, Adj, 1), (diplomatic, Adj, 1), (tie, N, 1), (cairo, N, 1), (base, V, 1), (common, Adj, 1), (Islamic, Adj, 1), (heritage, N 1), (demand, V, 1), (pursue, V, 1), (claim, N, 1), (mauritania, N, 1), (part, N, 1), (Algeria, N, 1), (sahara, N, 1)}.

$E_{Orphelins}$ = {(Hassan, 7), (aullist, 1), (istiqlal, 3) }.

- **Désambiguïsation des termes :**

On se propose de désambiguïser le mot simple ' **morocco** ' dans son contexte global.

Le nombre d'occurrence du terme morocco est de trois, donc le contexte global du terme morocco est l'union de ces trois contextes locaux.

$CL1_{morocco}$ = {discard/V, eggshell/N, young/Adj, monarch/N, shaky/Adj, new/Adj, country/N, worse/Adj, choose/V, charles_de_gaulle, model/N, rule/V}.

$CL2_{morocco}$ = {king/N}.

$CL3_{morocco}$ = {leaders/N, press/V, close/Adj, economic/Adj, diplomatic/Adj, tie/N, cairo/N, base/V, common/Adj, islamic/Adj, heritage/N, demand/V, pursue/V, claim/N, spanish_sahara, mauritania/N, part/N, algeria/N, sahara/N }.

$CG_{morocco} = CL1_{morocco} \cup CL2_{morocco} \cup CL3_{morocco}$ = { discard/V, eggshell/N, young/Adj, monarch/N, shaky/Adj, new/Adj, country/N, worse/Adj, choose/V, charles_de_gaulle, model/N, rule/V, king/N, leaders/N, press/V, close/Adj, economic/Adj, diplomatic/Adj, tie/N, cairo/N, base/V, common/Adj, islamic/Adj, heritage/N, demand/V, pursue/V, claim/N, spanish_sahara, mauritania/N, part/N, algeria/N, sahara/N }.

morocco est associé à deux synsets (concepts) dans WordNet :

1. ***Morocco***, *Kingdom of Morocco, Maroc, Marruecos, Al-Magrib -- (a kingdom (constitutional monarchy) in northwestern Africa with a largely Muslim population; achieved independence from France in 1956)*

2. ***morocco*** *-- (a soft pebble-grained leather made from goatskin; used for shoes and book bindings etc.)*

En calculant le score de désambiguïsation associé à chacun des concepts de morocco on obtient les résultats suivant :

- *Concept 1 (morocco#n#1) :*

Score (morocco#n#1) = Resnik (morocco#n#1, discard#v#1) + Resnik (morocco#n#1, eggshell#n#1) +……..+ Resnik (morocco#n#1, sahara #n#1) = 90.38047132133626.

- *Concept 2 (morocco#n#2) :*

Score (morocco#n#2) = Resnik (morocco#n#1, discard#v#1) + Resnik (morocco#n#2, eggshell#n#1) +……+ Resnik (morocco#n#2, sahara #n#1) = 36.0698232558338.

Le premier concept de **morocco** (morocco#n#1) obtient le score le plus élevé. On n'en conclut que le sens adéquat du terme **morocco** dans son contexte global est sa définition 1 issue de WordNet qui fait référence au pays Maroc.

III.3. Extension de Terrier à l'indexation sémantique :

Avant de présenter le système ***Sem-Terrier*** résultant de l'intégration de notre approche d'indexation dans Terrier, nous allons d'abord présenter l'architecture et le fonctionnement de Terrier.

III.3.1. Présentation de Terrier :

Terrier, *Terabyte RetrIEveR a* été développé par le département informatique de l'université Glasgow en Ecosse. C'est un moteur de recherche robuste et efficace, Il est open source et entièrement écrit en java. Terrier offre une plate forme idéale destinée à l'indexation de volumes importants de documents: jusqu'à 25 millions de documents. Terrier implémente les différents modules intervenant dans le processus de RI classique à savoir l'indexation et la recherche, et offre en plus un cadre pour l'évaluation des résultats de recherche.

Dans notre cas, nous allons nous intéresser au processus d'indexation et au processus de recherche que nous allons modifier par la suite pour intégrer notre approche d'indexation sémantique.

III.3.1.1. Le processus d'indexation de Terrier :

Pour indexer une collection de documents, Terrier est doté d'un processus d'indexation classique à quatre étapes (Voir la figure III.3) :

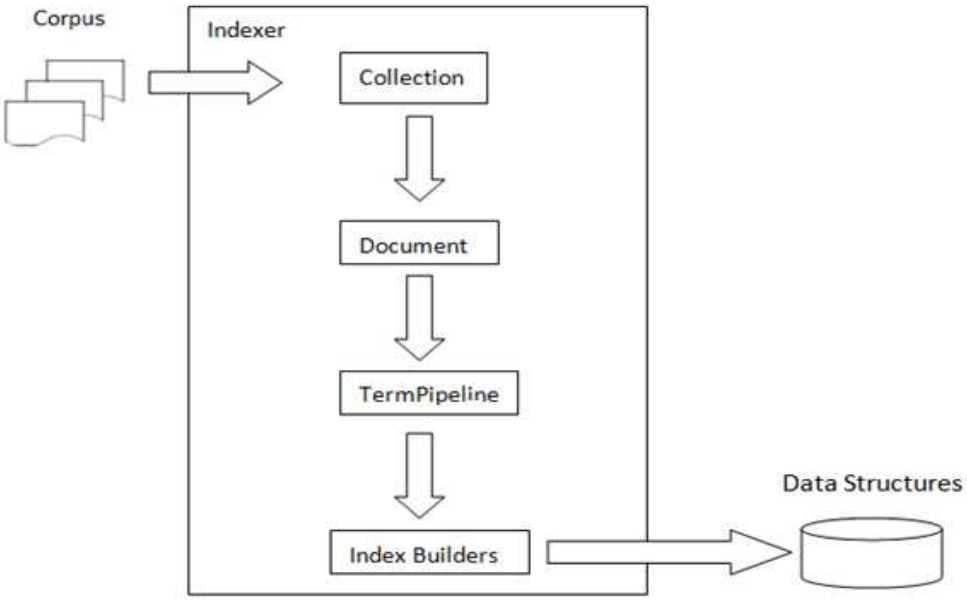


Figure III.3 : Processus d'indexation de Terrier.

- ***Splitter la collection de documents*** **:** consiste à parcourir l'ensemble du corpus et à envoyer chaque document à l'étape suivante.
- ***Extraire les termes (Tokenize document):*** consiste à parser chaque document reçu et en extraire les différents termes.
- ***Traitement des termes extraits :*** consiste à traiter tous les termes extraits via le composant TermPipeline qui va éliminer les mots vides et lemmatiser.
- ***Construction de l'index.***

Toutes ces étapes, sont présentées de façon plus détaillée en Annexe 2.

III.3.1.2. Le processus de recherche de Terrier :

Dans le processus de recherche, chaque requête doit passer par les étapes suivantes (Voir la figure III.4) :

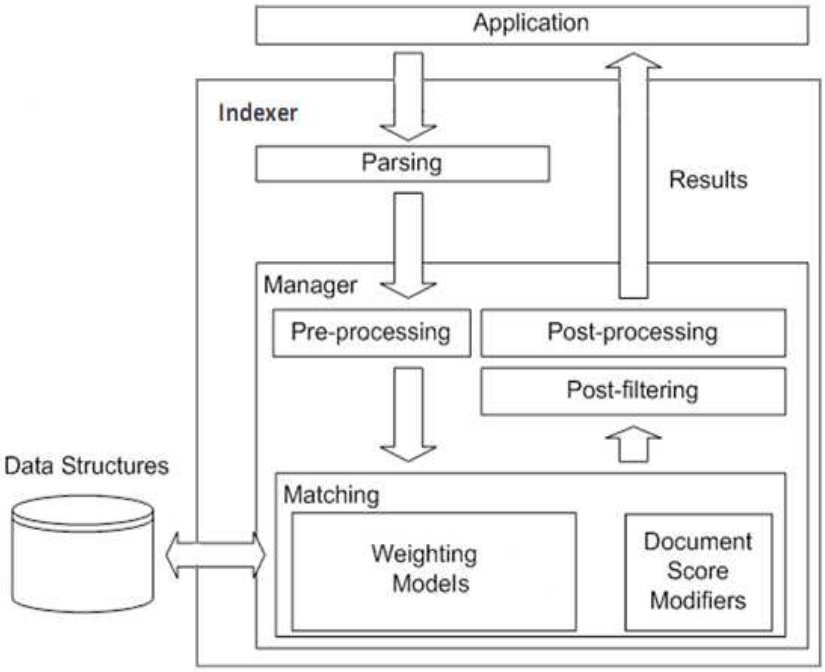


Figure III.4 : Processus de recherche de Terrier.

- **Parsing** : consiste à parser la requête et en extraire les différents termes.
- **Pre-processing** : applique le Termepipeline aux différents termes extraits (Elimine les mots vides et lemmatise).

- **Matching** : responsable de l'initialisation du WeightingModels (permet de définir un modèle de pondération : TF_IDF, BM25…etc) et du calcul des scores entre la requête et les documents.
- **post-filtrage** : Filtre les documents pertinents pour la requête
- **post-traitement** : Reclasse les documents pertinents s'il y a une expansion de la recherche

Toutes ces étapes et les modules en charge de leur exécution seront détaillés en Annexe 2.

III.3.2. Présentation de Sem-Terrier:

Le but de cette section est d'expliciter le principe de l'intégration de notre approche d'indexation sémantique à la plate forme Terrier, étendant ainsi Terrier à la prise en charge de la sémantique. Pratiquement, nous avons choisi d'intégrer notre approche d'indexation sémantique à la plate forme *Terrier 3.5.* Nous appellerons la plateforme résultante ***Sem-Terrier.***

Notre approche d'indexation est basée sur la détection des collocations et la désambiguïsation de chaque terme simple par rapport à son contexte global. Ainsi, l'identification des termes et la désambiguïsation doivent se faire préalablement à tout autre traitement comme la tokenization ou la lemmatisation de Terrier. De ce fait, l'identification et la désambiguïsation des termes sera en fait vue comme une étape préliminaire de l'indexation des documents dans Terrier. Ces mêmes traitements doivent aussi être appliqués à la requête, qui elle est traitée par le module de recherche de terrier. Le nouveau schéma de pondération des termes doit ensuite être intégré au processus de recherche de Terrier. Ainsi, nous devons:

- Redéfinir le processus d'indexation de Terrier, en y intégrant un module d'identification et de désambiguïsation des termes du document,
- Redéfinir le processus de recherche de Terrier en y intégrant (1) un module d'identification et désambiguïsation des termes de la requête et (2) notre nouveau modèle de pondération.

Les processus d'indexation et de recherche sont alors redéfinis comme décrit dans les sections suivantes.

III.3.2.1. Processus d'indexation de Sem-Terrier:

Avant de lancer la première phase du processus d'indexation de terrier c'est-à-dire *Splitter* la collection de documents, chaque document du corpus sera d'abord traité par notre module d'identification et de désambiguïsation des termes. L'indexation de Terrier se fera ensuite sur les documents préalablement désambiguïsés.

La figure III.5 illustre le processus d'indexation de Sem-Terrier.

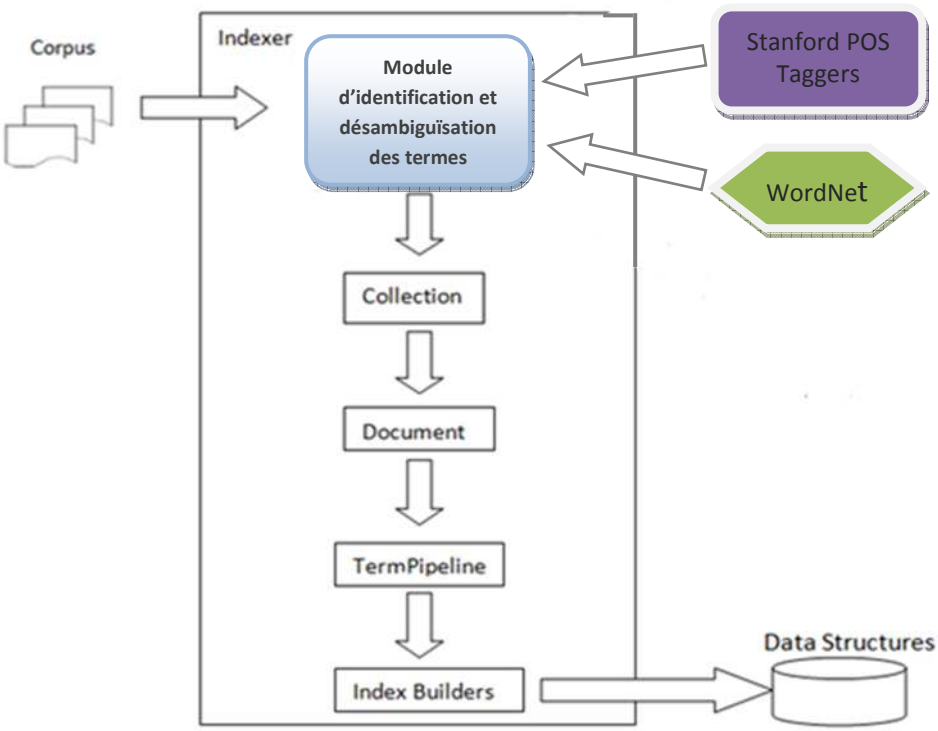


Figure III.5 : Processus d'indexation de Sem-Terrier

III.3.2.2. Processus de recherche de Sem-Terrier:

Dans le processus de recherche de Terrier deux éléments seront intégrés :

- Un module d'identification et désambiguïsation des termes de la requête
- Un nouveau modèle de pondération appelé *Sem_TF_IDF*:

Avant de lancer la première phase du processus de recherche de Terrier c'est-à-dire le parsing, chaque requête sera d'abord traitée par notre module d'identification et de désambiguïsation des termes. Puis, à la recherche, les termes de la requête sont pondérés par notre nouveau schéma de pondération Sem_TF_IDF, selon cas comme suit : Si le terme est un concept alors il est pondéré sur la base de sa distance sémantique, sinon la formule classique tf*idf est appliquée.

La figure III.6 illustre le processus de recherche de Sem-Terrier :

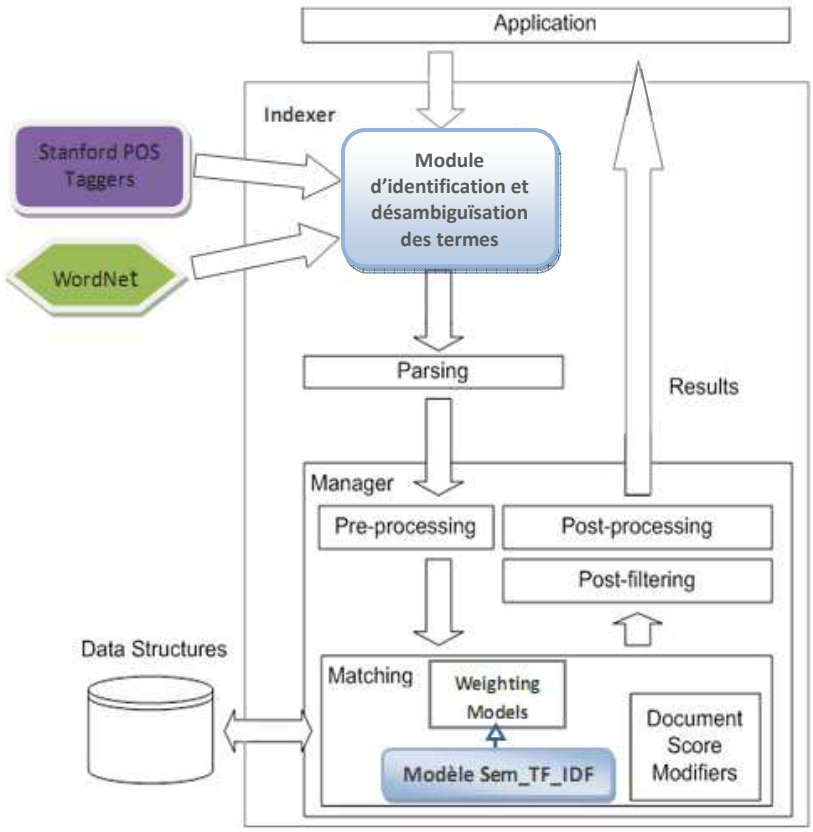


Figure III.6 : Processus de recherche de Sem-Terrier

A la fin, nous aurons intégré à la plate forme de *Terrier* trois éléments :

- Un module d'identification et désambiguïsation des termes du document.
- Un module d'identification et désambiguïsation des termes de la requête.
- Un nouveau modèle de pondération.

La figure III.7 montre la structure finale de la plate forme *Sem-Terrier*.

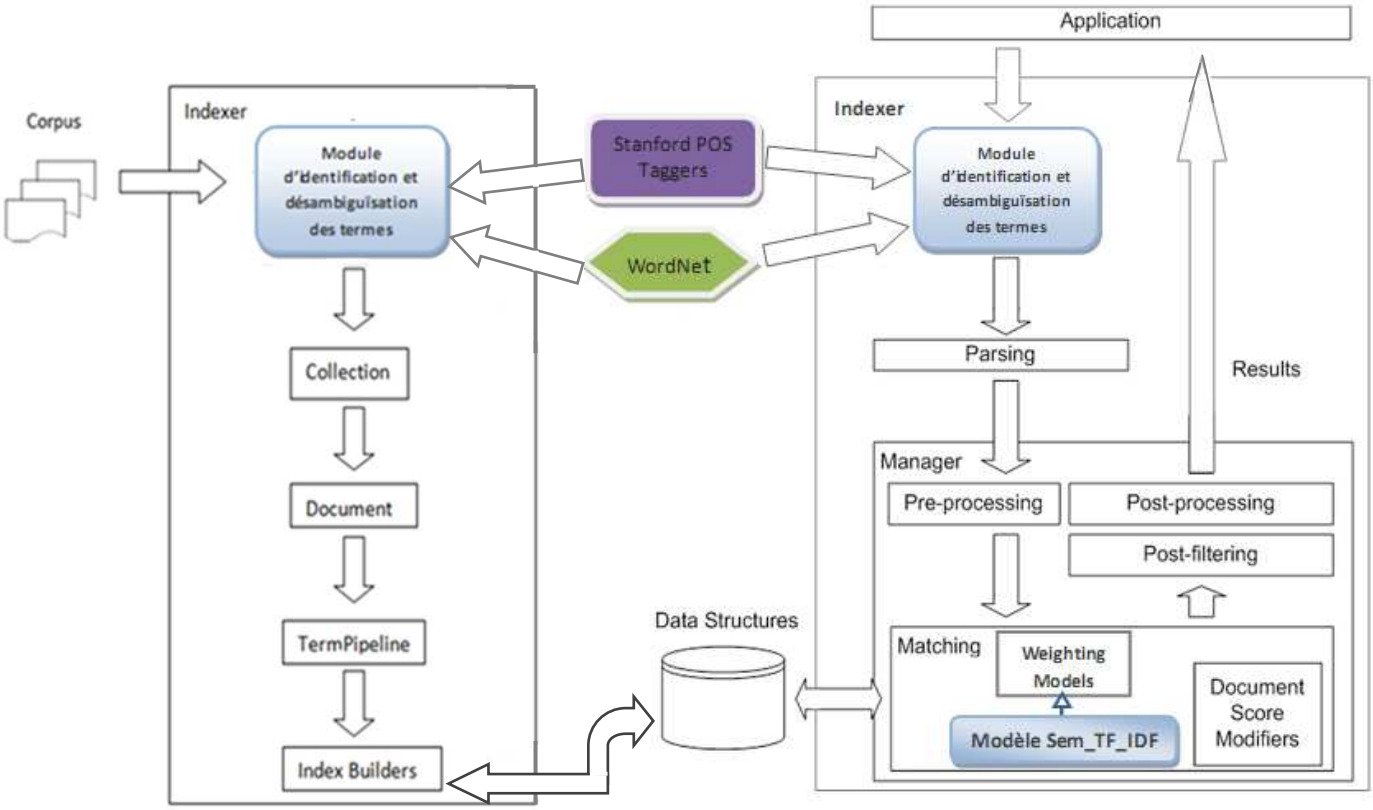


Figure III.7 Présentation finale de *Sem-Terrier*

III.3.2.3.Organisation des classes dans Sem_Terrier :

La figure III.8 présente les différentes contributions apportées par Sem_Terrier.

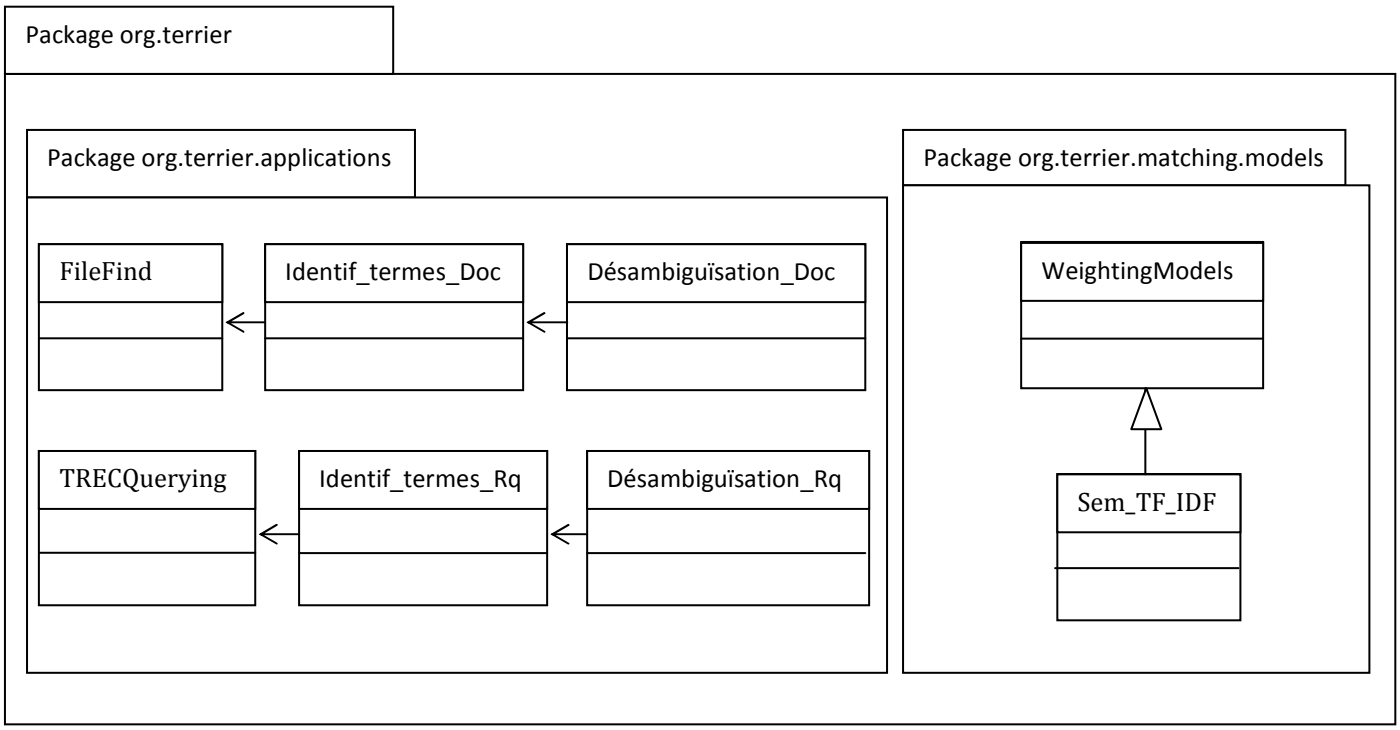


Figure III.8 : Organisation des classes.

Nous décrivons dans le Tableau III.1 les classes modélisées:

Classe	Descriptif
Identif_termes_Doc	Cette classe permet d'abord de retrouver la catégorie syntaxique de chaque mot dans le document puis, à détecter les différents termes du document.
Désambiguïsation_Doc	Cette classe permet la désambiguïsation de chaque mot simple dans son contexte global dans le document.
Identif_termes_Rq	Cette classe permet d'abord de retrouver la catégorie syntaxique de chaque mot dans la requête puis, à détecter les différents termes de la requête.
Désambiguïsation_Rq	Cette classe permet la désambiguïsation de chaque mot simple dans son contexte global dans la requête.
Sem_TF_IDF	C'est la clase qui implémente un nouveau modèle de pondération.

Tableau III.1 : description des classes proposées.

Les classes *FileFind.java, TRECQuerying.java* et *WeightingModels.java* sont les principales classes de Terrier qui permettront d'intégrer nos propres classes. Néanmoins, d'autres classes de Terrier seront modifiées.

III.4. Conclusion :

Nous avons présenté dans ce chapitre un double aspect de notre contribution : dans un premier temps, la description détaillée d'une approche d'indexation sémantique basée sur l'utilisation conjointe de WordNet et de Stanford POS Tagger. Cette approche permet la détection des termes d'index, leur désambiguïsation et leur pondération. Puis dans un second temps, nous avons présenté l'intégration de cette approche sémantique à la plate forme *Terrier 3.5*, formant ainsi une nouvelle plate forme *Sem-Terrier.*

Nous pouvons à présent implémenter et évaluer notre approche d'indexation sémantique en utilisant la plateforme de RI *Sem-Terrier*. Le chapitre suivant est consacré à la présentation détaillée de cette implémentation.

Chapitre IV

Implémentation et tests

IV.1. Introduction :

Nous avons présenté dans le chapitre précédent, d'une part, notre approche d'indexation sémantique et d'autre part, l'intégration de cette approche à la plate forme Terrier 3.5. Dans ce chapitre, nous allons d'abord présenter l'environnement technologique utilisé pour le développement de notre approche, quelques extraits du code de son implémentation, puis nous présentons les résultats de l'évaluation de notre approche d'indexation sémantique (*Sem-Terrier),* que nous comparons aux résultats d'une indexation classique (issue de Terrier 3.5).

IV.2. Environnement de développement:

Les composantes technologiques utilisées pour le développement de notre application sont décrites ci-dessous :

1) **Eclipse :** nous avons utilisé l'environnement de développement (IDE) Eclipse, pour le développement de notre application. Le choix de Java s'est imposé vu que l'Open Source de Terrier est entièrement écrit en Java.

2) **WordNet 2.1 :** ressource linguistique développée par l'université de Princeton. Elle est utilisée comme ressource pour l'identification des termes d'index et leur désambiguïsation. Nous avons utilisé deux APIs Java – WordNet :

- **L'API JWNL**[1] (Java WordNet Library): c'est une API qui permet d'interroger la base de données de WordNet (le dictionnaire), nous avons utilisé la version 1.3.

Un exemple de code d'utilisation de cette API est donné dans ce qui suit :

Exemple de code :

```
 /************Initialisation du dictionnaire***********/
JWNL.initialize(newFileInputStream("C:\\jwnl21\\config\\file_properties.xml"));
 /*********** Rechercher les Synsets  du mot mouse de type nom ************/
Dictionary dict = Dictionary.getInstance();
String mot = "mouse";
IndexWord word = dict.lookupIndexWord(POS.NOUN, mot);
Synset[] syns = word.getSenses();
       for (int i = 0; i < syns.length; i++)
            {
      System.out.println (syns[i]);     // Afficher le synset (concept)  i+1.
      System.out.println (syns[i].getKey()); // Afficher le key du synset  i+1
(le key d'un synset est un numéro qui identifie un synset de manière unique).
             }
```

- **L'API JWS**[2] (Java WordNet Similarity) : nous avons utilisé l'API Java WordNet Similarity.beta.11.01. C'est l'API qui nous permet de calculer la proximité sémantique entre deux concepts. Cette API propose plusieurs mesures de similarité dont les mesures de Resnik, AdaptedLesk, Lin…etc.

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

[1] http://jwordnet.sourceforge.net/handbook.html
[2] http://www.sussex.ac.uk/Users/drh21/

Exemple de code :

```
String dir = "C:/Program Files/WordNet"; // le chemin ou se trouve WordNet.
JWS   ws = new JWS (dir, "2.1"); // spécifier la version de Wordnet .
Resnik res = ws.getResnik(); // mesure de similarité utilisé 'Resnik'.
/**Calcul de la similarité sémantique entre les deux concepts mouse#n#4 et
computer#n#1**/
String mot1= "mouse"; mot2= "computer"; pos="n" ;
int num_syns1=4; num_syns2=1;
double score = res.res (mot1, num_syns1, mot2, num_syns2, pos) ;
```

3) **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 sont contexte. Nous avons utilisé le stanford-postagger-full-2012-03-09. Un exemple de code d'utilisation de ce tagger est présenté dans ce qui suit :

Exemple de code :

```
 /*******Initialisation du tagger*******/
MaxentTagger tagger = new MaxentTagger("tagger/wsj-0-18-left3words-
distsim.tagger");
/**********Tagger une phrase***********/
String phrase="the mouse is a hand-operated electronic device that controls the
coordinates of a cursor on your computer screen";
String phrase_taggé= tagger.tagString(phrase);
```

IV.3. Implémentation :

IV.3.1. Programme d'identification des termes d'index :

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

```
public class Identif_termes_Doc {

//**********Rechercher toutes les collocations qui commence par s dans fichier
collocations_21 puis les ordonnés par ordre décroissant*******

public static  ArrayList<String> identification_colloc(String s, String Tag)
      {
            ……;
            return(myArr);
      }
//******************Si le terme est un mot simple**************

public static boolean mot_simple(String s)
      {
            boolean a=false;

            ……;
            return (a);
      }
public static void identification_termes(File direction) throws JWNLException,
IOException, ClassNotFoundException
{   …… ;
//connexion wordnet
JWNL.initialize(new FileInputStream("C:\\ file_properties.xml"));
Dictionary dict = Dictionary.getInstance();

//Initialize the tagger
MaxentTagger tagger = new MaxentTagger("D:\\tagger\\wsj-0-18-left3words-
distsim.tagger");
…… ;
//pour chaque document
for( int j=0; j<fichier.length; j++)
        {…… ;
         while((ligne=in.readLine())!=null)
         {
           //tagger le document
            String r= tagger.tagString(ligne.toLowerCase());
               ……;
            //pour chaque mot du document
             for(int ii=0; ii<mot.size(); ii++)
               {
ArrayList<String>myArr=identification_colloc(mot.get(ii),myArrTagg.get(ii));
……;
If (trouve==false)
Boolean b = mot_simple(mot);
……;
 }
}
```

IV.3.2. Programme de désambiguïsation des termes :

Après identification des termes d'index, chaque mot simple du document (respectivement de la requête) est désambiguïsé dans son contexte global. Un extrait de notre programme de désambiguïsation des termes est donné dans ce qui suit :

```
public class Desambiguisation_Doc {

public static void Désambiguïsation() throws JWNLException, IOException,
ClassNotFoundException
      {
//connexion wordnet
JWNL.initialize(new FileInputStream("C:\\ file_properties.xml"));
Dictionary dict = Dictionary.getInstance();
// Similarité entre mots
String dir = "C:/Program Files/WordNet";
JWS   ws = new JWS(dir, "2.1");
Resnik res = ws.getResnik();
……;
//pour chaque ensemble de mots simples d'un document
for( int j=0 ; j < fichier_s.length ; j++)
      {
   //pour chaque mot simple
for (int s =0; s <terme_simple.size();s++)
{
/** ..............Recuperer le contexe global du terme.............*/
   …… ;
/**.................Desambiguisation du terme.............*/
  ……;

IndexWord word = dict.lookupIndexWord(pos,terme);
Synset[] syns = word.getSenses();

double max_score = 0;

ArrayList<Double>sc=new ArrayList<Double>();

for (int ss = 0; ss < syns.length; ss++)
{
   double   score =     0;

      for (int k = 0; k <contexte.size() ; k++)
      {
      String mot = contexte.get(k);

if(terme.equals(mot)==false &&type.equals(pos_contexte.get(k).toLowerCase()))
        {
          TreeMap<String, Double> scores = res.res(terme, ss+1, mot, type);

          for(String scor : scores.keySet())
            score = score + scores.get(scor);
         }

     }
      sc.add(score);

      if(score > max_score)
```

```
      {
         max_score= score;

      }
}
…… ;}
```

IV.3.3. Programme de pondération des termes :

Un score est affecté à chaque terme de la requête dans le document. Si le terme est un concept alors il est pondéré sur la base de sa distance sémantique, sinon le terme est un orphelin, la formule classique tf*idf est alors appliquée. Un extrait de notre programme de pondération des termes est donné dans ce qui suit :

```
public class Sem_TF_IDF extends WeightingModel {
……;
// Similarité entre mots
      String dir = "C:/Program Files/WordNet";
      JWS   ws = new JWS(dir, "2.1");
      Resnik res = ws.getResnik();

double aa =0.8;
double bb=0.2;
public final double score(
            double tf,
            double docLength,
            double documentFrequency,
            double termFrequency,
            double keyFrequency)
      {……;
/******Cas d'un concept*********/
…… ;
/***calculer la distance sémantique entre le concept à pondérer et les autres
concepts de la requête***/
            for (int k = 0; k <concept_rq.size() ; k++)
            {
            String m = concept_rq.get(k);
            if(m.contains("#"+pos+"#"))
              {
                  StringTokenizer tt= new StringTokenizer(m,"#");
                  String mot2= tt.nextToken();
                   tt.nextToken();
                  int num_syn = Integer.parseInt(tt.nextToken());

      dist_rq = dist_rq + res.res(mot, num_syns, mot2,num_syn, pos);
            }
         }
…… ;
/***calculer la distance sémantique entre le concept à pondérer et les autres
concepts du document***/
            for (int k = 0; k <concept_doc.size() ; k++)
                        {
            String m = concept_doc.get(k);
            if(m.contains("#"+pos+"#"))
                         {
```

```
        StringTokenizer tt= new StringTokenizer(m,"#");
        String mot2= tt.nextToken();
        tt.nextToken();
        int num_syn = Integer.parseInt(tt.nextToken());

dist_doc = dist_doc + res.res(mot, num_syns, mot2,num_syn, pos);
        }
}
…… ;
double idf = Idf.log(numberOfDocuments/documentFrequency+1);

return (aa * keyFrequency + bb * dist_rq ) * (aa * tf * idf + bb * dist_doc);
        }
}
```

IV.4. Evaluation Expérimentale:

L'objectif de l'évaluation expérimentale est d'étudier l'impact de notre approche d'indexation sémantique 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 un système de référence ou baseline (en l'occurrence le modèle vectoriel de Terrier 3.5) sur une collection de test donnée.

IV.4.1. Collections de tests utilisées :

Pour mener cette expérimentation nous avons opté pour l'utilisation de deux collections la MuchMore[3] et la TIME[4] :

1) **La collection MuchMore :**

Le corpus MuchMore est un corpus de résumés médicaux scientifiques obtenus à partir du site web de Springer. Il est composé de 7823 documents et de 25 requêtes ainsi que des jugements de pertinence pour chaque requête.

Vu la complexité des calculs induits par les méthodes d'identification des termes, de désambiguïsation et de pondération inhérentes à notre approche, nous avons testé notre approche que sur 851 documents de la collection avec les 25 requêtes.

[3] http://muchmore.dfki.de/about.html
[4] ftp://ftp.cs.cornell.edu/pub/smart/time/

- Exemple de document MuchMore (document *Arthroskopie.80110028.eng.abstr*) :

  Because of better clinical examination, MRI and arthroscopy, more ligamentous tears of the anterior cruciate ligament are being detected at an earlier stage. Conservatively treated secondary meniscal lesions develop very rapidly. In operative treatment there is the concern that damage to the growth plates may occur because of drilling the bone tunnels, with subsequent leg length differences and axial deviations.

- Exemple de requête MuchMore :

  104: Indication for implantable cardioverter defibrillator (ICD).

- Exemple de jugement de pertinence :

  104 0 DerAnaesthesist.80470320.txt 1

  104 0 Herzschrittmachertherapie.80090094.txt 1

  104 0 MedizinischeKlinik.00950517.txt 1

  104 0 MonatsschriftKinderheilkunde.01480138.txt 1

  104 0 ZfuerHerzThoraxGefaesschirurgie.9013s007.txt 1

2) La collection TIME :

La collection TIME est une petite collection composée de 423 documents issus d'articles de presse du magazine Time de 1963. Elle propose en outre un nombre suffisamment important de requêtes (83) et des jugements de pertinence. Nous avons testé notre approche sur toute la collection (423 documents) avec les 83 requêtes.

- Extrait d'un document TIME (document 018.txt) :

  RUSSIA WHO'S IN CHARGE HERE ? IT WAS IN 1954 THAT NIKITA KHRUSHCHEV LAUNCHED HIS GRANDIOSE " VIRGIN LANDS " GAMBLE . PART OF THE PLAN WAS TO PLOW UP 32 MILLION ACRES OF MARGINAL LAND IN KAZAKHSTAN, AND SETTLE IT WITH COMMUNIST " PIONEERS, " WHO WERE TO PLANT AND PRODUCE HUGE QUANTITIES OF DESPERATELY NEEDED GRAIN WITHIN TWO YEARS . NIKITA'S SCHEME FLOPPED . THERE WAS NOT ENOUGH RAINFALL, AND THE PIONEERS DID NOT TAKE TO TRACTOR LIFE ON THE BLEAK FRONTIER . EXCEPT FOR 1958, EACH HARVEST HAS BEEN LOWER THAN THE PREVIOUS YEAR'S .

- Exemple de requête TIME :

  6: CEREMONIAL SUICIDES COMMITTED BY SOME BUDDHIST MONKS IN SOUTH VIET NAM AND WHAT THEY ARE SEEKING TO GAIN BY SUCH ACTS.

- Exemple de jugement de pertinence :

  6 0 257.txt 1
  6 0 268.txt 1
  6 0 288.txt 1
  6 0 304.txt 1
  6 0 308.txt 1

IV.4.2. Protocole d'évaluation:

L'évaluation de notre approche est faite selon le protocole TREC. 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 Avg_P sont calculées.
La précision au point x (P@x) est le ratio des documents pertinents parmi les x premiers documents restitués. Avg_P est la moyenne des P@x, x=1,....,1000.
Il s'agit ensuite de comparer les résultats obtenus à partir de notre approche (Sem-Terrier) à ceux restitués par une approche d'indexation classique (modèle vectoriel de Terrier 3.5).

IV.4.3. Evaluation de l'approche d'indexation sémantique :

1) Avec la collection MuchMore :

Nous avons indexé les 851 documents de la collection MuchMore d'une part avec Terrier 3.5, et d'autre part avec Sem-Terrier.

Nous avons ensuite interrogé les deux systèmes avec les 25 requêtes respectivement. Les schémas de pondération utilisés sont TF_IDF pour Terrier 3.5, et Sem_TF_IDF pour Sem_Terrier. Nous avons ensuite évalué les résultats des deux recherches. Les résultats obtenus sont présentés dans le tableau IV.1.

Information	Indexation classique : Terrier 3.5 (Modèle: TF_IDF)	Indexation sémantique : Sem-Terrier (Modèle: Sem_TF_IDF)
Number of queries Retrieved Relevant Relevant retrieved	25 6696 749 339	25 3987 749 290
Average Precision : **R Precision :**	**0.2740** **0.2925**	**0.1237** **0.1547**
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 :	0.6800 0.6400 0.5867 0.5600 0.5280 0.4520 0.3813 0.3500 0.2773 0.2008 0.1280 0.0672 0.0271 0.0136	0.4000 0.3000 0.2800 0.2800 0.2560 0.2040 0.1867 0.1760 0.1427 0.1128 0.0832 0.0540 0.0232 0.0116
Precision at 0% : Precision at 10% : Precision at 20% : Precision at 30% : Precision at 40% : Precision at 50% : Precision at 60% : Precision at 70% : Precision at 80% : Precision at 90% : Precision at 100% :	1.2727 1.0051 0.7908 0.6677 0.5982 0.4322 0.2757 0.1566 0.0846 0.0000 0.0000	0.7202 0.5152 0.4103 0.2491 0.2540 0.1800 0.0919 0.0402 0.0185 0.0000 0.0000
Average Precision :	0.2740	0.1237

Tableau IV.1 : contenus des fichiers d'évaluation (fichier.eval) pour la MuchMore

Comme nous le constatons dans le tableau IV.1, la précision moyenne est passée de **0.2740** pour l'indexation classique à **0.1237** pour l'indexation sémantique. Ainsi, notre approche d'indexation sémantique est moins efficace qu'une indexation classique en utilisant la collection MuchMore. Nous pensons que ceci est du à la spécialisation de la collection Muchmore. En effet, la collection Muchmore est une collection médicale alors que WordNet

est une ressource de l'anglais générale; Ainsi, avec MuchMore on obtient de nombreux termes orphelins (des termes médicaux n'ayant pas d'entrée dans WordNet) à l'exemple des termes : *metachromatic*, *mechanoreceptors*, *intraligamentous*…etc. De plus, la majorité des collocations existantes dans le corpus MuchMore n'appartient pas à WordNet comme par exemple dans le document (Arthroskopie.00130003*.eng.abstr*), la collocation *posterior_cruciate_ligament* n'existe pas dans WordNet.

Pour y remédier, on pense qu'il est préférable d'utiliser une autre ressource que WordNet, la ressource ***Mesh***[5] par exemple qui est une ressource médicale, pour l'identification des termes d'index et leur désambiguïsation.

C'est pour cela qu'on testera notre approche avec une autre collection, en l'occurrence la collection TIME, plus générale.

2) Avec la collection TIME :

Nous avons indexé les 423 documents de la collection TIME d'une part avec Terrier 3.5, et d'autre part avec Sem-Terrier.

Nous avons ensuite interrogé les deux systèmes avec les 83 requêtes respectivement. Les schémas de pondération utilisés sont TF_IDF pour Terrier 3.5, et Sem_TF_IDF pour Sem_Terrier. Nous avons évalué les résultats des deux recherches. Les résultats obtenus sont présentés dans le tableau IV.2.

[5] http://www.ashburypress.com/medline/resources/mesh.html

Information	Indexation classique : Terrier 3.5 (Modèle: TF_IDF)	Indexation sémantique : Sem-Terrier (Modèle: Sem_TF_IDF)
Number of queries Retrieved Relevant Relevant retrieved	82 18604 323 125	82 13703 323 102
Average Precision : **R Precision :**	**0.0090** **0.0025**	**0.0108** **0.0103**
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 :	0.0000 0.0000 0.0041 0.0030 0.0024 0.0061 0.0049 0.0043 0.0033 0.0056 0.0057 0.0056 0.0030 0.0015	0.0000 0.0000 0.0000 0.0061 0.0049 0.0098 0.0065 0.0073 0.0069 0.0059 0.0060 0.0055 0.0025 0.0012
Precision at 0% : Precision at 10% : Precision at 20% : Precision at 30% : Precision at 40% : Precision at 50% : Precision at 60% : Precision at 70% : Precision at 80% : Precision at 90% : Precision at 100% :	0.0250 0.0232 0.0172 0.0166 0.0147 0.0116 0.0042 0.0033 0.0032 0.0029 0.0029	0.0292 0.0275 0.0227 0.0144 0.0130 0.0111 0.0069 0.0046 0.0046 0.0046 0.0046
Average Precision :	0.0090	0.0108

Tableau IV.2 : contenus des fichiers d'évaluation (fichiers.eval) pour la TIME.

Comme nous le constatons dans le tableau IV.2, contrairement à la collection MuchMore nous avons une amélioration de la *Précision moyenne* et de la *R Précision.* La précision moyenne est passée de **0.0090** pour l'indexation classique à **0.0108** pour l'indexation sémantique, tandis que la *R Précision* est passée de **0.0025** pour l'indexation classique à **0.0103** pour l'indexation sémantique.

Le graphique (Figure IV.1) montre l'augmentation de la précision moyenne (average precision) et de la R-precision avec notre approche d'indexation sémantique.

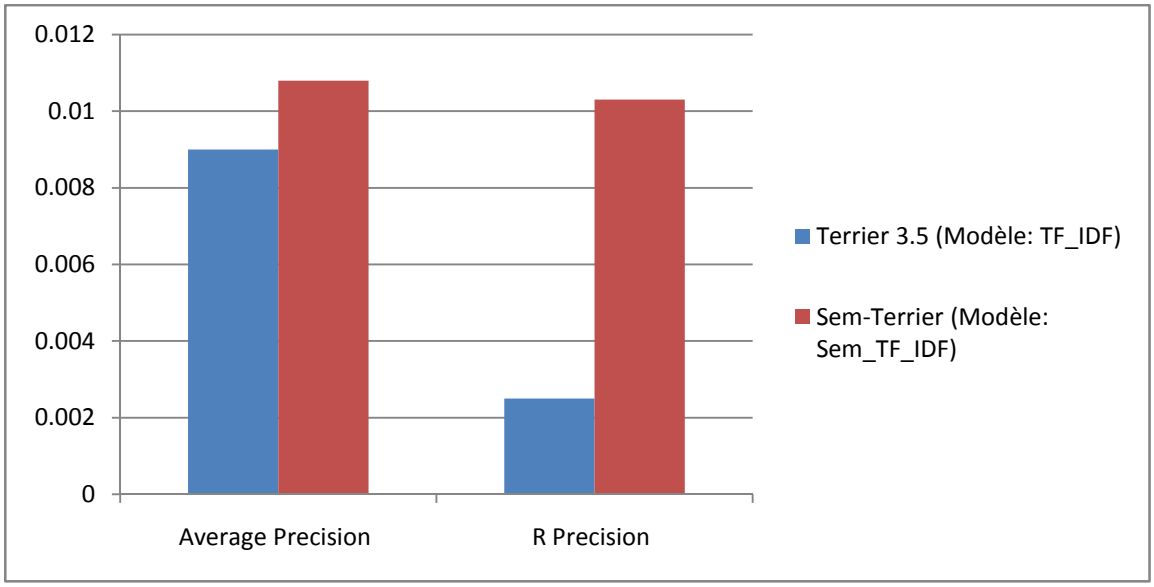


Figure IV.1 : Augmentation de la précision moyenne et de la R-precision.

Un récapitulatif des comparaisons réalisées entre l'indexation classique (Terrier 3.5) et notre indexation sémantique (Sem_Terrier) est représenté à travers le graphique (Figure IV.2):

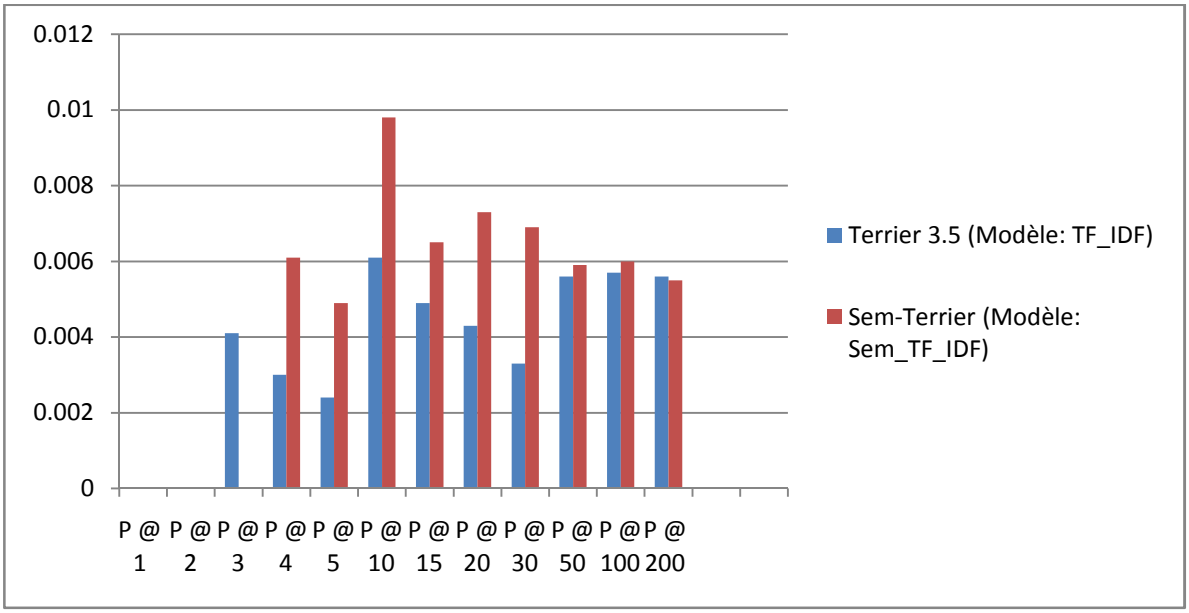


Figure IV.2 : Augmentation des précisions (P@4.....P@100).

Notre index sémantique, pondéré par Sem_TF_IDF présente de meilleurs résultats que l'index classique pondéré par TF_IDF pour les précisions comprises entre P@4 et P@100.

IV.5. Conclusion :

Dans ce chapitre nous avons présenté l'implémentation et les résultats de l'évaluation de l'approche d'indexation sémantique décrite dans ce mémoire. Les premiers résultats obtenus montrent des dégradations de performances avec la collection médicale Muchmore, et une nette amélioration des performances est observée avec la collection générale TIME. Les résultats doivent néanmoins être confirmés sur des collections de tailles plus grandes.

Conclusion générale et perspectives

Notre mémoire s'inscrit dans le domaine de la recherche d'information (RI) et plus particulièrement dans le cadre de l'indexation sémantique basée sur la désambiguïsation exogène, en se basant sur une ressource linguistique externe en l'occurrence WordNet, pour retrouver le sens correct d'un mot ambigu.

Nous avons décrit dans ce mémoire une nouvelle approche d'indexation sémantique (proposée par Amirouche). Cette approche porte sur les trois aspects de l'indexation sémantique : (1) la détection des termes d'index qui consiste à retrouver l'ensemble des collocations, l'ensemble des termes simples et l'ensemble des termes orphelins. (2) la désambiguïsation des termes qui consiste à sélectionner le sens correct de chaque mot simple dans le document et (3) la pondération des termes, où nous avons proposé un nouveau schéma de pondération basé sur la similarité sémantique (plus le terme est relié sémantiquement aux autres termes du document plus il est important). Par la suite, nous avons intégré cette nouvelle approche à la plateforme de Terrier 3.5, formant ainsi Sem-Terrier, et ce, en ajoutant un module pour l'identification des termes d'index et leur désambiguïsation, et le nouveau modèle de pondération pour le calcul des scores.

Nous avons testé notre approche avec une partie de la collection Muchmore, l'évaluation des résultats montre des dégradations de performances, ceci est du à la spécialisation de la collection. La collection Muchmore est une collection médicale alors que la ressource linguistique utilisée (WordNet) est une ressource de l'anglais générale. Cependant, nous avons testé notre approche avec une autre collection, la TIME, l'évaluation des résultats montre une nette amélioration de la précision moyenne de Sem-Terrier par rapport aux résultats obtenus en utilisant Terrier classique, ce qui montre que notre approche est assez efficace.

Les perspectives pour notre travail se déclinent en deux points, le premier est d'optimiser le temps d'exécution de nos programmes. En effet, le calcul des similarités sémantiques entre les concepts, le temps d'accès au dictionnaire WordNet et au tagger Stanford POS Tagger sont beaucoup trop lont. Le second est d'effectuer de nouvelles

évaluations sur d'autres collections de tailles plus grandes pour valider l'efficacité de notre approche.

Références bibliographiques

[Amirouche, 08] : Fatiha BOUBEKEUR-AMIROUCHE. Contribution à la définition de modèles de recherche d'information flexibles basés sur les CP-Nets. Thèse de doctorat en informatique de l'Université Toulouse III - Paul Sabatier. Juillet 2008.

[Amirouche, 11] : Fatiha AMIROUCHE , Sarah Chiout , Wassila Azzoug , Mohand Boughanem , Indexation sémantique de documents textuels Dans: 14e Colloque International sur le Document Electronique (CIDE14), Rabat, Maroc, Décembre 2011.

[Audibert, 03] : Audibert L., Outils d'exploration de corpus et désambiguïsation lexicale automatique. Thèse de Doctorat en Informatique de l'Université de Provence. Décembre 2003.

[Aussenac, 02] : N. Aussenac, Support de cours conçu par N. Aussenac-Gilles, J. Charlet, P. Laublet et B. Bachimont. Cours sur les Ontologies, les Terminologies et les Bases de Connaissances Terminologiques : http://www.irit.fr/GRACQ, (2002).

[Baziz , 05] : BAZIZ M., Indexation conceptuelle guidée par ontologie pour la recherche d'information, Thèse de Doctorat en Informatique de l'université Paul Sabatier de Toulouse, décembre 2005.

[Baziz et al, 04] : Mustapha Baziz, Nathalie Aussenac-Gilles, Mohand Boughanem Exploitation des Liens Sémantiques pour l'Expansion de Requêtes dans un Système de Recherche d'Information.

[Baeza-Yates & al., 99]: Ricardo A. Baeza-Yates, Berthier A. Ribeiro-Neto, "Modern Information Retrieval" ACM Press / Addison-Wesley. 1999.

[Berry et al.., 99] : M. W. Berry, Z. Drmac, E. R. Jessup, Matrices, vector spaces and information Retrieval. SIAM Review Vol. 41. No 2, pages : 335-362, 1999.

[Boucham, 09] BOUCHAM Souhila. Une approche basée Ontologies pour l'indexation automatique et la Recherche d'Information Multilingue (RIM), 2009.

[Boughanem, 92] : M. Boughanem, "Les Systèmes de Recherche d'Information : d'un modèle classique à un modèle connexionniste", Thèse de Doctorat de l'Université Paul Sabatier, Toulouse (France), Décembre 1992.

[Boughanem & al., 04] : Mohand Boughanem, Wessel Kraaij, Jian-Yun Nie, "Modèles de langue pour la recherche d'information " In Les systèmes de recherche d'informations, pages 163–182. Hermes-Lavoisier. 2004.

[Chahine, 11] : Carlo Abi Chahine, Indexation et recherche conceptuelles de documents pédagogiques guidées par la structure de Wikipédia. Thèse de Doctorat en Informatique de L'Institut National des Sciences Appliquées de Rouen, Octobre 2011.

[Champclaux, 09]: Un modèle de recherche d'information basé sur les graphes et les similarités structurelles pour l'amélioration du processus de recherche d'information. Thèse de Doctorat en Informatique de l'université de Toulouse, 2009.

[Croft & al., 92] : James P. Callan, W. Bruce Croft, Stephen M. Harding: The INQUERY Retrieval System. DEXA 1992: 78-83.

[Deerwester & al, 90] : Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Richard Harshman."Indexing by Latent Semantic Analysis", Journal of the American Society of Information Science, 1990.

[Dumais ,95] : S.Dumais Latent Semantic Indexing (LSI), TREC-3 Report Inproceedings of TREC-3, pages : 319-230, 1995

[Fellag, 06]: Samia FELLAG, Recherche d'information dans des documents semi-structurés XML, Magistère en informatique de l'Université MOULOUD MAMMERI de TIZI-OUZOU, septembre 2006.

[GIL, 06]: Gilles BALMISSE WWW.gillesbalmisse.com/v2/spip.php?article164.

[Gruber, 93]: T. R. Gruber, "Toward Principles for the design of Ontologies used for Knowledge Sharing," in Proc of International Workshop on Formal Ontology, Padova, Italy, March 1993.

[Guarino et al., 01] : N. Guarino, C. Welty, Identity and Subsumption, In The Semantics of Relationships : an Interdisciplinary Perspective, R. Green, C.A. Bean, S. Hyon Myseng (Eds), Kluwer, pp 111-126, 2001.

[Harwood, 90] : J.D. Harwood. "Neural Network Implementation of a novel heuristic neural algorithm". M.S. Maryland University, College Park, 1990.

[Hlaoua, 07] : Lobna HLAOUA Reformulation de Requêtes par Réinjection de Pertinence dans les Documents Semi-Structures. Thèse de Doctorat en Informatique de l'université de Toulouse, 2007.

[Ingwersen, 92]: P. Ingwersen. Information retrieval interaction. London, Taylor Graham, 1992.

[Jiang et al., 97]: Jiang J. & Conrath D. (1997) Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings on International Conference on Research in Computational Linguistics, Taiwan, 1997.

[Katz et al., 98] : Özlem Uzuner, Boris Katz, Deniz Yuret : Word Sense Disambiguation for Information Retrieval. AAAI/IAAI 1999 : 985

[Khan, 00] : Latifur R. Khan, "Ontology-based Information Selection, Phd Thesis", Faculty of the Graduate School, University of Southern California. August 2000.

[Krovetz, 93]: Krovetz R, "Viewing Morphology as an Inference Process", in Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 191-202, 1993.

[Krovetz, 97]: R.Krovetz. 1997. Homonymy and polysemy in information retrieval . In Proceedings of the 35 th Annual Meeting of the Association for Computational Linguistics (ACL-97}, pages 72-79.

[Krovetz & al., 92]: R. KROVETZ and W. B. CROFT. Lexical Ambiguity and InformationRetrieval.ACMTransactionsonInformationSystems,Vol.10,No2,pp.115_141. April 1992.

[Kwok, 89]: K.L. Kwok. "A neural network for probabilistic information retrieval". 12th International ACM SIGIR Conference on Research and Developpement in Information Retrieval, pp 21-30, 1989.

[Kwok, 95] : K.L. Kwok. "A network approach to probabilistic information retrieval". ACM transactions on information systems. Pages 324-353, 1995.

[Leacock et al., 98]: Leacock, C., Miller, G. A., and Chodorow, M. 1998. Using corpus statistics and WordNet relations for sense identification. Comput. Linguist. 24, 1 (Mar. 1998), 147-165.

[Lesk, 86]: Lesk M.E., Automatic sense disambiguation using machine readable dictionaries : How to tell a pine cone from a nice cream cone. In Proceedings of the SIGDOC Conference. Toronto, 1986.

[Lin, 98]: D. Lin. (1998) An information-theoretic definition of similarity. In Proceedings of 15th International Conference On Machine Learning, 1998.

[Mallak, 11] : Ihab Mallak . De nouveaux facteurs pour l'exploitation de la sémantique d'un texte en Recherche d'Information. Thèse de Doctorat en Informatique de l'université Paul Sabatier de Toulouse, décembre 2011.

[Mammeri ,09] : MAMMERI Karima. Recherche d'information par croisement de média texte et image. Magister en Informatique de l'université M'hamed BOUGARA de BOUMERDES. 2009.

[Maron & al., 60] : Maron, M. E., & Kuhn, J. L.. "On relevance, probabilistic indexing, and information retrieval". Journal of the Association for Computing Machinery, 7(3), 216-244. 1960

[Mataoui , 07] : Mataoui m'hamed. Reformulation de requête dans les systèmes de RI dans les documents XML.

[Mihalcea et al., 00] Mihalcea, R. and Moldovan, D. : Semantic indexing using WordNet senses. In Proceedings of ACL Workshop on IR & NLP, Hong Kong, October 2000

[Mothe, 94] : J. Mothe : "Modèle Connexionniste pour la Recherche d'Information, Expansion dirigée de requêtes et apprentissage", Thèse de Doctorat de l'Université Paul Sabatier, Toulouse (France), 1994.

[Ponte & al, 98] : Jay M. Ponte, W. Bruce Croft: "A Language Modeling Approach to Information Retrieval". SIGIR, 1998.

[Prié, 00] : Yannick Prié, « Modélisation de documents audiovisuels en Strates Interconnectées par les Annotations pour l'exploitation contextuelle » Thèse disponible sur l'url : http://lisi.insa-lyon.fr/yprie/these/node1.html, (2000).

[Quillian, 68]: Quillian, M.R. (1968). Semantic Memory. In M. Minsky (Ed.), Semantic Information Processing (pp. 227-270). MIT Press.

[Rada et al. ,89] : [Rada et al., 1989] Rada, R., Mili, H., Bicknell, E., and Blettner, M. (1989). Development and application of a metric on semantic nets. IEEE Transaction on Systems, Man, and Cybernetics, 19(1):17–30.

[Resnik, 99]: Resnik, P., "Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language", Journal of Artificial Intelligence Research (JAIR), 11, pp. 95-130, 1999.

[Rijsbergen, 79]: C. van Rijsbergen. Information retrieval. Butterworths. 1979.

[Robertson et al., 97]: S. E. Robertson and S. Walker. On relevance weights with little relevance information. In Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, pages 16–24.ACM Press, 1997.

[Roberston, 76]: Robertson, S. E., & Sparck Jones, K. "Relevance weighting of search terms". Journal of the American Society for Information Science, 27, 129–146. 1976

[Robertson, 94]: ROBERTSON S. E., WALKER S.,«Some Simple Effective Approximations tothe2%Poisson Model for Probabilistic Weighted Retrieval», Proceedings of SIGIR 1994,

[Said L'hadi, 09]: SAID L'HADJ Lynda. Recherche Conceptuelle d'Information Modèle d'Indexation Mixte : concepts-mots. Magister en Informatique de l'ecole nationale Supérieure d'Informatique (E.S.I) (ex. I.N.I). 2009 .

[Salton & al., 83]: Salton, G., E.A. Fox, H. Wu. "Extended Boolean information retrieval system". CACM 26(11), pp. 1022-1036, 1983.

[Salton, 71] : G. Salton, "A Comparison between manual and automatic indexing methods". Journal of the American Documentation, 20(1), pp. 6171, 1971.

[Savoy, 03] : Jacques SAVOY. "Modèles en Recherche d'information". In Assitance Intelligente à la Recherche d'Information. Edition lavoisier. 2003

[Shaw et al, 97]: W. Shaw, R. Burgin, and P. Howell. Performance standards and evaluations in IR test collections : Cluster-based retrieval models. Information Processing and Management, 33(1) :1–14. 1997.

[Singhal et al., 1995] A. Singhal, G. Salton, M. Mitra, and C. Buckley. Document length normalization. Information Processing and Management, 32(5) :619–633. 1995.

[Tebri, 04]: Hamid TEBRI Formalisation et spécification d'un système de filtrage incrémental d'information, Thèse de Doctorat en Informatique 2004.

[Vasilescu, 03] : Florentina Vasilescu. Désambiguïsation de corpus monolingues par des approches de type Lesk. Thèse de Doctorat en Informatique de l'université de de Montréal. Aout 2003.

[Voorhees, 93]: Voorhees, E. M. Using WordNet to disambiguate word senses for text retrieval. Association for Computing Machinery Special Interest Group on Information Retrieval. (ACM-SIGIR-1993) : 16thAnnual International Conference on Research and Development in Information Retrieval, 171–180. (1993).

[Weiss, 73]: Weiss, S. F. (1973). Learning to disambiguate. Information Storage and Retrieval, 9, 33_41.

[Wong & al., 85]: Wong, S., Ziarko, W. et Wong, P. (1985). Generalized vector spaces model in information retrieval. In Proc. of the 8th ACM-SIGIR conference, pages 18-25. Montreal, Quebec.

[Woods, 97]: William A. Woods. 1997. Conceptual indexing : A better way to organize knowledge. Technical Report SMLI TR-97-61, Sun Microsystems Laboratories, Mountain View, CA, April. ww.sun.com/research/techrep/1997/abstract-61.html.

[Zargayouna, 05] : Haïfa Zargayouna. Indexation sémantique de documents XML. Thèse de Doctorat en Informatique de l'université Paris XI Orsay. Décembre 2005.

[Zemirli, 04] : Zemirli W.Nesrine. Vers le développement d'un système de recherche d'information personnalisé intégrant le profil utilisateur. Formation Doctorale en informatique Université Paul Sabatier & Institut National Polytechnique de Toulouse. 2004.

Annexe I

WordNet

1. Introduction :

Cette annexe est consacrée à la présentation de WordNet, une ressource linguistique utilisée dans notre approche d'indexation sémantique pour l'identification des termes d'index et leur désambiguïsation.

2. Présentation de WordNet :

WordNet est une base de données lexicale construite manuellement par des linguistes du laboratoire des sciences cognitives de l'université de Princeton. C'est un réseau sémantique de la langue anglaise, qui se fonde sur une théorie psychologique du langage. La première version diffusée remonte à juin 1991.

Son but est de répertorier, classifier et mettre en relation de diverses manières le contenu sémantique et lexical de la langue anglaise. Le système se présente sous la forme d'une base de données électronique qu'on peut télécharger sur un système local. Des interfaces de programmation sont disponibles pour de nombreux langages.

3. Structure de WordNet :

WordNet contient seulement les noms, les verbes, les adjectifs et les adverbes, organisés dans différentes structures, car ils suivent des règles grammaticales différentes.

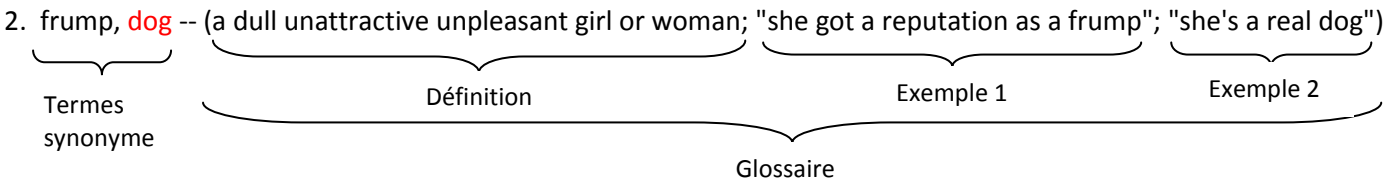
Noms, verbes, adjectifs et adverbes sont organisés en un réseau de nœuds et de liens, les nœuds sont constitués par des ensembles de termes synonymes appelés synsets et les liens représentent les relations entre les synsets qui sont de type lexicales ou sémantiques.

3.1. Les synsets :

Un synset désigne la composante atomique (ou concept) sur laquelle repose WordNet. Il est constitué de trois parties :

1) Le terme représentant du Synset : c'est le terme pour lequel le concept (synset) est identifié.
2) Les termes synonymes : une liste de termes synonymes exprimant un même concept, séparés par des virgules.
3) Le glossaire : il est mis entre parenthèses. Il contient une définition du concept avec éventuellement un ou plusieurs exemples du monde réel.

L'exemple suivant montre un nœud correspondant au sens numéro 2 du mot "dog" dans WordNet:



3.2. Les relations entre synsets :

Il existe deux types de relations entre les synsets : lexicales et sémantiques.

3.2.1. Les relations lexicales :

Les relations sont exprimées à partir des formes des mots. On trouve les relations suivantes [Vasilescu, 03] :

- **Antonymie** : deux mots sont antonymes s'ils comportent des sens opposés l'un à l'autre (par exemple, *skilled/unskilled, animate/inanimate*). Dans WordNet l'antonymie est considérée plutôt comme une relation lexicale car, dans beaucoup de cas, elle suppose l'ajout d'un préfixe (*un, in, non*) ou d'un suffixe (*less*).

- **Pertainymie** : relation d'appartenance appliquée aux adjectifs relationnels de WordNet pour indiquer le nom de provenance (par exemple, *academic* est relié par ce type de relation au synset *academia, academe*).
- **Participe** : un adjectif est en relation de participe avec le verbe d'où il dérive (WordNet relie, par exemple, l'adjectif *applied* au synset *use, utilise, apply, employ*).
- **Voir aussi** : sont des renvois qui apportent des informations supplémentaires à la description d'un synset (par exemple, le synset *drink*, *imbibe* comporte une relation voir aussi vers le synset *drain_the_cup*, *drink_up = drink to the last drop*).
- **Dérivé d'un adjectif** : relation qui relie un adverbe et l'adjectif d'où il dérive (*negatively/negative*).

3.2.2. Les relations sémantiques :

Les relations sont établies à partir des sens des mots. On trouve les relations :

- **hyperonymie/hyponymie** : on entend par hyperonyme un terme dont le sens inclut d'autres termes, qui sont ses hyponymes. X est un hyponyme de Y si X est un type de (kind of / is a) Y, Y est alors un hyperonyme de X. Par exemple, le synset *canine, canid* est l'hyperonyme de *dog* qui est lui-même l'hyperonyme de *working_dog*,
- **méronyme/holonyme** : X est un méronyme de Y si X est une partie constituante (part of), ou membre (member of) de Y. Y est alors dit un holonyme de X. par exemple : *hound, hound_dog* est un membre méronyme du holonyme *pack*.
- **Engendrement** : relation qui suppose l'enchaînement logique entre deux synsets verbaux, comme par exemple, *wear, have_on* suppose logiquement *dress, get_dressed.*
- **Cause** : relation qui exprime l'aspect causatif/résultatif entre deux synsets verbaux (*show/see*, produce, *bring_on, bring_out/appear*).
- **Similarité** : relation indiquant le fait que la classe de noms modifiés par un adjectif est incluse dans la classe de noms correspondant à un autre adjectif.

- **Attribut** : relation qui relie les adjectifs descriptifs de WordNet avec les noms qu'ils peuvent déterminer (par exemple, l'adjectif *short* est relié par une relation d'attribut au synset *duration, length*).

WordNet ne traite pas de relations de type syntagmatique, i.e. de relations établies entre les mots appartenant à des catégories syntaxiques différentes dans le cadre de la phrase, les 4 catégories fondamentales (nom, verbe, adjectif et adverbe) étant traitées séparément (chaque catégorie à sa propre structure).

3.3. Les noms :

Les synsets de noms sont organisés de façon hiérarchique, à partir d'une racine unique *{nommée Entity}*, puis 25 nœuds fils décrivant des champs sémantiques différents qui sont :

{act, action, activity}	*{event, happening}*	*{natural object}*	*{quantity, amount}*
{animal, fauna}	*{feeling, emotion}*	*{natural phenomenon}*	*{relation}*
{artifact}	*{food}*	*{person, human being}*	*{shape}*
{attribute, property}	*{group, collection}*	*{plant, flora}*	*{state, condition}*
{body, corpus}	*{location, place}*	*{possession}*	*{substance}*
{cognition, knowledge}	*{motive}*	*{process}*	*{time}*
{communication}			

Figure 1 : racines des noms dans WordNet

Les relations fondamentales qui relient les synsets de noms sont les relations hyponyme/hyperonyme qui créent la hiérarchie, et on trouve aussi des relations d'antonymie et de méronymie/holonymie comme le montre la figure 2.

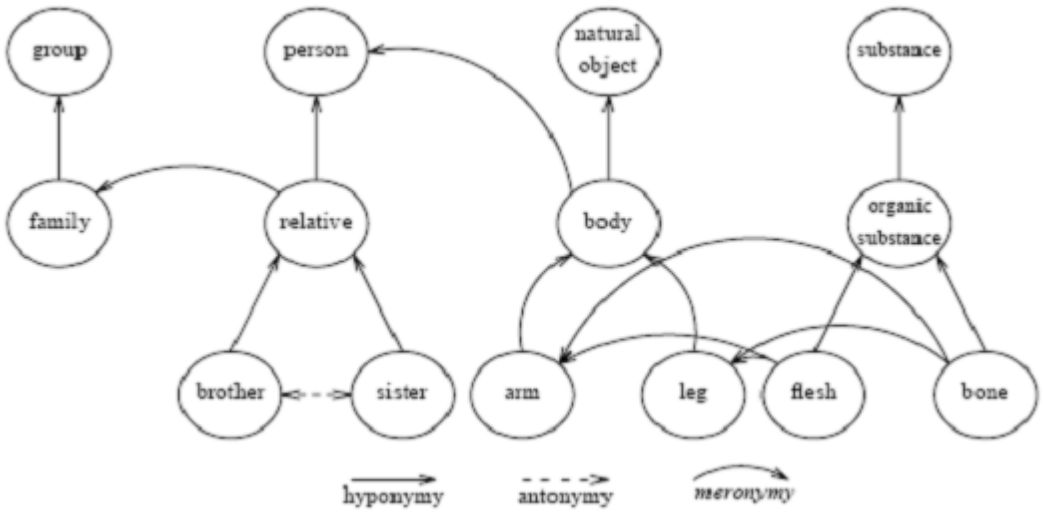


Figure 2: exemple de la hiérarchie des noms dans WordNet [Miller et al., 1990]

La relation de synonymie entre les noms est implicite dans l'arbre, car tous les mots qui sont dans le même synset sont des mots synonymes.

Par exemple, si nous cherchons le mot « *car* » dans WordNet, les sens possibles sont :

1. *car, auto, automobile, machine, motorcar -- (a motor vehicle with four wheels; usually propelled by an internal combustion engine; "he needs a car to get to work")*
2. *car, railcar, railway car, railroad car -- (a wheeled vehicle adapted to the rails of railroad; "three cars had jumped the rails")*
3. *cable car, car -- (a conveyance for passengers or freight on a cable railway; "they took a cable car to the top of the mountain")*
4. *car, gondola -- (the compartment that is suspended from an airship and that carries personnel and the cargo and the power plant)*
5. *car, elevator car -- (where passengers ride up and down; "the car was on the top floor")*

Notamment, dans le cas où le mot « *car* » est utilisé comme sens numéro 1, les mots *car, auto, automobile, machine, motorcar* sont des mots synonymes.

3.4. Les verbes :

Selon le même principe hiérarchique que pour les noms, les verbes sont beaucoup moins nombreux dans la langue que les noms et aussi beaucoup plus polysémiques, ce qui signifie que les sens des verbes sont plus flexibles que ceux des noms. Les verbes les plus utilisés (have, be, run, make, set, go, take...) sont aussi les plus polysémiques, et leur sens ont souvent une forte dépendance avec les noms avec lesquels ils co-occurrent.

Les verbes dans WordNet sont divisés en 15 groupes. 14 le sont à partir d'un critère sémantique, on distingue ainsi : les verbes de *mouvement, perception, contact, communication, compétition, changement, cognition, consommation, création, émotion, possession, soin et fonctions du corps, ou encore comportement social et interaction*.

Certains champs sémantiques sont représentés par plusieurs arbres indépendants, comme par exemple les verbes de mouvement qui comportent deux racines exprimant deux concepts distincts (move1 – mouvement de translation, move2 – mouvement sans déplacement) et les verbes de communication dissociés en deux branches indépendantes, verbes de communication verbale et non verbale.

3.5. Les adjectifs :

Dans le cas des adjectifs, l'organisation est complètement différente. L'idée de hiérarchie ne s'applique pas de la même manière que pour les noms et les verbes. La classification des adjectifs dans WordNet dépend du type de celui-ci, qui peut être soit descriptif, soit relationnel.

- **Les adjectifs descriptifs** : sont les adjectifs qui correspondent à la valeur d'un attribut, les adjectifs de type *beautiful, interesting, possible, married* sont implicitement reliés à la notion d'attribut. Par exemple, la phrase The package is heavy implique l'attribut WEIGHT tel que : WEIGHT(package) = heavy. Les antonymes heavy/light peuvent être considérés ainsi comme des valeurs possibles de l'attribut WEIGHT. Les adjectifs descriptifs sont reliés par des relations de type attribut aux noms qu'il peuvent modifier (par exemple, heavy est relié au nom weight).

Comme nous pouvons le voir dans la Figure, ces adjectifs sont organisés en termes d'oppositions binaires (antonymie) et de similarité de sens (≈ synonymie).

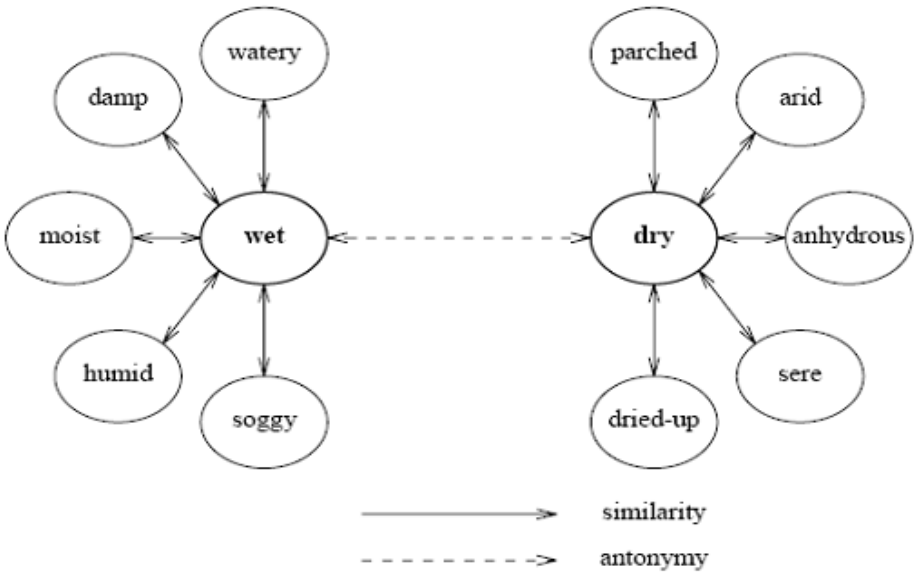


Figure 3: exemple de la structure bipolaire des adjectifs dans WordNet [Miller et al., 1990]

Souvent, les adjectifs descriptifs qui n'ont pas d'antonyme direct, ont des antonymes indirects à partir des adjectifs similaires, comme par exemple (voir figure 4) :

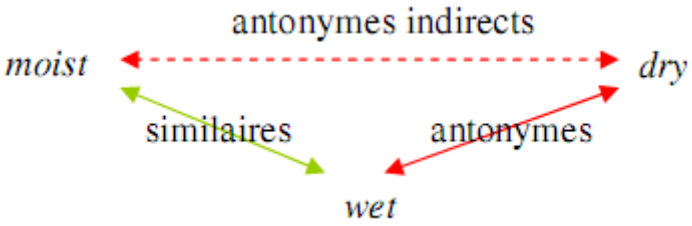


Figure 4 : antonymes indirects.

- **Les adjectifs relationnels** sont les adjectifs dérivés de noms comme par exemple *electrical* est un dérivé du nom *electricity*.

  Cette relation implique un lien de type sémantique et morphologique avec le nom d'origine. Pourtant le lien morphologique n'est pas toujours direct, comme dans le cas de l'adjectif *dental* relié au synset *tooth* via le mot latin *dens*.

  A la différence des adjectifs descriptifs, les adjectifs relationnels ne supposent pas une relation d'attribut avec le nom déterminé et n'acceptent pas de degrés de comparaison (les expressions telles que : * the hygiene is dental ou * the very electrical field ne sont pas acceptables).

3.6. Les adverbes :

À ce jour, il n'y a aucune catégorisation hiérarchique définie pour les adverbes. Les adverbes sont définis à partir des adjectifs dont ils dérivent et ainsi héritent de la même structure (pertainym). En plus, il existe aussi des relations d'antonymie.

4. Quelques données statistiques :

Le tableau 1 montre la structure de WordNet 2.1 en nombre de mots, nombre de synsets et nombre de sens, globalement et par catégorie grammaticale.

Partie de discours	Chaîne de caractères unique	Synsets	Total de couples mot-sense
Noms	117097	81426	145104
Verbes	11488	13650	24890
Adjectifs	22141	18877	31302
Adverbes	4601	3644	5720
Totales	155327	117597	207016

Tableau 1 : Nombre de mots, de synsets et de sens de mots dans WordNet 2.1.

Le tableau 2 donne des informations sur le caractère monosémique ou polysémique des mots selon leur catégorie grammaticale.

Partie de discours	Mots et sens monosémiques	Mots polysémiques	Sens polysémiques
Noms	101321	15776	43783
Verbes	6261	5227	18629
Adjectifs	16889	5252	14413
Adverbes	3850	751	1870
Totales	128321	27006	78695

Tableau 2 : répartition de la polysémie et la monosémie dans WordNet 2.1.

Le tableau 3 montre le degré moyen de polysémie des mots dans WordNet 2.1.

Partie de discours	Polysémie moyenne (incluant les mots monosémiques)	Polysémie moyenne (excluant les mots monosémiques)
Noms	1.23	2.77
Verbes	2.16	3.56
Adjectifs	1.41	2.74
Adverbes	1.24	2.49

Tableau 3 : Polysémie moyenne dans WordNet 2.1.

5. Limites de WordNet :

WordNet a été très utilisé en recherche d'information, néanmoins, il a été critiqué sur de nombreux points notamment par [Baziz, 03] et [Chaumartin, 07]. Ces critiques peuvent être résumées en ces points :

- WordNet ne couvre pas la totalité des mots de la langue anglaise, certains mots pourtant courants ne sont pas reconnus.
- WordNet ne précise pas l'étymologie, la prononciation, les formes de verbes irréguliers et ne contient que des informations limitées sur l'usage des mots.
- La contrepartie de son importante couverture est que WordNet est très précis dans le sens des définitions. On a une granularité très (trop ?) fine des sens. En effet, pour un concept donné on peut avoir une multitude de sens différents. Par exemple, le verbe to give (« donner ») n'a pas moins de 44 sens. Ce qui complique la désambiguïsation.
- L'ordre des sens retournés par WordNet n'est pas toujours celui attendu. Comme par exemple pour le mot « whale », WordNet retourne le premier sens correspondant à une personne de corps volumineux, pour ensuite donner en deuxième position le vrai sens de « whale » qui est de baleine.

Annexe II

La plateforme de RI Terrier 3.5

1. Introduction :

Cette annexe est consacré 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.

2. Présentation de Terrier :

Terrie , *Terabyte RetrIEveR* est un moteur de recherche robuste et efficace, développé par le département informatique de l'université Glasgow 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 formulés par l'utilisateur.
- **L'évaluation :** permet d'évaluer les résultats de la recherche.

3. 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[1].
- 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 plate forme de Terrier.
- Décompresser le contenu du fichier terrier-3.5.zip téléchargé.

4. Structure de Terrier :

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

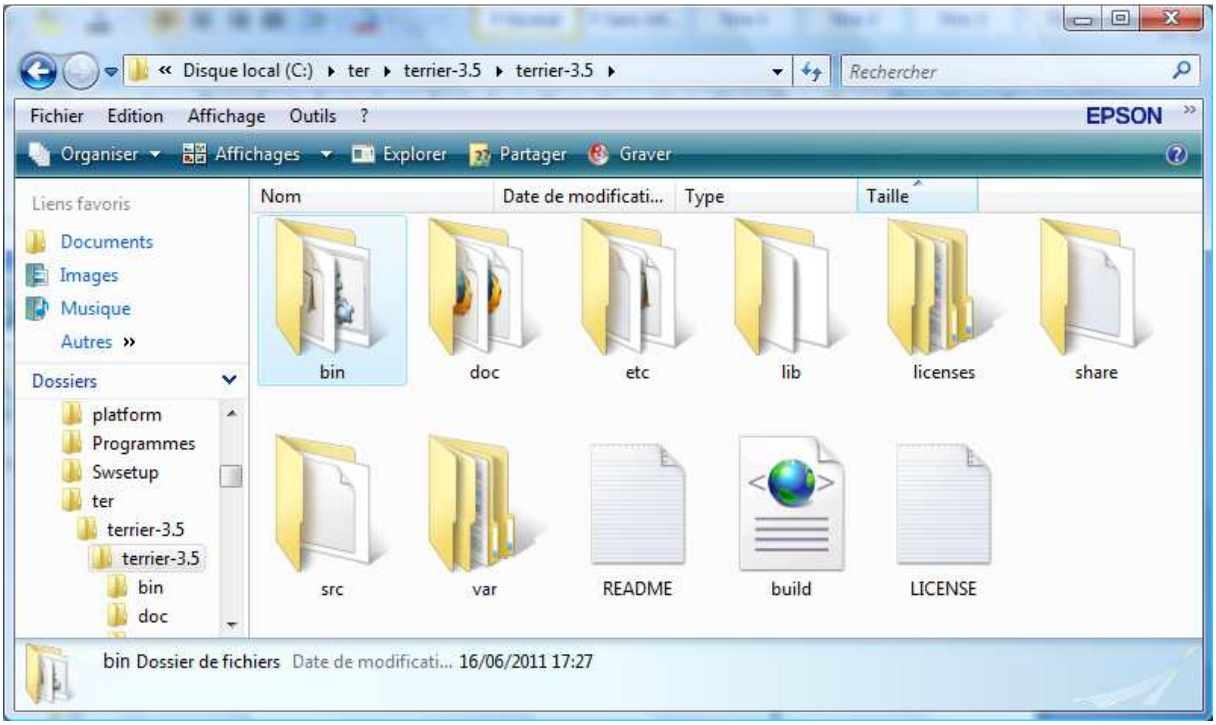


Figure 1 : Structure de Terrier 3.5.

[1] http://www.java.com/fr/

- bin\ : contient l'ensemble des scripts, nécessaire 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 crée après une indexation, le second est crée 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.

5. Composants de Terrier :

5.1. 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.

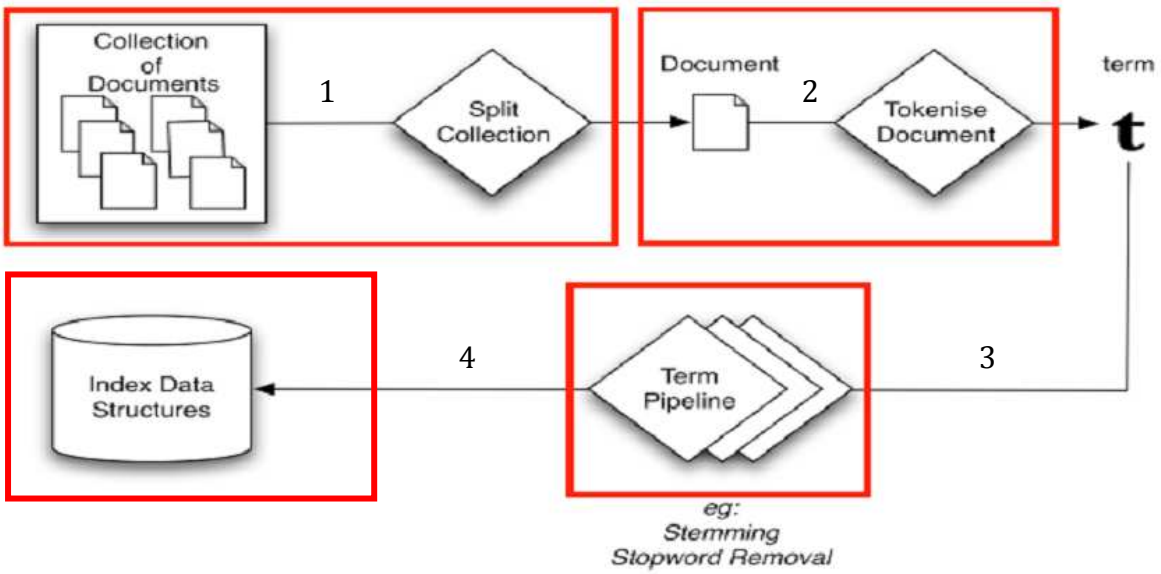


Figure 2 : Illustration du processus d'indexation de Terrier.

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

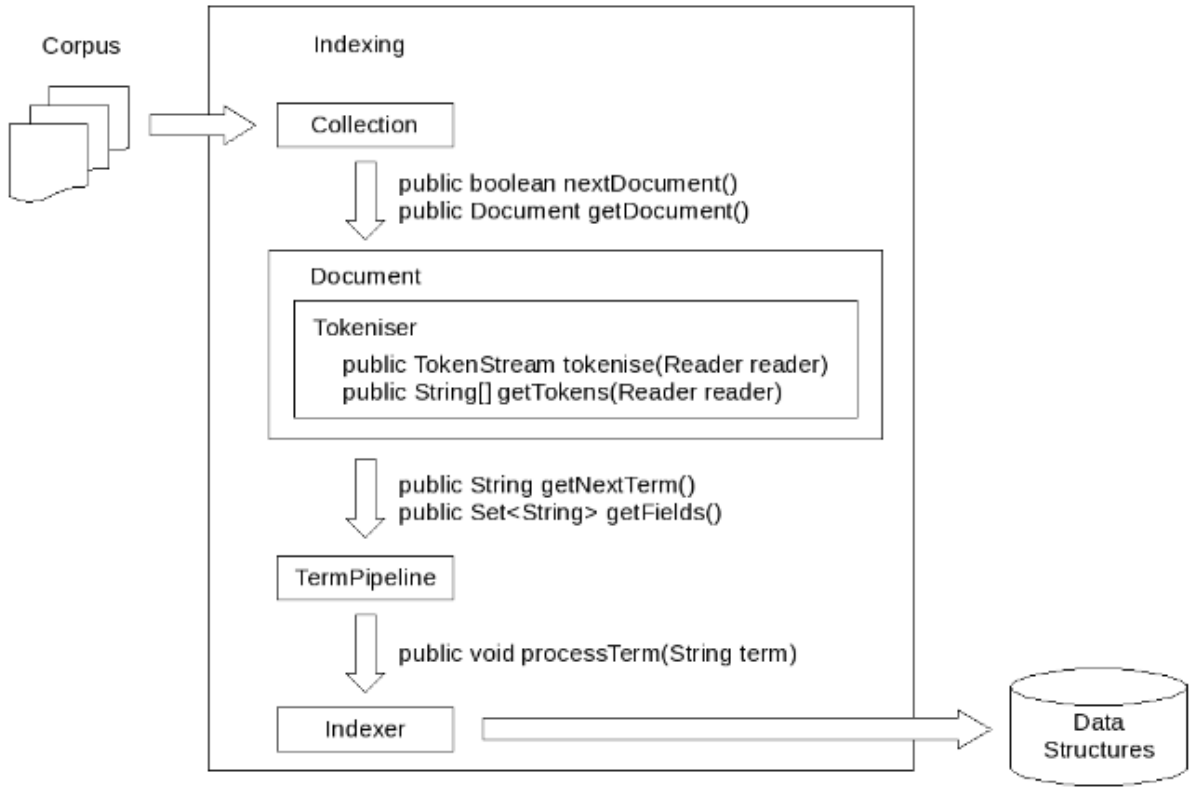


Figure 3 : Processus d'indexation de Terrier.

5.1.1 Collection:

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

- public Document getDocument();
- public boolean nextDocument();
- public String getDocid();
- public boolean endOfCollection();

Plusieurs Classes implémentent l'interface collection selon le format du document : SimpleFileCollection: PDF, TXT, HTML,….., TrecCollection, SimpleXMLCollection : XML,….etc.

5.1.2 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 boolean endOfDocument();

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

5.1.3 Term Pipeline :

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'algorithme de Porter (PorterStemmer).

5.1.4. Indexer :

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

Il existe deux types d'indexeur dans Terrier : *BasicIndexer, BlockIndexer*.

5.1.5. 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 (Informations sur 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.

5.2. L'API de recherche :

Après le processus d'indexation le processus de recherche est réalisé, la figure 4 présente le processus de recherche de Terrier.

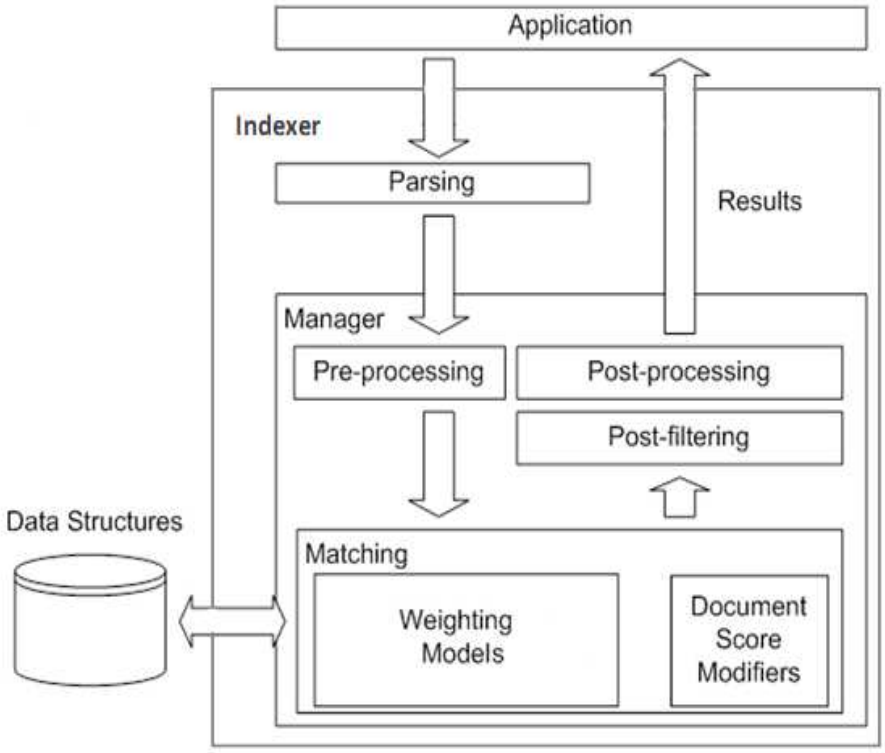


Figure 4 : Processus de recherche de Terrier.

2.5.2.1 Query

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

Terrier supporte Trois modèles de requête:

- 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).

5.2.2 Manager :

Ce composant est chargé de la gestion de la recherche il utilise quatre composants :

- **Pre-processing**: Il permet d'appliquer le Tokeniser et le TermPipeline 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.

5.2.3 Set-Results :

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

6. Les applications de Terrier :

Terrier est livré avec trois applications :

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

6.1. 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.

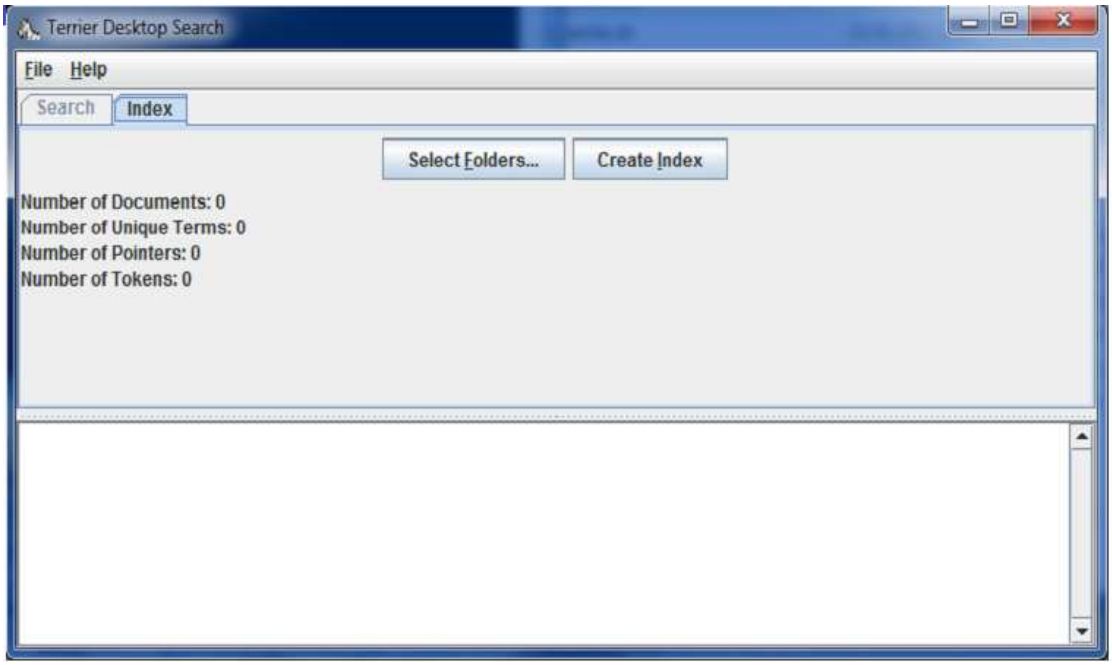


Figure 5: présentation de l'interface Desktop Terrier

Desktop Terrier propose deux onglets principaux :

- **Index :** Il Sélectionne le dossier qui contient les documents à indexer, puis il crée l'index. Il prend en charge que les documents de type: txt, text, pdf, MS, Word, MS Excel, MS PowerPoint, HTML, XML, XHTML, Tex. Les formats complexes ne sont pas traités avec Desktop Terrier (exemple: Collections Trec).
- **Search :** Il retourne les documents pertinents pour une requête saisie dans le champ texte à partir de l'index créé.

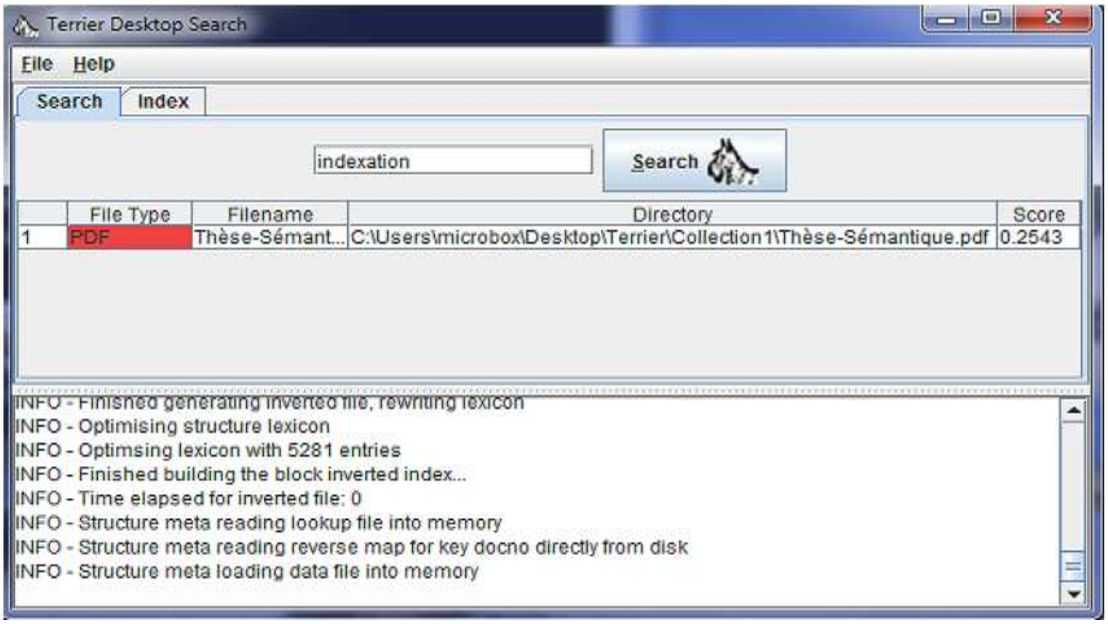


Figure 6 : Illustration du processus de recherche avec Desktop Terrier.

6.2. 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> (voir figure 7).

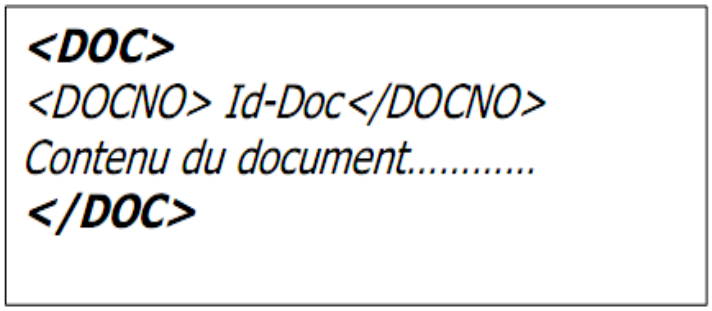


Figure 7 : Format d'un document Trec.

Le Trec (Batch) Terrier est exécuté en utilisant l'invité de commandes. C'est l'application la plus recommandé pour effectuer des travaux complexes de l'indexation et de la recherche.

6.2.1. Exécution de TREC Terrier sous Windows:

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

- **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.properties*** : 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.properties,* 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 lemmatiser :
- *termpipelines= WeakPorterStemmer*

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'invité 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	*rec_terrier --printstats*

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

➢ **Processus de recherche :**

1. Ajouter la 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 Parser d'analyser une requête simple par ligne :
  trec.topics.parser=SingleLineTRECQuery
- Configuration du modèle de pondération :
  Trec.model= TF_IDF (utiliser le modele 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-Ponderation*.

➢ **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.

6.3. 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.

7. Compilation de Terrier :

Terrier est extensible. En effet, on peut modifier son code source et lui intégrer de nouveaux modules (classes). Pour se 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.

- Configuration du Ant de IDE Eclipse :

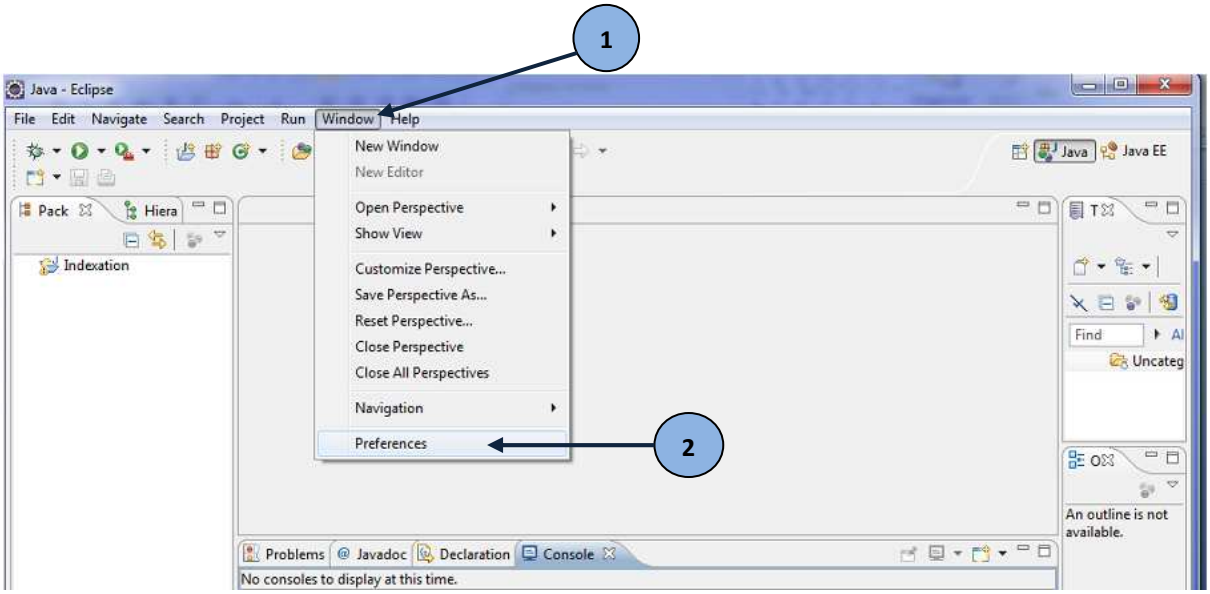


Figure 8: Configuration Ant : étape 1.

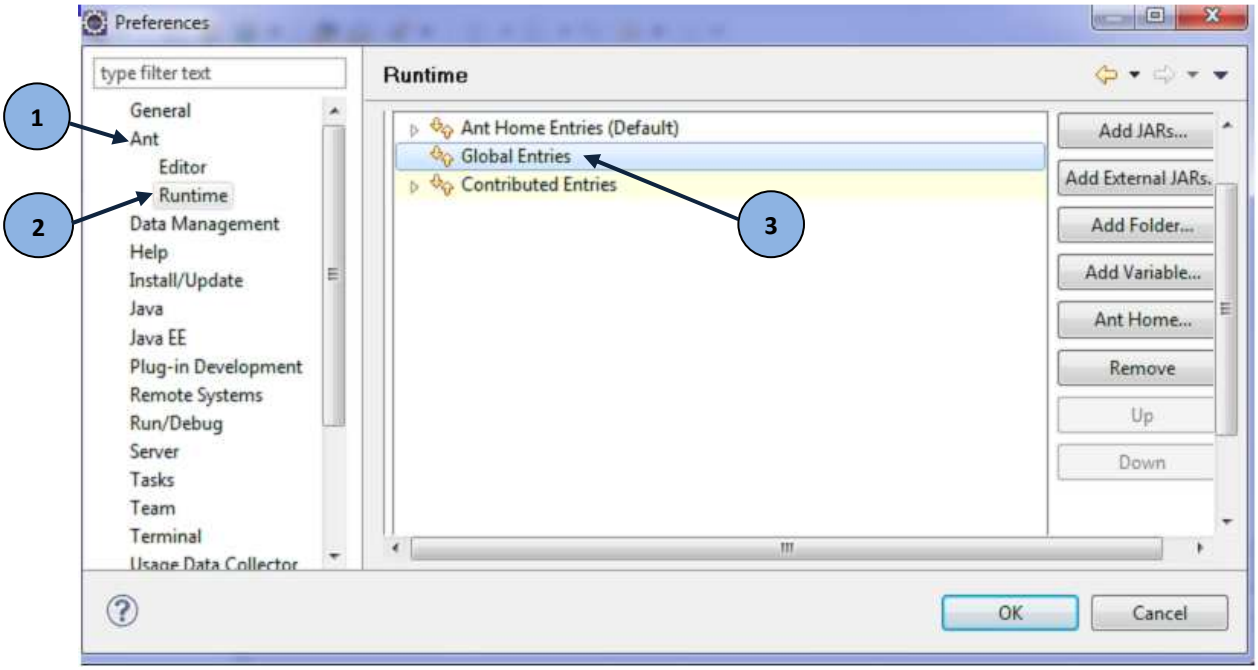


Figure 9 : Configuration Ant : étape 2.

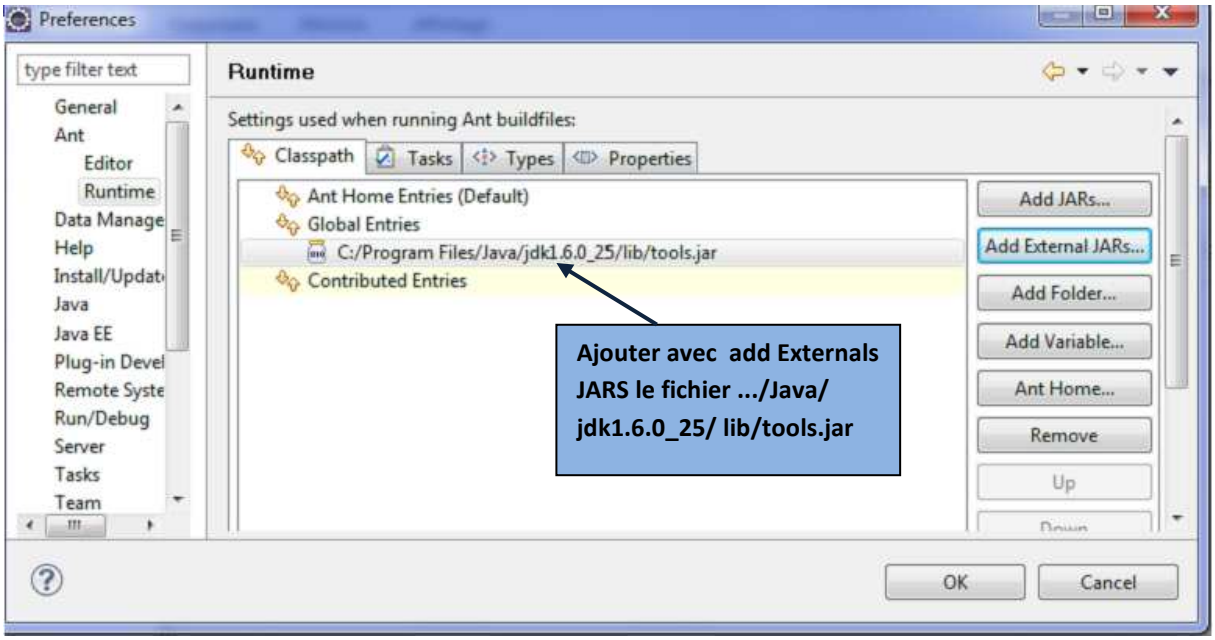


Figure 10 : Configuration Ant : étape 3.

– Exécution du fichier build.xml :

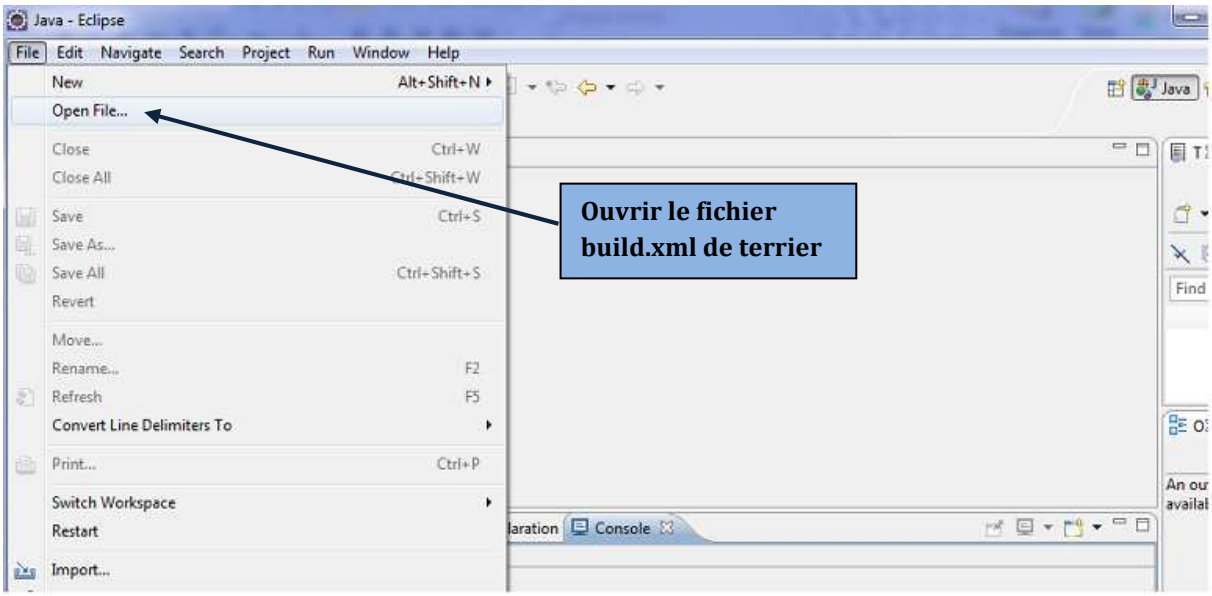


Figure 11 : Ouverture du fichier.

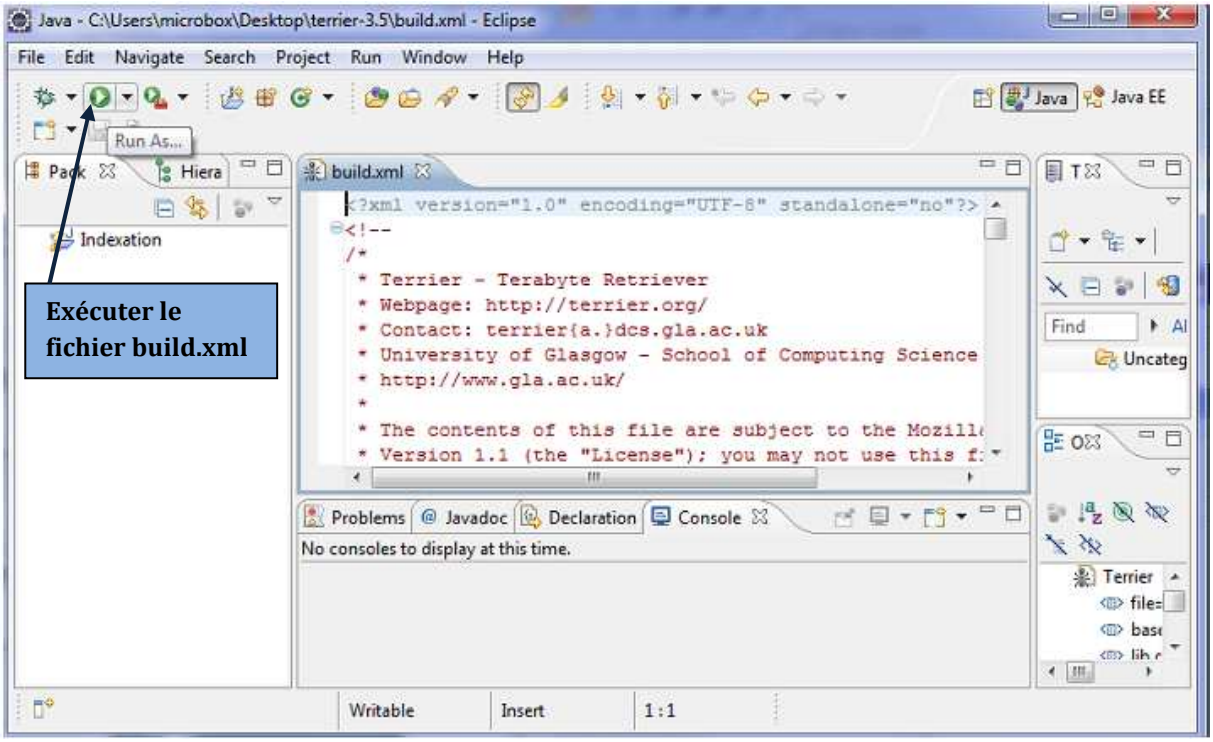


Figure 12 : Exécution du script.

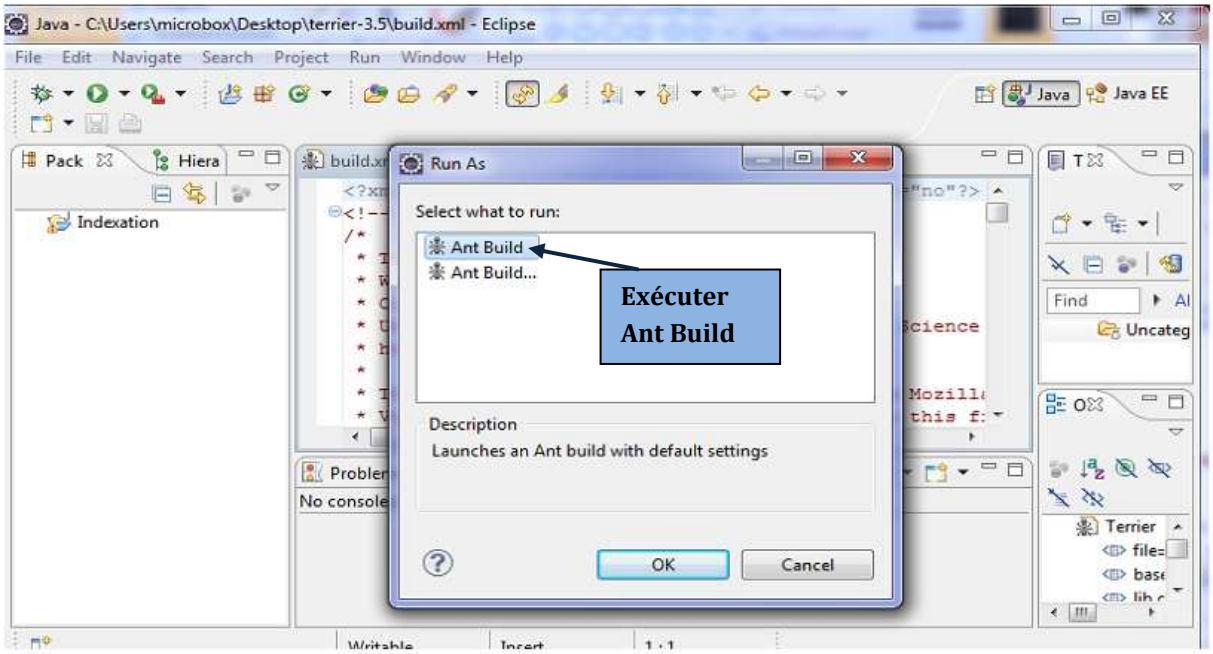


Figure 13 : Exécution du Ant Build.

– Résultats de la compilation :

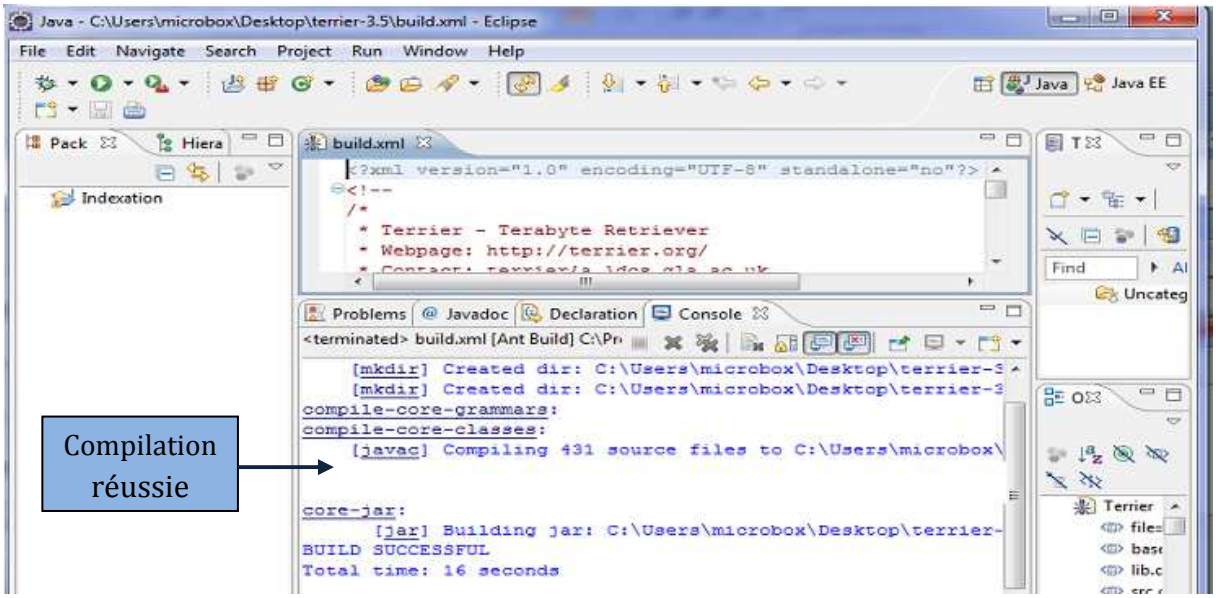


Figure 14 : Compilation réussie (BUILD SUCCESSFUL).

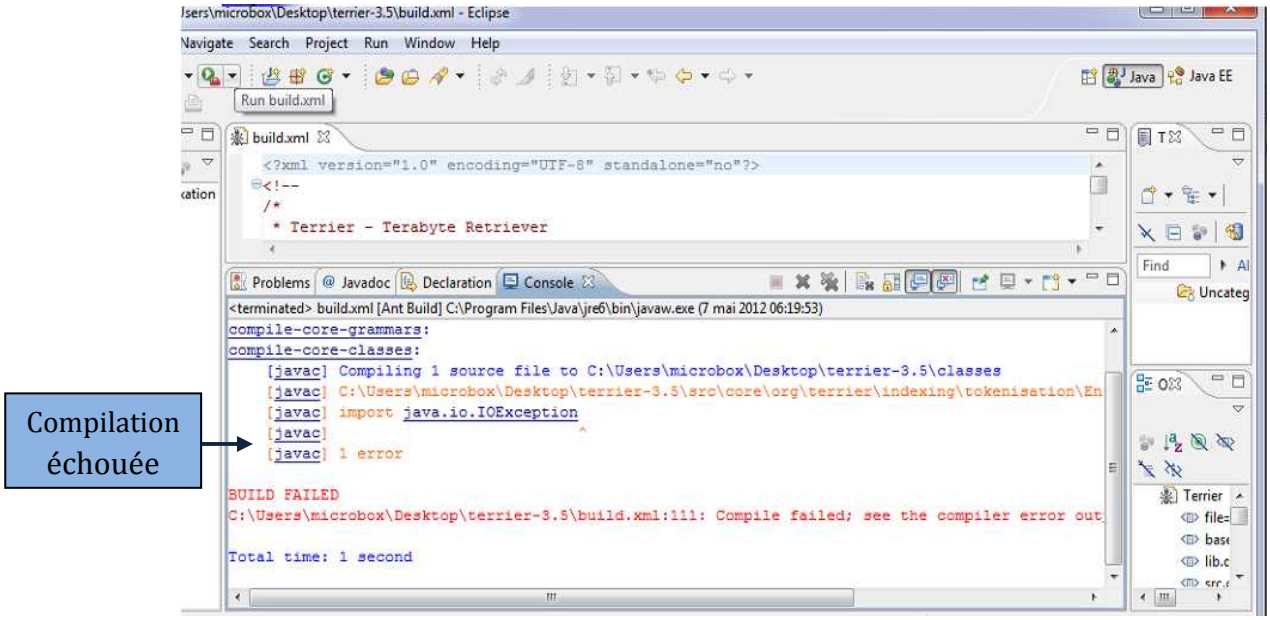


Figure 15 : Compilation échouée (BUILD FAILED).

Annexe III

Stanford POS Tagger

1. Introduction :

Cette annexe est consacrée à la présentation de Stanford POS Tagger, un outil utilisé dans notre approche d'indexation sémantique pour déterminer la catégorie syntaxique de chaque mot dans son contexte d'utilisation.

2. Présentation de Stanford POS Tagger :

Stanford POS Tagger (POS pour *Part-of-Speech*) a été développé par The Stanford Natural Language Processing Group, C'est un outil à double fonctionnalités il est considéré comme étant :

- **Un étiqueteur morphosyntaxique :** Il consiste à identifier pour chaque mot sa classe morpho-syntaxique (catégorie grammaticale) à partir de son contexte.
- **Un lemmatiseur :** désigne l'analyse lexicale consistant à associer une forme canonique (lemme) à chaque mot du texte. Si le mot ne peut pas être lemmatisé (nombre, mot étranger, mot inconnu), aucune information ne lui est associée. La lemmatisation suppose que l'analyse morpho-syntaxique a déjà été faite.

Stanford POS Tagger 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 1. Cette liste correspond aux jeux d'étiquetage *Treebank Tag set.*

Catégories	*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	Référence
MD	Auxiliaires (*can, should, may, would, will, might*)
NN	Nom
NNP	Nom propre
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$	Pronom possessif
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 ,adjectifs interrogatifs relatifs*
WP$	Whose
WPUNC′	Ponctuation faible
WRB	*How, when, whence, whenever, where, whereby, wherever, why*
ZTRM′	Fin de phrase

Tableau 1 : Liste des catégories grammaticales.