RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU
FACULTE DE GENIE ELECTRIQUE ET INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

MÉMOIRE

En vue de l'obtention du grade

Master en informatique

Spécialité : conduite de projets informatiques

Thème :

Implémentation d'une méthode de Clustering pour le calcul de la probabilité de pertinence a priori d'un document

Réalisé par :
M^{elle} AÎCHE Yasmina
M^{elle} KOULOUGLI Dalila

Soutenu le 7 juillet 2013 devant le jury composé de :

Mr. HAMMACHE Areski	Directeur de mémoire
Mme.OUKFIF Karima	Présidente du Jury
Mr.DJADEL Hacen	Examinateur du mémoire
Mr.SADOU Samir	Examinateur du mémoire

Année Universitaire : 2012/2013

PDF created with pdfFactory Pro trial version www.pdffactory.com

Remerciements

Nous tenons tout particulièrement à témoigner notre profonde gratitude et nos remerciements les plus sincères à M^{r} HAMMACHE Areski pour avoir accepté et proposé ce sujet qui nous a permis de creuser dans ce domaine qui nous tenait particulièrement à cœur, nous le remercions pour sa présence, sa disponibilité ainsi pour ces judicieux conseils et orientations, et son soutien et pour tout le temps qu'il a consacré au bon déroulement de ce travail.

Nos vifs remerciements vont aux membres du jury pour l'honneur qu'ils nous ont fait d'examiner et de juger notre mémoire.

Nous remercions ensuite tous les enseignants de notre cursus, en particulier ceux du département d'Informatique de l'université MOULOUD MAMERI TIZI-OUZOU.

Enfin, nous remercions tous ceux qui ont contribués de prés où de loin la réalisation de ce modeste travail.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Dédicaces

Je dédie ce modeste travail :

À mes très chers parents en reconnaissance de leurs soutiens tout au long de mes études.

À mes frères Djamel, Smail, Ghiles.

À mes tantes et oncles ainsi que toutes leurs familles

À toute les personnes que j'aime qui de près ou de loin m'ont accompagné dans mes études, sans oublié tout mes camarades de promotion

D. KOULOUGLI

PDF created with pdfFactory Pro trial version www.pdffactory.com

Dédicaces

Je dédie ce modeste travail :

À ma mère

À la mémoire de mon père

À ma sœur Abir et mon frère Nazim

À mes grands parents

À mes tantes et oncles ainsi que toutes leurs familles

À tout mes amis

Y. AICHE

PDF created with pdfFactory Pro trial version www.pdffactory.com

Table des matières

CHAPITRE1 : ETAT DE L'ART DE LA RECHERCHE D'INFORMATION

1. Introduction 1
2. Définition de la RI (Recherche d'Information) 1
3. Définition d'un SRI (Système de Recherche d'Information) 1
4. Concepts de base de la recherche d'information 1
   4.1. Le Document et la collection de documents 2
   4.2. La requête 3
   4.3. La pertinence 3
5. Le processus de Recherche d'Information 4
   5.1. Le processus d'indexation 4
      5.1.1. L'indexation manuelle 4
      5.1.2. L'indexation automatique 5
         5.1.2.1. L'analyse lexicale 5
         5.1.2.2. L'élimination des mots vides 5
         5.1.2.3. La lemmatisation 6
         5.1.2.4. La pondération 6
      5.1.3. L'indexation semi-automatique 7
   5.2. Le processus d'appariement 8
   5.3. Le processus de Reformulation de la requête 8
      5.3.1. Les méthodes locales 8
      5.3.2. Les méthodes globales 8
6. Modèles de la recherche d'information 9
   6.1. Les modèles reposant sur la théorie des ensembles 10
      6.1.1. Le modèle booléen de base 10
   6.2. Le modèle vectoriel 11
      6.2.1. Le modèle vectoriel de base *(Vector Space Model)* 11
   6.3. Le modèle probabiliste 13
      6.3.1. Le modèle de probabiliste de base 13
      6.3.2. Les modèles de langue 14

PDF created with pdfFactory Pro trial version www.pdffactory.com

7. Evaluation ses systèmes de recherche d'information 15
7.1. Les mesures d'évaluation 15
7.1.1. La précision et le Rappel 15
7.1.2. La courbe rappel – précision 16
7.2. Mesures alternatives 18
7.2.1. La précision à X documents 18
7.2.2. La précision exacte ou la R-précision 18
7.1.3. La précision moyenne 18
7.1.4. F-mesure 18
7.3. Les compagnes d'évaluation et les collections de référence 19
6.2.1. Les compagnes TREC 19
8. Conclusion 19

CHAPITRE 2 : Modèle de langue pour la RI

1. Introduction 20
2. Modèle de langue en linguistique informatique 20
2.1 Idée de base 20
2.2. Lissage 22
3. Modèle de langue en RI 24
3.1 RI comme génération de la requête par le document 26
3.1.1. Modèle de Pont Croft 26
3.1.2. Modèle de Hiemstra et al. Est de Miller et al. 27
3.2. Génération de documents par le modèle de la requête 29
3.3. Comparaison entre le modèle de requête et le modèle de document 31
4. Prise en compte des informations à priori dans les modèles de langues 33
4.1. Formules Générale 33
4.2. Information à priori 34
4.2.1. Taille du document 34
4.2.2. La structure des liens 34

PDF created with pdfFactory Pro trial version www.pdffactory.com

4.2.2.1. Algorithme HITS (Hyperlinked Induced Topic Search) 34
4.2.2.2. L'Algorithme PageRank 36
4.2.2.3. Rapport Information/Bruit 37
4.2.2.4. Type d'URL du document 37
5. Conclusion 38

CHAPITRE3 : Clustering

1. Introduction 39
2. L'apprentissage Automatique 40
2.1. Apprentissage supervisé 40
2.2 Apprentissage semi-supervisé 41
2.3 Apprentissage non supervisé 42
3. Quelques bonnes raisons de s'intéresser à l'apprentissage non supervisé 42
4. Les différentes approches à la classification non supervisé 42
4.1. Classification par partition 42
4.2. Classifications hiérarchiques 43
5. Clustering 43
5.1. Définition d'un Clustering 43
5.2. But de Clustering 44
5.3. L'évaluation d'un système de clustering 44
5.3.1. Pureté 45
5.3.2. Entropie 45
5.3.3. Information mutuelle 45
5.4. Applications possibles 45
5.5. Types de données pour l'analyse de clusters 46
5.6. Notion de similarité 46
5.6.1. Vocabulaire 46
5.6.2. Notations 46
5.6.3. Mesure de la distance d(x1, x2) entre 2 points x1 et x2 47
5.6.4. Mesure de la distance D (C1, C2) entre 2 classes C1 et C2 47
5.7. Mesures de similarités 48
5.7.1. Attributs numériques 48

PDF created with pdfFactory Pro trial version www.pdffactory.com

5.7.2. Attributs catégoriels 48
5.8.. Méthodes de clustering 49
5.8.1. Méthode Hiérarchiques 50
5.8.1.1. Classification hiérarchique Ascendante (CHA) 51
5.8.1.1.1. Les critères d'agrégations 51
5.8.1.1.2. L'Algorithme 52
5.8.1.1.3. Exemple 52
5.8.1.1.4. Méthode Chameleon 53
5.8.1.2. Classification Hiérarchique Descendant : (CHD) 54
5.8.1.3. Avantages et Inconvénients de la classification hiérarchique 55
5.8.2. Classification par partitionnement 55
5.8.2.1. Méthode K-means 55
5.8.2.1.1. L'Algorithme de K-means s'exécute en 5 étapes 55
5.8.2.1.2. Organigramme de l'algorithme de k-means 56
5.8.2.1.3. K-means: illustration 56
5.8.2.1.4. Les avantages et les inconvénients 58
5.8.3. Classement par modélisation (Modélisation statique) 59
5.8.3.1. Notion de modèles de mélanges 59
5.8.3.2. L'Algorithme EM 60
5.8.3.3. Remarque sur l'Algorithme EM 61
5.8.4. Algorithme basé sur la densité 62
5.8.5. Algorithmes sur une grille 62
6. Conclusion 63

Chapitre 4 : Implémentation de l'approche

1. Introduction 64
2. Description de l'approche implémentée 64
2.1. Hypothèse de l'approche 64
2.2. Fonction d'appariement utilisée 64
3. Exemple illustratif : 67
4. Présentation de l'architecture du système développé 72

PDF created with pdfFactory Pro trial version www.pdffactory.com

5. Description des collections de tests utilisées 72
6. L'environnement de développement 73
7. Conclusion 82

Bibliographie 63

Chapitre4 : Implémentation de l'approche

1. Introduction 63
2. Description de l'approche implémentée 63
   2.1 hypothèse de l'approche 63
   2.2 fonction d'appariement utilisée 63
3. Présentation de l'architecture du système 68
   3.1 Présentation de la plate forme Terrier 69
      3.1.1 Processus d'indexation 69
      3.1.2 Construction d'index 69
      3.1.3 La recherche dans Terrier 70
      3.1.4 Le langage de programmation java 71
      3.1.5 Présentation de NetBeans 72
   3.2 Présentation de la plate forme Lemur 4.12 73
4. Description des collections de tests utilisées 73
5. Tests et résultats de l'approche 75
   5.1 L'indexation avec lemur 75
   5.2 Le fichier obtenue 76
6. Conclusion 77

PDF created with pdfFactory Pro trial version www.pdffactory.com

Table des figures

Figure 1.1 : Le processus U de la recherche d'information.

Figure 1.3: Le modèle vectoriel

Figure 1.2: Taxonomie des modèles de RI.

Figure 1.4: Partition de la collection pour une requête

Figure 1.5: Courbe de Rappel-Précision de la requête R1

Figure 2.1: Les différentes techniques de lissage.

Figure 2.2: Modèle de Markov caché à deux états

Figure 2.3: Exemple du modèle de Relevance.

Figure 2.4: Répartition des pages en bon Hubs, bonnes autorités et pages indépendantes.

Figure 3.1 : les types d'apprentissage automatique.

Figure 3.2: Classification hiérarchique.

Figure 3.3: exemple de CHA.

Figure 3.4: Résultat de l'algorithme de Chameleon.

Figure 3.5 : Organigramme de l'algorithme de k-means

Figure 3.6: Exemple d'illustration de [S.Sankararam].

Figure 3.7: Présentation d'une grille

PDF created with pdfFactory Pro trial version www.pdffactory.com

Table des tableaux

Tableau 1.1: Exemple de calcul de rappel et précision pour les requêtes R1 et R2

Table 2.1 : Exemple d'estimation de vraisemblance maximale.

Tableau 3 : Tableau de comparaison avec K-means.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Introduction Générale

La recherche d'information (RI), est une branche de l'informatique qui s'intéresse à l'acquisition, et 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.

La mise en œuvre de la recherche d'information est effectuée grâce à un système de recherche d'information (SRI). C'est un ensemble de logiciel qui possède trois fonctions fondamentales : représenter le contenu des documents grâce à un processus d'indexation, représenter le besoin en informations à l'utilisateur et comparer ces deux représentations grâce a un processus d'appariement. A la suite de cette comparaison, le SRI retourne à l'utilisateur une liste de documents qui est généralement ordonnée de façon à ce que les documents les plus susceptibles d'intéresser l'utilisateur apparaissent en première position.

Le problème qui se pose actuellement n'est plus la disponibilité de l'information, mais la capacité d'accès et de sélection de l'information répondant aux besoins précis d'un utilisateur.

Un document n'est pas caractérisé uniquement par son contenu textuel. D'autres informations peuvent être exploité sur un document, afin de calculer la probabilité a priori de pertinence du document on combinera cette dernière avec le score obtenu par le contenu textuel dont l'objectif de calculer la pertinence finale du document, dont le but d'améliorer les performances de recherche d'information, parmi ces informations on trouve la structure des liens (nombre des liens entrant) la taille du document etc.

Dans notre cas nous explorons l'introduction d'un nouveau facteur pour le calcul de pertinence à priori qui est basé sur la notion de Clustering. Plus spécifiquement on assignera une plus grande probabilité de pertinence pour un document qui est éloigné des différents clusters.

Cette idée est basée sur l'intuition suivante : « un document qui traite beaucoup de thématique tend à être proche des différents clusters. Donc on peut le qualifier du document générique par conséquent il est a priori moins pertinent.

Notre mémoire est structuré de la manière suivante :

Chapitre1 : Qui porte sur les concepts de base de la RI. Dans ce chapitre, on présentera les notions de documents, requête et de pertinence, ensuite on parlera du processus de recherche d'information et des différents modèles de RI et enfin de l'évaluation des SRI.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Chapitre2 : qui est consacré au modèle de langue. D'abord on présentera l'idée de base de ces modèles, ensuite les différentes catégories de ces modèles et enfin nous verrons l'intégration des informations a priori de pertinence dans les modèles de langue.

Chapitre3 : ce chapitre sera consacré à la technique de Clustering et aux différentes méthodes qui existent illustré par des exemples nous présenterons un algorithme de k-means qui permet de regrouper les documents dans un ensemble des clusters homogènes.

Chapitre4 : dans ce dernière chapitre, nous abordons l'approche implémentée, qui consiste dans notre cas à implémenter d'une méthode de Clustering pour le calcul de la probabilité de pertinence à priori d'un document.

Pour finir, nous présentons deux plates formes Terrier 2.1 et Lemur 4.12

PDF created with pdfFactory Pro trial version www.pdffactory.com

1. Introduction

La croissance très importante des informations disponibles sur Internet nécessite des outils de recherche de plus en plus performants et permettant de discerner efficacement les informations pertinentes parmi des centaines voire des milliers de documents [Amati & Crestani, 99], [Lagus et al., 96], [Fuhr & Buckley, 91] .

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 [Salton et al., 86] intègrent un ensemble de techniques et d'outils (programmes informatiques) permettant de sélectionner dans une collection de documents ceux qui sont susceptibles de répondre aux besoins en information de l'utilisateur.

Dans ce premier chapitre a pour but de présenter les principaux concepts de la RI. Nous commençons tout d'abord par donner quelques définitions, puis nous décrivons les étapes d'un processus de RI, Nous passons ensuite en revue les différents modèles de recherche. Enfin, nous présentons les plus importants critères d'évaluation d'un SRI.

2. Définition de la RI (Recherche d'Information)

La recherche d'information (RI) [Mooers, 48] est une discipline ancienne, elle remonte aux années 50.

La Recherche d'Information (RI) [Rijsbergen, 79] [Grossman et al., 98] [Salton, 71] [Baeza-Yates et al., 99] [G.Salton, 1970] [81] [Grossman et al.,98] [Yates et al.,99] est traditionnellement définie comme l'ensemble des techniques et de méthodes et de procédures permettant de sélectionner à partir d'une collection de documents, ceux qui sont susceptibles de répondre aux besoins de l'utilisateur. Dans un sens plus large, la RI est toute opération (ou ensemble d'opérations) ayant pour l'objectif la recherche, la collecte et l'exploitation d'informations en réponse à une question sur un sujet précis.

3. Définition d'un SRI (Système de Recherche d'Information)

Un SRI [Salton et al., 86] est un ensemble de programmes informatiques qui a pour but de sélectionner des informations pertinents répondant aux besoins des utilisateurs, exprimées sous forme de requête.

Un système de recherche d'information, intègre un ensemble de modèles pour la représentation des unités d'informations (documents et requêtes). Il intègre également un mécanisme de recherche/sélection. Ce dernier permet de sélectionner l'information pertinente en réponse aux besoins exprimes par l'utilisateur a l'aide d'une requête.

PDF created with pdfFactory Pro trial version www.pdffactory.com

4. Concepts de base de la recherche d'information

Afin de répondre aux informations voulues par l'utilisateur, un SRI met en œuvre un certain nombre de processus pour réaliser la mise en correspondance des informations contenues dans un fond documentaires d'une part, et des besoins en informations des utilisateurs d'autre part.

Le processus de recherche [Belkin et al 1992], couramment appelé Processus en U de recherche d'information est schématiquement représenté sur la figure 1.1

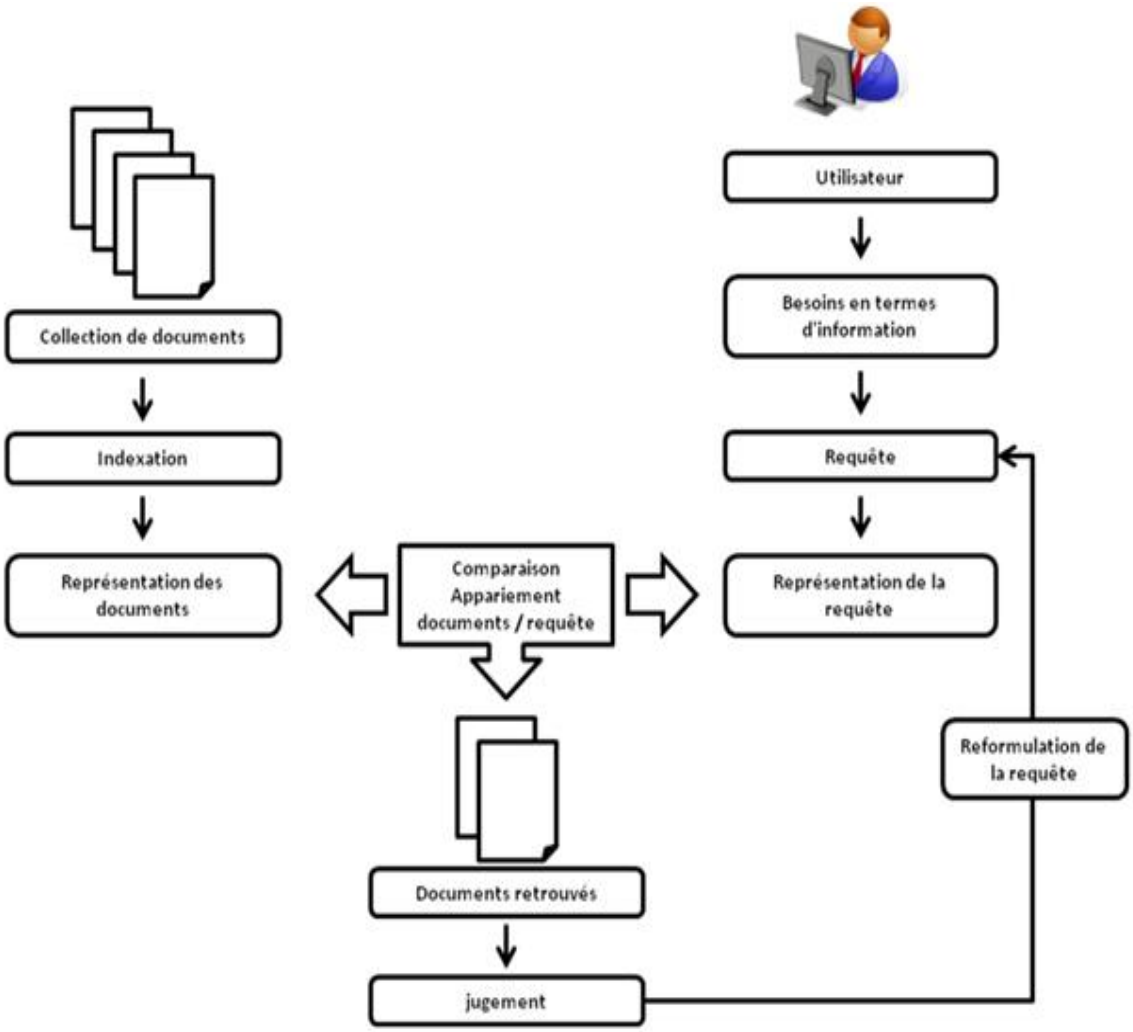


Figure 1.1 : Le processus U de la recherche d'information.

La figure 1.1 illustre l'architecture générale d'un système de recherche d'information.
Plusieurs éléments clés y sont distingues :

- les documents et la collection de documents
- le besoin en information (requête),
- la pertinence,

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

Plusieurs concepts clés s'articulent autour de cette définition :

PDF created with pdfFactory Pro trial version www.pdffactory.com

4.1. Le Document et la collection de documents

Le Document est un élément central du SRI, c'est un objet complexe sans cesse en évolution car lié au développement des technologies de communication. Un document peut être un texte, une page Web, une image, une bande vidéo, etc. [Belkin et al., 97]

Le document constitue l'information élémentaire d'une collection de documents. L'information élémentaire, appelée aussi granule de document, peut représenter tout ou une partie d'un document.

Un document représente dans un SRI l'élément objectif [Saracevic 96], dans le cadre de la RI traditionnelle, c'est du texte libre, qui peut être caractérisé selon trois vues :

- Ø la vue représentation, c'est la mise en forme d'un document texte (entêtes, paragraphes, alignement. . .) ;
- Ø la vue logique qui présente la structure logique d'un document, elle porte des informations sur la structure ;
- Ø la vue de contenu, qui se focalise sur le sens ou la sémantique d'un document le plus souvent par un ensemble de mots.

La collection de documents (ou fond documentaire, corpus) constitue l'ensemble des informations exploitables et accessibles. Elle est constituée d'un ensemble de documents.

4.2. La requête

Un besoin en information d'un utilisateur est exprimé par une requête.

Pour exprimer son besoin l'utilisateur compose une suite de mots-clés (requête), souvent séparés par des opérateurs logiques (et, ou et non), ou par des variables linguistiques telles que proche de, contient. . ..etc.

On distingue deux types de requêtes :

- ü les requêtes basiques : la requête est un ensemble de mots-clés ;
- ü les requêtes logiques (booléennes) : la base des requêtes est un ensemble d'opérateurs logiques (et, ou et non) ;

4.3. La pertinence

La pertinence est une notion fondamentale et cruciale dans le domaine de la RI car toutes les évaluations s'articulent autour de cette notion, c'est aussi la notion qui constitue le cœur du SRI. Elle est l'objet de tout système de recherche d'information.

Elle a été sujette aux nombreuses études au cours du développement de la RI et plusieurs définitions lui ont été attribuée :

- la correspondance entre un document et une requête ;
- mesure d'informativité du document à la requête ;
- le degré de relation (chevauchement, relativité,…) entre le document et la requête ;

PDF created with pdfFactory Pro trial version www.pdffactory.com

- le degré de la surprise qu'apporte un document, qui a un rapport avec le besoin de l'utilisateur.

Les travaux de recherche [Saracevic 75], [Mizzaro 97], s'accordent sur la difficulté de la définition de la pertinence et mettent en exergue deux types de pertinence : la pertinence système et la pertinence utilisateur.

- **ü** la pertinence système [CLEVERDON 70] est déterministe, objective et elle est définie à travers les modèles de RI. Elle est souvent traduite par un score évaluant l'adéquation du contenu des documents vis-à-vis de celui de la requête ;
- **ü** la pertinence utilisateur [Harter 92], [Saracevic 96] [Mizzaro 97] quant à elle, est liée à la perception de l'utilisateur sur l'information renvoyée par le système. Elle est subjective, deux utilisateurs peuvent juger différemment un même document renvoyé pour une même requête. Elle peut évoluer dans le temps d'une recherche. Une information jugée non pertinente à l'instant t pour une requête peut être jugée pertinente à $t + 1$ car la connaissance de l'utilisateur sur le sujet à évolué.

5. Le processus de Recherche d'Information

Pour répondre aux besoins en information de l'utilisateur, un SRI met en œuvre un certain nombre de processus pour réaliser la mise en correspondance des informations contenues dans un fonds documentaire d'une part, et les besoins en information des utilisateurs d'autre part.

Ce processus est composé de trois fonctions principales : l'indexation des documents et des requêtes, l'appariement requête-document ou l'interrogation et la reformulation de la requête.

5.1. Le processus d'indexation

Afin d'assurer la recherche dans des conditions acceptables de coût et d'efficacité, une étape primordiale doit s'effectuer avant l'étape de recherche effective de l'information. Cette étape consiste à analyser le document lors de l'organisation du fond documentaire afin de produire un ensemble de mots clés, appelés aussi *descripteurs*, que le système pourra gérer aisément puis utiliser dans le processus de recherche ultérieur. Cette opération est appelée *indexation* [Salton, 71][SparkJones, 79][Rijsbergen, 79][Deerwester et al., 90][Soule Dupuy, 90]. Cet ensemble de mots clés peut être regroupé dans un thésaurus [Crouch et al., 89] [Crouch et al., 92] [Frakes et al., 92], mais en pratique, un thesaurus représente une notion plus large qu'une liste de mots clés. Il regroupe plusieurs relations de types linguistique (équivalence, association, hiérarchie) et statistique (pondération).

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

PDF created with pdfFactory Pro trial version www.pdffactory.com

5.1.1. L'indexation manuelle

Chaque document est analysé par un spécialiste du domaine ou par un documentaliste, qui se charge de recenser selon ses connaissances propres, les concepts dont traite un document et de les présenter à l'aide d'un langage libre ou contrôlé.

L'indexation manuelle à l'avantage d'assurer une meilleure correspondance entre les documents et les termes d'indexation choisis. Ceci a pour conséquence une meilleure précision dans les documents que le SRI retourne en réponses aux requêtes des utilisateurs [Nie et al., 99].

L'inconvénient majeur de cette méthode est l'effort intellectuel qu'elle exige (en nombre de personnes). De plus, un degré de subjectivité lié au facteur humain fait que pour un même document, des termes différents peuvent être sélectionnés par des indexeurs différents. Il peut même arrivé qu'une même personne, à des moments différents, indexe différemment le même document.

5.1.2. L'indexation automatique

L'indexation automatique est la technique d'indexation la plus utilisée lors d'un processus de RI. Elle offre l'avantage d'être totalement automatisée et donc plus adaptée pour les bases documentaires de grandes tailles.

Elle repose sur un ensemble des méthodes automatisées sur un document comme l'analyse lexical, l'élimination des mots vides, la lemmatisation (radicalisation), la pondération des termes et la création de l'index.

5.1.2.1. L'analyse lexicale

L'analyse lexicale est le processus qui permet de convertir le texte d'un document en un ensemble de termes. Un terme est une unité lexicale ou un radical [Fox 92]. L'analyse lexicale permet de reconnaître les espaces de séparation des mots, des chiffres, les ponctuations, etc.

5.1.2.2. L'élimination des mots vides

Un des problèmes majeurs de l'indexation consiste à extraire les termes significatifs et à éviter les mots vides (les articles, les propositions et les conjonctions, pronoms personnels, prépositions,...). Ce sont des mots qui ne sont pas discriminatoires lors d'une recherche, car ils ne traitent pas du sujet du document et sont présents en grand nombre dans les documents de la collection.

On distingue deux techniques pour éliminer les mots vides :

- L'utilisation d'une liste de mots vides (aussi appelée anti-dictionnaire ou stop-list) : 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 & Buckley, 1988].

PDF created with pdfFactory Pro trial version www.pdffactory.com

- 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.

Même si l'´elimination des mots vides a l'avantage évident de réduire le nombre de termes d'indexation, elle peut cependant réduire le taux de rappel, c'est à dire la proportion de documents pertinents renvoyés par le système par rapport à l'ensemble des documents pertinents.

5.1.2.3. La lemmatisation

L'objectif de la lemmatisation (radicalisation) est de rendre l'ensemble des formes des mots de la même famille représentées par un seul mot pour toute la famille. C'est la forme commune entre eux, qui est la forme de base (radical, par exemple flexibl pour flexible, flexiblement, flexibilité. . .).

Bien que le passage à la forme canonique présente un avantage, puisqu'elle permet d'indexer en un seul mot (lemme ou radical) sa famille morphologique, cette opération supprime la sémantique originale. Mais ceci ne présente aucun inconvénient puisque l'indexation passe totalement inaperçue par l'utilisateur : il s'agit d'un moyen de codage de l'information sans perte.

Les plus courants algorithmes pour la radicalisation des mots de la langue anglaise est l'algorithme de Porter [Porter 80]. La méthode de radicalisation de Porter permet de construire progressivement le radical en inférant les modifications grammaticales potentielles qui se manifestent le plus souvent par des postfixes ou préfixes. En français [Adamson et al., 74], il n'existe pas de l'algorithme fiable pour lemmatisation vu, par exemple, la complexité des formes prises par les verbes du troisième groupe (le verbe être peut prendre les formes : est, sommes, fûmes..).

5.1.2.4. La pondération

La pondération consiste à affecter à chaque terme t d'un document d un poids w. Ce poids exprime le degré de représentativité du terme dans le document.

La pondération d'un terme s'exprime en fonction de deux pondérations : une pondération locale et une pondération globale. La pondération locale reflète l'importance locale du terme dans le document. La pondération globale exprime l'importance globale du terme dans la collection.

La pondération d'un terme t dans un document d est souvent notée $TF * IDF$. Elle est donnée par le produit de la pondération locale de ti dans dj par sa pondération globale dans la collection des documents.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Ø La pondération locale tf

La pondération locale d'un terme t dans un document d, est une fonction de fréquence de ce terme dans le document d. La fréquence d'un terme représente le nombre d'occurrences de ce terme dans un document. Cette pondération est notée souvent tf. Dans la littérature on trouve de nombreuses formules de la pondération locale. Nous citons en particulier [Rorbertson et al., 1997] [Singhal et al., 1997]:

$$tf = f(t, d) \quad (1.1)$$

$$tf = 1 + \log(f(t, d)) \quad (1.2)$$

$$tf = 0.5 + 0.5 * \frac{f(t,d)}{\max t \in d\ (f(t,d))} \quad (1.3)$$

Où (t, d) est la fréquence du terme t dans le document d.

La formule 1.3 s'applique au document long. Elle permet d'atténuer l'effet des fréquences très élevées. Dans cette formule la fréquence de chaque terme dans le document est normalisée par la valeur maximale des fréquences des termes d'indexation.

Ø La pondération globale IDF

La pondération globale se base sur l'idée que si un terme se distribue d'une manière uniforme dans tous les documents de la collection, ce terme ne permet pas de distinguer les documents les uns des autres. Ce terme ne possède pas de pouvoir discriminant et on doit lui affecter une pondération faible. A l'opposé de ces termes, les termes qui apparaissent dans peu de documents sont utiles pour la discrimination et une pondération importante est alors attribuée à ces termes discriminants.

La pondération globale d'un terme est exprimée en fonction du nombre total de documents dans le corpus et en fonction du nombre de documents contenant ce terme. Cette pondération est notée IDF (Inverse Document Frequency). Dans la littérature plusieurs formules de calcul de cette mesure ont été proposées, dont nous citons [Rorbertson et al., 1997] [Singhal et al., 1997]:

$$IDF = \log\left(\frac{n}{N}\right) \quad (1.4)$$

$$IDF = \log\left(\frac{N-n}{N}\right) \quad (1.5)$$

Où

n est le nombre de documents où le terme t apparait

N est le nombre total de documents dans le corpus.

De ce fait, par cette double pondération locale et globale, les fonctions de pondérations sont souvent référencées sous le nom de TF * IDF.

Ces deux mesures (tf et IDF) sont combinées en tf * IDF pour fournir le poids du terme t ce produit représente une bonne approximation du poids du terme, particulièrement si les documents de la collection sont de taille (longueur) homogène cependant, si la collection

PDF created with pdfFactory Pro trial version www.pdffactory.com

contient des documents très hétérogènes en termes de longueur, les documents les plus longs se voient favorisés (car plus un document est long, plus un terme a de chance d'apparaitre). Pour remédier à cela, un facteur de normalisation peut être utilisé [Singhal et al.,96] [Robertson et al.,97].

5.1.3. L'indexation semi-automatique

Le processus d'indexation se fait en premier lieu d'une manière automatique, le documentaliste intervient seulement pour ajouter des mots-clés qu'il trouve intéressants pour représenter un document.

Le choix final revient au spécialiste ou au documentaliste, qui intervient souvent pour établir des relations sémantiques entre mots-clés et choisir les termes significatifs (Synonymes….etc) à partir du thésaurus ou d'ontologie [MdG91]. Cette méthode est une combinaison des deux méthodes précédentes.

5.2. Le processus d'appariement

Le processus d'appariement consiste à comparer la représentation de la requête avec les représentations des documents. Il calcule pour chaque couple (requête, document), une mesure appelée pertinence système qui reflète le degré de similarité entre la requête et le document considéré.

Le processus d'appariement se base sur une fonction de similarité (ou de correspondance) notée RSV (d,q) (Retrieval Status Value), où *d* est un document et *q* est une requête.

Traditionnellement le système de recherche retourne à l'utilisateur une liste de documents classés par RSV. Cette fonction est différente d'un modèle de recherche d'information à un autre. D'ailleurs un modèle de RI est caractérisé par sa fonction de similarité et son modèle d'indexation.

Selon [Ricardo et al., 1999], plusieurs modèles de recherche d'information ont été proposés. Nous citons dans la section 6 les principaux modèles.

5.3. Le processus de Reformulation de la requête

La reformulation de requête consiste, à partir d'une requête initiale formulée par l'utilisateur, à construire une requête qui répond mieux à son besoin informationnel. Ce processus a pour but d'améliorer la performance du système en offrant un mécanisme de raffinement de la requête utilisateur. Les techniques de reformulation de requête se classifient en méthodes locales et méthodes globales [Boubekeur, 2008].

5.3.1. Les méthodes locales

La méthode locale, nommée aussi réinjection de pertinence ou "relevance feedback" en anglais est la méthode la plus utilisée [Boughanem et al., 1999]. Elle consiste à présenter à l'utilisateur les documents similaires à la requête initiale.

PDF created with pdfFactory Pro trial version www.pdffactory.com

L'utilisateur sélectionne à partir de cette liste les documents pertinents. La requête initiale sera enrichie par les termes d'indexation des documents sélectionnés et la pondération des termes sera ajustée [Hlaoua, 2007].

5.3.2. Les méthodes globales

La méthode globale, se base sur l'utilisation des informations externes pour reformuler la requête initiale. Cette méthode consiste à enrichir la requête par des termes d'indexation initialement absents. En général, ces termes sont issus d'une ressource externe telle que, les bases lexicales, les thésaurus et les ontologies.

La ressource définit des relations sémantiques entre des termes d'indexation, ces relations sont exploitées pour sélectionner des termes qui peuvent être ajoutés à la requête. Parmi ces relations on peut citer la synonymie, l'équivalence, l'hyperonymie, la spécification/ généralisation, etc.

6. Modèles de la recherche d'information

La première fonction d'un système de recherche d'information est de mesurer la pertinence d'un document vis-à-vis d'une requête. Un modèle de RI a pour rôle de fournir une formalisation du processus de recherche d'information. Il doit accomplir deux rôles [Mammeri ,09] [K. Amrouche, 2008]:

- Créer une représentation interne pour un document ou une requête basée sur les termes de l'indexation.
- Définir une méthode de comparaison entre une représentation de document et une représentation de requête afin de déterminer leur degré de similarité (mesure de pertinence).

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

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

La figure 1.2 présente une classification des principaux modèles utilisés en recherche d'information [Baeza-Yates & al., 99].

PDF created with pdfFactory Pro trial version www.pdffactory.com

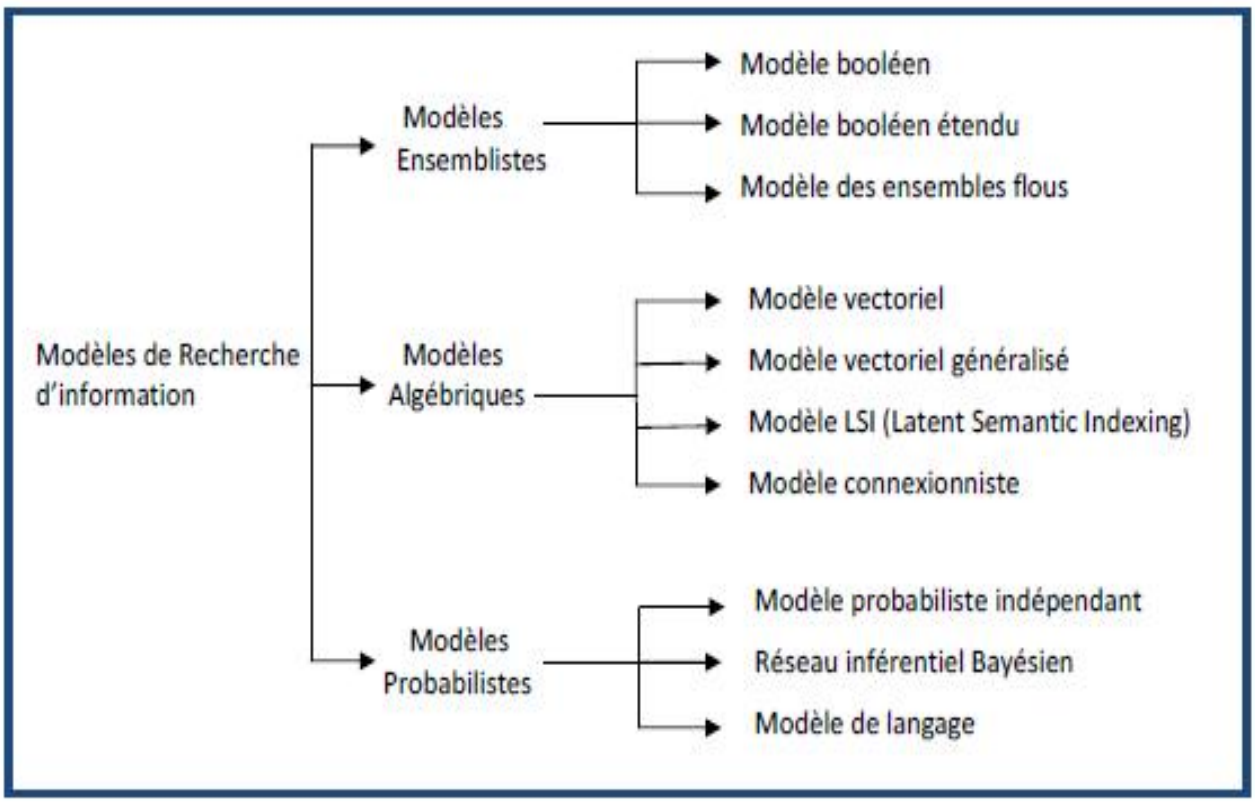


Figure 1.2 : Taxonomie des modèles de RI.

6.1. Les modèles reposant sur la théorie des ensembles

6.1.1. Le modèle booléen de base

Le modèle booléen [Cooper 70] est le premier modèle utilisé en RI. Dans ce modèle chaque document *d* est représenté comme une conjonction logique de termes (non pondérés) qui le composent, par exemple :

$$d = \{t1, t2, \ldots, tn\}.$$

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

q = (t1 and t2) ou (t3 et (not t4))

La correspondance *RSV (dj,qk)* entre une requête *qk* et un document *dj* , (tel que *qi* est un terme de la requête *qk*) est déterminée comme suit :

RSV (dj,qi) = 1 si qi dj ; 0 sinon,
RSV (dj, q1 q2) = 1 si RSV (dj,q1) = 1 et RSV (dj,q2) =1 ; 0 sinon,
RSV (dj, q1 q2) =1 si RSV (dj,q1) = 1 ou RSV (dj,q2) =1 ; 0 sinon,
RSV (dj, ¬qi) =1 si qi _ dj ; 0 sinon,

PDF created with pdfFactory Pro trial version www.pdffactory.com

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 (1 ou 0) et spécifique [Frakes et al.,92].

Cependant, ce modèle nécessite de l'utilisateur la connaissance du langage booléen, qui est assez complexe. De plus, l'utilisateur doit pouvoir exprimer très précisément son besoin en informations, sans quoi, la réponse du système risque d'être insatisfaisante [Cooper et al.,88], par ailleurs, l'inconvénient majeur de ce modèle est de ne pas pouvoir classer les documents par ordre de pertinence. Cet inconvénient se fait d'autant plus ressentir si la requête retourne un grand nombre de documents. Pour remédier à ces inconvénients, le modèle booléen étendu et le modèle des ensembles flous ont été développés.

- Le modèle booléen étendu : permet d'améliorer les résultats de la recherche en pondérant les opérateurs et les mots clés attachés à une requête ainsi qu'aux documents. Ce modèle permet aussi de classer les résultats en plaçant les documents jugés les plus similaires à la requête en premier.
- Le modèle des ensembles flous : il nous permet de mesurer le degré de correspondance entre un document et une requête dans un intervalle [0,1], on peut ordonner ainsi les documents dans l'ordre décroissant de leur correspondance avec la requête.

L'inconvénient majeur de ces modèles est qu'ils ne sont pas adaptés au classement (ranking) des documents pertinents, étant donné que les scores de pertinence qu'ils attribuent aux documents sont calculés par des fonctions *min* ou *max* qui ne prennent pas nécessairement en compte toutes les valeurs de pertinences des termes de la requête. Cependant des extensions ont été proposées à ces modèles [Bordogna et al., 00][Loiseau et al, 2004] notamment pour améliorer l'ordonnancement (ranking) des documents sélectionnés.

6.2. Les modèles vectoriels

Les modèles vectoriels englobent Le modèle vectoriel de base (Vector Space Model) le modèle vectoriel généralisé (generalized vector model), Latent Semantic Indexing (LSI) et le modèle connexionniste.

6.2.1. Le modèle vectoriel de base

Le premier système vectoriel de recherche d'information apparaît dans les années 1970 avec le système SMART [G.Salton, 1970].

Dans le modèle vectoriel, les vecteurs de poids de poids représentent document et requête. Chaque poids dans un vecteur désigne l'importance du terme correspondant dans le document ou dans la requête. Pour qu'un vecteur prenne une signification, il faut préalablement définir un espèce vectoriel. L'espace engendré par les N termes d'indexation (t_1, t_2, t_3, …, t_N) que le système a rencontré durant l'indexation, c'est-à-dire l'ensemble des termes de la collection de document.

PDF created with pdfFactory Pro trial version www.pdffactory.com

De manière formelle, les documents et requêtes sont des vecteurs dans un espace vectoriel de dimension N et représenté comme suit : [S. FELLAG 2006]

$D_j =$ $Q_k =$

Où :

d_{ij} : poids du terme t_i dans le document D_j,

q_i : poids du terme t_i dans la requête Q.

La figure 1.3 illustre de façon graphique la représentation de trois documents D1, D2 et D3 et d'une requête Q.

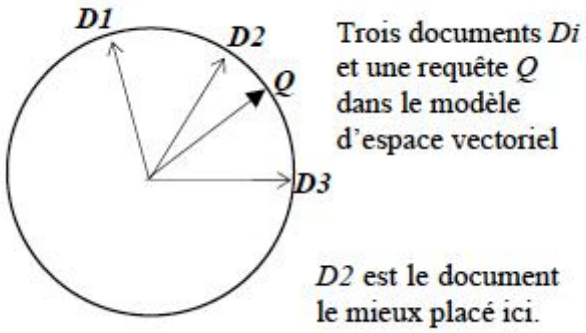


Figure 1.3: Le modèle vectoriel

Le mécanisme de recherche consiste à retrouver les vecteurs documents qui s'approchent le plus du vecteur requête.

Sous l'angle de ce modèle, le degré de pertinence d'un document relativement à une requête, est perçu comme le degré de corrélation entre les vecteurs associés. Ceci nécessite alors, la spécification d'une fonction de calcul de similarité entre vecteurs. La fonction de similarité permet de mesurer la ressemblance des documents et de requête.

Les mesures les plus répondus sont [H.TEBRI, 2004] :

- Le produit interne (ou produit scalaire, non normalisé) :

  $RSV(D_j, Q_k) =$ (1.6)

- La mesure de Dice (rapport sur la moyenne arithmétique des normes) a permis de diminuer l'influence de la langueur des documents, elle est représentée comme suit :

  $RSV(D_j, Q_k) =$ ———— (1.7)

PDF created with pdfFactory Pro trial version www.pdffactory.com

Ø Mesure de cosinus (intersection normalisée compris entre 0 et 1) :

$$RSV(D_j, Q_k) = \frac{\sum_{i=1}^{N} d_{ij}\ q_{ik}}{\sqrt{\sum_{i=1}^{N} d_{ij}^2 * \sum_{i=1}^{N} q_{ik}^2}} \quad (1.8)$$

L'avantage du modèle vectoriel est de pouvoir classer les documents par ordre de pertinence par rapport à une requête. Il offre aussi la possibilité de limiter le nombre de documents retournés à l'utilisateur en ignorant les documents avec un degré de similarité inférieur à un certain seuil. En outre, les SRI se basant sur ce modèle présentent en général des résultats plus satisfaisants que ceux qui se basent sur le modèle booléen [Turtle et al.,91].

Néanmoins, ce modèle ne considère pas les éventuels liens qui peuvent exister entre les termes, chacun des termes est considéré comme indépendant des autres [Croft et al., 92]., [Yates et al 1999]. Pour remédier à cet inconvénient, le modèle vectoriel généralisé et le modèle LSI ont été développés.

Le modèle vectoriel généralisé (Generalized Vector Space Model) : [Wong et al, 85] permet cependant de résoudre le problème d'indépendance des termes.

Modèle Latent Semantic Indexing (LSI) : [Deerwester et al., 1990][Berry et al., 1995] Comparativement au modèle vectoriel, la technique LSI réduit la dimension de l'espace de représentation aux seuls vecteurs de représentation de l'information sémantique, et ce, en réduisant l'effet de variation d'utilisation des termes. Cette technique propose une représentation optimale à partir d'un espace de représentation termes-documents.

L'avantage de ce modèle [Dumais, 95] [Foltz, 90] [Furnas et al, 88] est qu'elle arrive à une représentation pseudo-conceptuelle des documents de la base, permettant de retrouver des documents même s'ils ne contiennent pas les mots des requêtes. Son inconvénient est qu'elle est sensible à la quantité et à la qualité des données traitées.

6.3. Le modèle probabiliste

6.3.1. Le modèle de probabiliste de base

Le modèle de recherche probabiliste est basé sur la théorie des probabilités [Robertson et al., 76] [Salton, 83] [Kuhn, 60][Robertson, 77]. Le processus de recherche se traduit par calcul de proche en proche, du degré ou probabilité de pertinence d'un document relativement à une requête. Pour ce faire, le processus de décision complète le procédé d'indexation probabiliste en utilisant deux probabilités conditionnelles :

P $(t\ |Pert)$: La probabilité pour que le terme t apparaisse dans un document donné sachant que ce document est pertinent pour la requête.

P $(t\ |NonPert)$: La probabilité pour que le terme t apparaisse dans un document donné sachant que ce document est non pertinent pour la requête.

PDF created with pdfFactory Pro trial version www.pdffactory.com

En supposant que la distribution des termes dans les documents pertinents est la même que leur distribution par rapport à la totalité des documents, et que les variables « documents pertinents » et « document non pertinent » sont indépendantes, la fonction de recherche est obtenue en calculant la probabilité de pertinence d'un document P .

P = ——————— (1.8)

P ——————————— (1.9)

Avec :

P(D) = P (D/ Pert) * P(Pert) + P (D / NonPert) * P(NonPert)

Où :

P (D/Pert) (respectivement *P (D/NonPert)*) : Probabilité d'observer D sachant qu'il est pertinent (respectivement non pertinent).

P (Pert) (respectivement *P (NonPert)*) : Probabilité à priori qu'un document soit pertinent (respectivement non pertinent).

Le coefficient de similarité requête document (*RSV*) peut être calculé par différentes formules. Robertson et Spark-Jones [Robertson 96] proposent la formule suivante :

$$RSV = \sum \log \left(\frac{(r+0.5)/(R-r+0.5)}{(n-r+0.5)/(N-n-R+r+0.5)} \right) \quad (1.10)$$

Où :

N: est le nombre total de documents de la base,

n: est le nombre de documents contenant le terme,

R: est le nombre de documents connus comme étant pertinents,

r: est le nombre de documents connus comme étant pertinents et contenants le terme.

L'ajout de 0.5 à tous les membres s'explique par la nécessité d'écarter tous les cas limites qui entraîneraient des valeurs nulles de ces membres.

De manière générale, le modèle probabiliste présente l'intérêt d'unifier les représentations des documents et concepts. Cependant, le modèle repose sur des hypothèses d'indépendance des variables pertinence non toujours vérifiées, ce qui entache les mesures de similitude d'imprécision.

L'inconvénient majeur de ce modèle est que les calculs des probabilités sont complexes et que l'indépendance des variables n'est pas toujours vérifiée voir pas prise en compte.

PDF created with pdfFactory Pro trial version www.pdffactory.com

6.3.2. Les modèles de langue

Les modèles de langues 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:

$$P(Q|d) = \prod_{i=1}^{n} P(ti / d) \quad (1.11)$$

Où :

n est le nombre de termes dans la requête

t*i* est un terme de la requête pour *i=1,2,…,n*

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

$$P(t_i / d) = \frac{tf(ti / d)}{\sum_t tf(t/d)} \quad (1.12)$$

Ou : *tf (ti/d)* est la fréquence du terme ti 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.

7. Evaluation des systèmes de recherche d'information

L'évaluation des systèmes de recherche d'information constitue une étape importante dans l'élaboration d'un modèle de recherche d'information.

Plusieurs critères sont utilisés pour l'évaluation des SRI, parmi ces critères on cite :

- Le temps de réponse du système.
- L'espace utilisé pour le stockage de l'information.
- La qualité des résultats renvoyés (l'efficacité).

Cependant, l'efficacité de système reste le critère le plus important, il est évalué par deux mesures : le rappel et la précision.

PDF created with pdfFactory Pro trial version www.pdffactory.com

7.1. Les mesures d'évaluation

7.1.1. La précision et le Rappel

Les mesures de rappel et de précision permettent d'évaluer la capacité d'un SRI à répondre aux deux objectifs principaux qui sont : retrouver tous les documents pertinents et rejeter tous les documents non pertinents. Afin de présenter ces deux mesures, nous introduisons le partitionnement de l'ensemble des documents restitués (noté B) par le SRI en deux sous ensembles (Figure 1.4) : un sous-ensemble de documents pertinents et un sous ensemble de documents non pertinents.

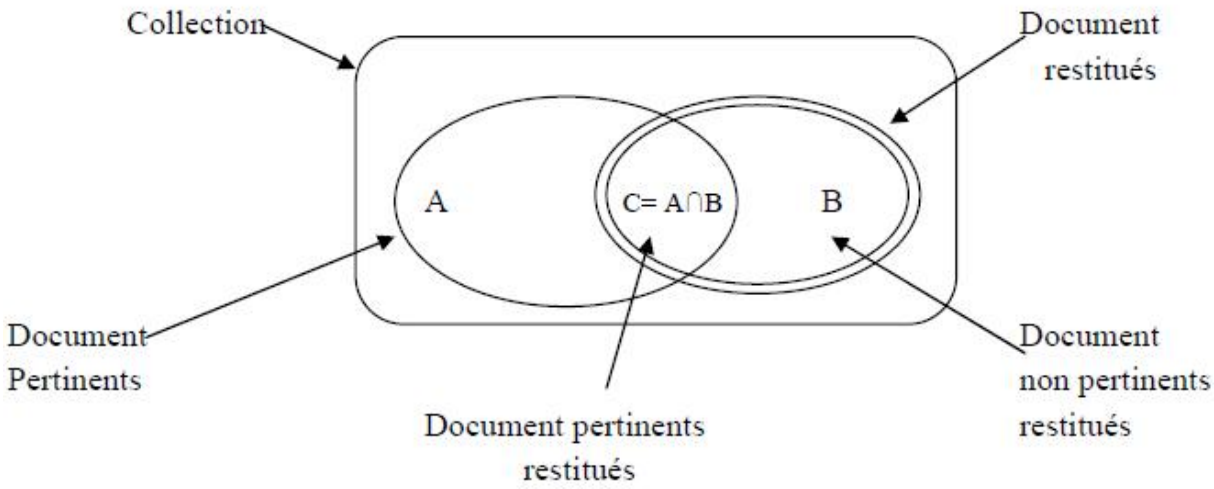


Figure 1.4 : Partition de la collection pour une requête

Les taux de rappel et de précision sont définis comme suit :

- **Taux de rappel :**

Il mesure la proportion de documents pertinents retrouvés parmi tous les documents pertinents de la collection. Il est exprimé par :

$$\text{Rappel} = \frac{\quad}{\quad}$$

Le rappel mesure la capacité du système de retrouver tous les documents pertinents répondant à une requête. Autrement dit, un taux de rappel de 1, signifie que tous les documents pertinents ont été restitués. Inversement, un taux de rappel de 0, signifie qu'aucun document pertinent n'a été restitué par le système.

Le rappel permet aussi de définir le *silence* documentaire qui représente la proportion des documents pertinents non retrouvés par le système :

$$\textit{Silence} = 1 - \text{Rappel}.$$

- **Taux de précision :**

PDF created with pdfFactory Pro trial version www.pdffactory.com

La précision mesure la proportion de documents pertinents restitués parmi tous les documents restitués. Elle est exprimée par :

$$\text{Précision} = \frac{\quad}{\quad}$$

La précision mesure la capacité du système à rejeter tous les documents non pertinents à une requête. Autrement dit, un taux de précision de 1 signifie que seulement les documents pertinents ont été restitués. Tandis qu'un taux de précision de 0, signifie qu'aucun des documents retournés n'est pertinent.

La précision permet aussi de définir le bruit documentaire qui mesure la proportion de documents non pertinents retournés par le système :

$$\textit{Bruit} = 1 - \text{précision}$$

7.1.2. La courbe rappel – précision

La précision mesurée indépendamment du rappel et inversement est peu significative. Pour pouvoir examiner les résultats efficacement, on calcule la paire des mesures (taux de rappel, taux de précision) à chaque document restitué. Le tableau 1 illustre des calculs de précision et de rappel pour les 10 premiers documents renvoyés par un système pour deux requêtes différentes, pour lesquelles la collection contient respectivement 5 et 3 documents pertinents. Les courbes de rappel-précision associées sont ensuite tracées.

	R1			R2		
Rang du doc restitué	Pertinent	Rappel	Précision	Pertinent	Rappel	Précision
1	x	0.20	1	x	0.25	1
2	x	0.40	1		0.25	0.50
3		0.40	0.67	x	0.50	0.67
4	x	0.60	0.75		0.50	0.50
5		0.60	0.60		0.50	0.40
6	x	0.80	0.67		0.50	0.33
7		0.80	0.57		0.50	0.29
8		0.80	0.50		0.50	0.25
9		0.80	0.44		0.50	0.22
10	x	1	0.50	x	0.75	0.30

Tableau 1.1: Exemple de calcul de rappel et précision pour les requêtes R1 et R2

Dans la courbe Figure 1.5, on peut le constater plusieurs valeurs de précision peuvent correspondre au même point de rappel. Afin d'obtenir des courbes plus aisées à lire, on ne représente généralement que la précision calculée à chaque point de rappel (c'est à dire à chaque document pertinent restitué).

PDF created with pdfFactory Pro trial version www.pdffactory.com

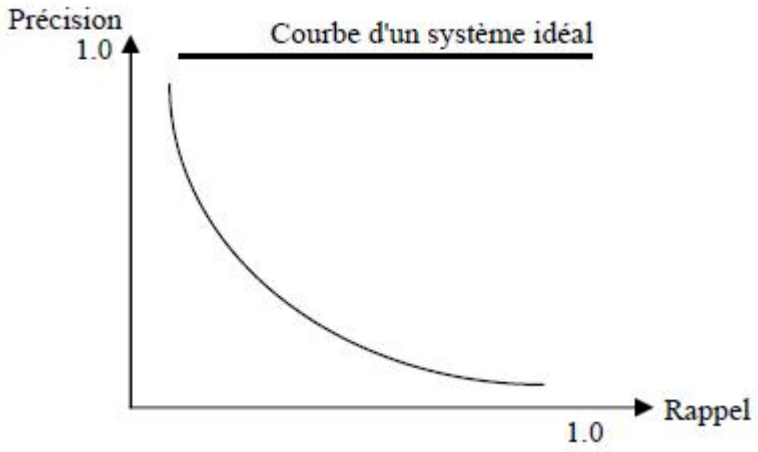


Figure 1.5: Courbe de Rappel-Précision de la requête R1

En effet, plus les deux valeurs de mesures de rappel et de précision sont proches de 1 pour un système donné, plus ce système est performant. Et, en pratique, plus l'une des deux valeurs augmente plus l'autre diminue. Ainsi en faisant varier l'une des valeurs on peut obtenir une courbe représentant la précision en fonction du rappel (ou inversement).

Il est possible, grâce a cette représentation, de comparer deux systèmes de recherche d'information. Particulièrement, si les deux courbes ne se croisent pas, on peut déduire que le système dont la courbe se trouve au dessus de l'autre, est plus performant (car il offre un taux de rappel et de précision plus élevé).

7.2. Mesures alternatives :

7.2.1. La précision à X documents

La précision à x est la précision obtenue en considérant les x premiers résultats retournés par le SRI (en supposant que ceux-ci sont triés suivant leur pertinence).

7.2.2. La précision exacte ou la R-précision

La précision exacte ou la *R-precision*. La précision exacte représente celle obtenue à l'endroit où elle vaut le rappel. Si la requête admet *n* documents pertinents, la précision exacte est celle calculée pour les *n* premiers documents de la liste ordonnée des documents restitués.

7.2.3. La précision moyenne

La précision moyenne est la moyenne des valeurs de précision à chaque document pertinent de la liste ordonnée. Elle tient compte à la fois de la précision et du rappel. Elle représente la moyenne des précisions (non interpolées) calculées pour chaque document pertinent à

PDF created with pdfFactory Pro trial version www.pdffactory.com

trouver, au rang de ce document. Si un document pertinent est retourné à la 10ème position, la précision pour ce document est « la précision à 10 documents ». Si un document pertinent n'a pas été trouvé par le système, la précision pour ce document est nulle.
Elle est calculée avec la formule suivante :

$$\text{Précision moyenne} = \sum_{x=1}^{n} \frac{p(x) * pert\ (x)}{npert} \quad (1.13)$$

Où :
n : indique le nombre de documents retournés par e système ;
p(x) : indique la précision à x ;
pert(x) : est une fonction qui vaut 1 si le document à la position x est pertinent, 0 sinon ;
npert est le nombre total de documents pertinents retournés

7.2.4. F-mesure

Van Rijsbergen [Rijsbergen, 79] a introduit la *F-mesure* comme combinaison du rappel et de la précision. La F-mesure définit la moyenne harmonique du rappel et de la Précision [Boubekeur, 08] qui est calculée comme suit :

$$\text{F-mesure} = \frac{2 * Rappel * Précisipn}{Rappel + Précision} \quad (1.14)$$

Maximiser la F-mesure revient à trouver le meilleur compromis entre le rappel et la précision.
La valeur maximale de la F-mesure est 1. Elle est obtenue quand tous les documents classés sont pertinents, et quand tous les documents pertinents ont été classés.

7.3. Les compagnes d'évaluation et les collections de référence

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.

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

Les compagnes d'évaluation définissent aussi les critères qui doivent être utilisés pour l'évaluation, les plus utilisés sont le rappel et la précision.

Parmi les compagnes d'évaluation qui ont été entreprises au cours de l'histoire, on trouve la compagne TREC qui constitue le modèle dominant et que nous développons par la suite.

PDF created with pdfFactory Pro trial version www.pdffactory.com

7.3.1. Les compagnes TREC

Les compagnes TREC (Text Retrieval Conferences) [97] [42] constituent l'approche la plus utilisée dans le domaine de l'évaluation des systèmes de recherche d'information. C'est un projet qui a été lancé en 1992 par le NIST (National Institue of Standards and Technology) avec le soutien financier de l'IAPRA (Intelligence Advanced Reseach Projects Activity). L'objectif de TREC est d'encourager les travaux dans le domaine de la recherche d'information, en fournissant l'infrastructure nécessaire (collections de tests) et en proposant les méthodologies d'évaluation des SRI.

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

8. Conclusion :

Dans ce premier chapitre, nous avons présenté les principales notions et les principaux concepts de la recherche d'information, nous avons développé les étapes d'un processus de recherche d'information, après nous avons vu les principaux modèles de recherche d'information. Et enfin nous avons présenté les principaux critères d'évaluation d'un SRI.

Le chapitre qui va suivre portera sur la présentation des modèles de langue pour la recherche d'information.

PDF created with pdfFactory Pro trial version www.pdffactory.com

1. Introduction

Les méthodes statistiques basées sur des corpus, au lieu de celles basées sur des connaissances préétablies, ont eu de grands succès dans la linguistique informatique [Manning99]. Ces méthodes tentent de capter, d'une manière statistique, les régularités d'une langue en observant des phrases dans un corpus d'entraînement. Les développements récents dans ce domaine ont démontré que les méthodes ainsi développées peuvent être utilisées avec succès dans l'étiquetage des catégories syntaxiques (*POS-tagging*), la reconnaissance de parole [Jelinek97], et même en traduction automatique [Brown93]. Ce succès est non seulement dû à l'avancement des méthodes utilisées, mais aussi à la disponibilité de plus en plus grande des corpus d'entraînement de différentes sortes. Par exemple, le Penn-treebank est un corpus largement utilisé pour entraîner l'étiqueteur syntaxique. Le Hansard est un corpus parallèle souvent utilisé pour entraîner des modèles de traduction statistiques.

Le domaine de recherche d'information s'est beaucoup inspiré du succès des méthodes statistiques en linguistique informatique. Dans une certaine mesure, la RI a des choses en commun avec la linguistique informatique : Les deux domaines possèdent de grandes masses de textes, ce qui permet d'entraîner des modèles statistiques.

L'idée de base des modèles de langue est de déterminer la probabilité $P(Q|D)$ – la probabilité que la requête Q puisse être générer à partir du document D. Cette formulation est similaire à l'idée derrière les modèles probabilistes formulée pour la première fois dans [Maron60]. Cependant, comme on peut voir plus loin, la façon de calculer $P(Q|D)$ dans les modèles de langue est différente de celle des modèles probabilistes traditionnelles.

Le terme « modèle de langue » est emprunté de la linguistique informatique, où l'objectif d'un modèle de langue est de capter les régularités linguistiques par une ou plusieurs fonctions probabilistes. Ainsi, nous commençons notre présentation par une brève description des méthodes développées pour la modélisation statistique de langue en linguistique informatique.

2. Modèle de langue en linguistique informatique

2.1. Idée de base

Par « modèle de langue », on désigne une fonction de probabilité P qui assigne une probabilité P(s) à un mot ou à une séquence de mots s en une langue. Une fois cette fonction définie, il est possible d'estimer la probabilité d'une séquence de mots quelconque dans la langue, ou d'un point de vue générative, d'estimer la probabilité de générer cette séquence de mots à partir du modèle de la langue.

Considérons la séquence s composée des mots suivants : m1, m2,…, mn. La probabilité P(s) peut être calculée comme suit :

PDF created with pdfFactory Pro trial version www.pdffactory.com

P(S) = $\prod_{i=1}^{l} P(mi|m1 \dots\dots\dots\dots\dots\dots m_{i-1}$ (2.1)

Si on utilise la règle de chaîne en théorie de probabilité pour calculer cette probabilité, il y a souvent trop de paramètres (c'est-à-dire P ($m_i/m_1 \dots m_{i-1}$)) à estimer, et ceci est souvent impossible de réaliser. Ainsi, dans les modèles de langue utilisés en pratique, des simplifications sont souvent faites. En général, on suppose qu'un mot m_i ne dépend que de ses n-1 prédécesseurs immédiats, c'est-à-dire :

P (mi|m1……..mi-1) = P (mi|mi-n+1……..mi-1) (2.2)

On utilise, dans ce cas, un modèle de langue n-gramme. En particulier, les modèles souvent utilisés sont les modèles uni-gramme, bi-gramme et tri-gramme comme suit :

Uni-gramme :

P(S) = $\prod_{i=1}^{l} P(mi)$ (2.3)

Bi-gramme :

P(S) = $\prod_{i=1}^{l} P(mi \mid m_{i-1}) = \prod \frac{p(m_{i-1}-m_i)}{p(m_{i-1})}$ (2.4)

Tri-gramme :

P(s) = $\prod_{i=1}^{L} p(m_i|m_{i-2}m_{i-1}) = \frac{p(m_{i-2}\, m_{i-1}\, m)}{p(m_{i-2}m_{i-1})}$ (2.5)

Ce que l'on doit estimer sont les probabilités P (mi) (un-gramme), P(mi-1m) (bi-gramme) et P(mi-2mi-1 mi) (tri-gramme) pour la langue. Cependant, il est difficile d'estimer ces probabilités pour une langue dans l'absolue. L'estimation ne peut se faire que par rapport à un corpus de textes C. Si le corpus est suffisamment grand, on peut faire l'hypothèse qu'il reflète la langue en général. Ainsi, le modèle de langue peut être approximativement le modèle de langue pour ce corpus – P (•|C). Selon les fréquences d'occurrence d'un n-gramme α, sa probabilité P(α|C) peut être directement estimée comme suit :

P(α)=$\frac{|\alpha|}{\sum_{\alpha_j \in C} |\alpha_j|} = \frac{|\alpha|}{|C|}$ (2.6)

où

| α | est la fréquence d'occurrence du n-gramme α dans ce corpus,

αi est un n-gramme de la même longueur que α, et

PDF created with pdfFactory Pro trial version www.pdffactory.com

|C| est la taille du corpus (c'est-à-dire le nombre total d'occurrences de mots). Ces estimations sont appelées les estimations de vraisemblance maximale (Miximum Likelihood, ou ML). On désignera aussi ces estimations par PML.

Nous donnons ici un exemple simple pour illustrer l'estimation de la probabilité uni-gramme ainsi que son utilisation pour calculer la probabilité d'une phrase.

Supposons un petit corpus contenant 10 mots, avec les fréquences comme montrées dans Table2.1

mot	Le	Un	Prof	Ml	Dit	Aime	De	Langue	Modèle	RI
fréq	3	2	2	1	2	1	4	2	1	2
P(□ ¢)	0.15	0.1	0 .1	0 .05	0.1	0.05	0.2	0.1	0.05	0.1

Table 2.1 : Exemple d'estimation de vraisemblance maximale.

En utilisant l'estimation de vraisemblance maximale, nous obtenons les probabilités comme illustrées dans la table (note : la fréquence totale de mots dans ce corpus est |C| = 20).

En utilisant ces probabilités estimées, nous pouvons calculer la probabilité de construire la séquence s = « le prof aime le ML » dans cette langue comme suit :

$$P(S) \approx P(S/C) = P(le/C) * P(prof/C) * P(aime/C) * P(le/C) * P(ML/C)$$

$$=0.15 * 0.1 * 0.05 * 0.15 * 0.05$$

2.2. Lissage

Plus le corpus utilisé pour ces estimations est grand, plus on peut espérer obtenir des estimations de probabilité justes. Cependant, quelle que soit la taille du corpus d'entraînement, il y a toujours des mots ou des séquences de mots absents du corpus (par exemple le mot « non » dans notre corpus jouet). Pour ces mots ou séquences de mots, leur estimation de probabilité est 0. La conséquence de cette probabilité nulle est qu'on attribuera une probabilité nulle à toute séquence de mots ou des phrases contenant un mot ou un n-gramme non rencontré dans le corpus. Par exemple, la phrase s = « le prof dit non » aura une probabilité P(S|C) = 0. En d'autres termes, les modèles ainsi construits ne sauraient reconnaître que les phrases dont les n-grammes sont tous apparus dans le corpus. Ce sont donc des modèles très limités. Afin de généraliser les modèles, on voudrait assouplir cette attribution systématique de probabilité nulle aux mots ou séquences de mots non rencontrés. Cette procédure qui consiste à attribuer une probabilité non-nulle á ces éléments est appelée le lissage. Le lissage peut aussi être vu comme une façon d'éviter le surentraînement d'un modèle sur un corpus, et de doter du modèle d'une plus grande capacité de généralisation.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Le principe de lissage peut être résumé ainsi : Au lieu de distribuer la totalité de masse de probabilité sur les n-grammes vus dans le corpus d'entraînement, on enlève une partie de cette masse et la redistribue aux n-grammes non vus dans le corpus. De cette façon, les n-grammes absents du corpus vont recevoir une probabilité non-nulle.

Sur la façon d'enlever et de redistribuer une partie de masse de probabilité, il y a une série de méthodes proposées dans la littérature. Ici, nous présentons quelques-unes unes classiques.

Les techniques de lissage les plus connues sont récapitulées dans le tableau suivant :

Méthode de lissage	Formules	Description
Lissage de Laplace	$P_{ajouter_{un}}\left(\frac{a}{C}\right) = \frac{\lvert a\rvert + 1}{\sum_{a_i \in V}(\lvert a_i\rvert + 1)}$	V : ensemble du vocabulaire d'indexes, $\lvert\alpha\rvert$: nombre d'occurrences du n-gramme, **è** Pas de performance sur les corpus de petite taille.
Lissage de Good-Turing	$P_{GT}(\alpha) = \frac{r^*}{\sum_{\alpha_i \in C}(\lvert\alpha_i\rvert)}$ $r^* = (r+1)\frac{n_{r+1}}{n_r}$	r: fréquence d'occurrence d'un n-gramme α, n_r:nombre de n-gramme apparus r fois, **è** Recommandé pour les n-gramme de faibles fréquences, car l'estimation GT est instable pour les n-grammes de grandes fréquences
Lissage de Backoff	$\mathrm{PKatz(mi\lvert mi-1)} = \begin{cases} \mathrm{PGT(mi\lvert mi-1)} & \text{si } \lvert \mathrm{mi-1-mi}\rvert > 0 \\ \alpha(m_{i-1})\mathrm{PKartz(mi)} & \text{sinon} \end{cases}$ Où : $\alpha(m_{i-1}) = \frac{1 - \sum P_{GT}(m_i \lvert m_{i-1})}{1 - \sum P_{ML}(m_i)}$	$\alpha(m_{i-1})$: paramètre de normalisation, $m_{i,}m_{i-1}$: n-grammes observés d'ordre i et i-1, **è** Utilise un modèle d'ordre inférieur aux n-grammes dont la fréquence est nulle.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Lissage par interpolation	$P_{JM}(m_i \mid m_{i-1})=\lambda_{m-1}P_{ML}(m_i \mid m_{i-1})+(1-\lambda_{m-1})P_{JM}(m_i)$. En RI, cette méthode prend un autre sens : $(1-\alpha)P\left(\frac{w}{d}\right)+\ \alpha P(\frac{w}{C})$	α:est un paramètre déterminé pour maximiser l'espérance des données, w : correspond à un mot observé dans le document ou dans la collection, **è** Interpolation du modèle du document avec celui de la collection.
Lissage de Dirichlet	$P_{Dir}=\frac{c(w,d)+\ \mu\ P(w\|C)}{\sum_w c(w,d)+\mu}$	$c(w,d)/\ P(w\|C)$: fréquence d'apparition du mot w dans le document d/ ou dans la collection C μ:est un paramètre appelé pseudo fréquence, **è** On peut aisément remarquer qu'il s'agit d'une généralisation du lissage de Laplace.

Figure 2.1 : Les différentes techniques de lissage.

3. Modèle de langue en RI

Dans les modèles traditionnels de RI, on tente de modéliser la relation de pertinence entre un document et une requête. Typiquement, dans le modèle probabiliste, on tente d'estimer la probabilité P (R|D, Q) – la probabilité de pertinence d'un document face à une requête. Typiquement, cette probabilité est calculée selon des méthodes paramétriques : on suppose que la distribution des mots suit une certaine norme (par exemple, distribution Poisson) parmi les documents pertinents (et non-pertinents). En fonction des distributions des mots parmi deux ensembles (pertinent et non-pertinent) de documents échantillons, on peut estimer les probabilités des mots pour la pertinence. La probabilité P (R|D, Q) pourra alors être calculée. Pour plus de détails, les lecteurs peuvent se référer au chapitre portant sur le modèle probabiliste dans ce livre.

Le principe des approches utilisant un modèle de langue est différent. On ne tente pas de modéliser directement la notion de pertinence dans le modèle; mais on considère que la pertinence d'un document face à une requête est en rapport avec la probabilité que la requête puisse être générée par le modèle de langue du document. Ainsi, on considère qu'un document D incarne un sous-langage, pour lequel on tente de construire un modèle de langue MD. Le score

PDF created with pdfFactory Pro trial version www.pdffactory.com

du document face à une requête Q est déterminé par la probabilité que son modèle génère la requête :

$$\text{Score}(Q, D) = P(Q|MD) \tag{2.7}$$

On écrira aussi P(Q|D) pour représenter la même probabilité dans les descriptions plus tard.
De façon générale, une requête peut être vue comme une suite de mots : Q = t1t2…tn. Nous avons donc :

$$\text{Score}(Q, D) = P(t1t2…tn|MD) \tag{2.8}$$

Ainsi formulé, le problème de la RI devient similaire à la modélisation statistique de langue. Il est donc possible d'utiliser ses techniques développées pour cette dernière en RI.
Le principe que nous venons de décrire est celui que la plupart des modèles de langues en RI utilisent. Il y a aussi quelques autres variantes.

$$\text{Score}(Q, D) = P(t1t2…tm|MQ) \tag{2.9}$$

Cependant, dans cette formulation, les documents longs, et contenant des mots fréquents vont être favorisés. Afin de résoudre ce problème, nous pouvons utiliser la loi de Bayes :
Nous pouvons aussi procéder dans le sens inverse : On tente de créer un modèle de langue pour la requête, et de déterminer le score d'un document D = t1t2…tm par la probabilité que le document peut être généré par ce modèle :

$$P(M_Q|D) = \frac{P(M_Q)P(D|M_Q)}{P(D)} \tag{2.10}$$

On suppose ensuite que P(D) ≈ P(D|C) et que $P(M_Q)$ est une constante *c.* Ainsi :

$$P(M_Q|D) \approx \frac{C\,P(D|M_Q)}{P(D|C)} \tag{2.11}$$

Dans cette nouvelle formule, nous tenons compte de la longueur du document et des mots contenus dans le document, en ajoutant le facteur P(D|C).
Il est aussi possible de construire un modèle de langue pour le document P(•|MD) et un autre pour la requête P(•|MQ). Le score d'un document face à la requête peut être déterminé par une comparaison entre les deux modèles. C'est le principe utilisé dans la méthode d'entropie croisée.
Finalement, on peut aussi voir la relation entre une requête et un document pertinent comme celle d'une traduction : un document pertinent est une « traduction » de la requête; et on tente de déterminer la probabilité de cette traduction.
Dans les sections qui suivent, nous présentons plus en détail les méthodes spécifiques pour implanter ces principes.

PDF created with pdfFactory Pro trial version www.pdffactory.com

3.1. RI comme génération de la requête par le document

Dans cette catégorie de modèle, la pertinence d'un document vis-à-vis d'une requête est vue comme la probabilité de générer la requête par le modèle de document.

Pour ce faire, un modèle de document est construit pour chaque document, soit : M_d ensuite on estime la probabilité de la requête, soit : $P(Q|M_d)$

Nous présentant ci-dessous deux modèles représentatifs de cette catégorie :

3.1.1. Modèle de Pont Croft

Le premier modèle à la RI base sur la modélisation de langue est [Ponte98]. L'intuition est qu'une requête n'est pas créée de façon aléatoire, mais l'utilisateur a une idée (un modèle) sur le document idéal qui peuvent apparaître dans D et de son modèle MD. À partir de MD, l'utilisateur choisit (génère) les termes pour constituer sa requête. Ainsi, la requête devrait être générée à partir d'un document pertinent ou de son modèle. La probabilité de cette génération est considérée comme le score du document :

$$\text{Score } (D|Q) = P(Q|M_D) \tag{2.12}$$

Nous apportons quelques remarques ici :
- Une requête peut être considérée comme une suite de mots : Q = t1, t2, … tn.
- Comme dans la plupart des modèles de langues utilisées en RI, la simplification suivante a été faite : on suppose que les mots dans une requête sont indépendants.
- Dans le modèle de Ponte et Croft, ils utilisent modèle de Bernoulli multiple, c'est-à-dire que non seulement les mots présents dans la requête, mais aussi ceux absents de la requête, sont pris en compte. Dans la plupart d'autres modèles, on utilise plutôt le modèle multinomial, où on ne considère que les mots présents dans la requête.

Supposons, sans perdre la généralité, que la requête Q est composée de l'ensemble de mots t_1, t_2, … t_n, et que les mots t_{n+1}, t_{n+2}, … t_l sont absents de la requête. P(Q|MD) peut être ré-exprimée comme suit :

$$P(Q|M_Q) = P(t_1 t_2 \dots\dots\dots\dots t_n|M_D) \times P(7t_{n+1}, 7t_{n+2}, \dots\dots\dots\dots\dots, 7t|M_D).$$
$$= \prod_{i=1}^{n} P(t_i|M_D) \times \prod_{i=n+1}^{i} \big(P(7t_i|M_D)\big).$$

$$= \prod_{t_i \in Q} P(t_i|M_D) \times \prod_{i \neq Q} (1 - P(t_i|M_D)) \tag{2.13}$$

Ponte et Croft ont proposé une estimation de la probabilité PPC(ti|MD) d'une façon similaire à l'approche Backoff. Ils combinent un modèle de langue du document avec un modèle de langue du corpus comme suit :

PDF created with pdfFactory Pro trial version www.pdffactory.com

$$P_{PC}(\ t_i|\ M_D) = \begin{cases} P_{ML}\ (t_i\ |\mathrm{D})^{1-R(ti,d)} * \ P_{avg}\ (t_i)^{R(ti,D)} & si\ tf\ (\mathrm{ti}\ ,\mathrm{D}) > 0 \\ P_{ML}\ (\mathrm{ti}|\mathrm{C}) & \text{sinon} \end{cases}$$

(2.14)

où

$tf\ (\mathrm{ti}\ ,\mathrm{D})$ est la fréquence de ti dans le document D.

Dans cette formule, on note deux éléments principaux $P_{ML}\ (t_i\ |\mathrm{D})$ et $P_{ML}\ (t_i\ |\mathrm{C})$. Il y a aussi quelques autres éléments ajoutés pour tenir compte de certaines particularités de la RI :

$P_{avg}\ (ti)$ est la probabilité moyenne du terme t_i dans les documents qui le contiennent; R(t_i |d)est une fonction de « risque » déterminée comme suit :

$$R(t,D)=\frac{1}{1+f}*(\frac{f}{1+f})^{tf(t,D)} \qquad (2.15)$$

où f est la fréquence moyenne du mot t dans les documents qui le contient. L'élément$P_{avg}\ (t_i)^{R(ti,D)}$ est ajouté pour contrer le « risque » de l'estimation P_{ML}(ti |D)en tenant compte de $P_{avg}\ (ti)$ calculée sur le corpus. Ceci ressemble à un lissage par interpolation.

Le fait de considérer les mots absents de la requête est intéressant : il permet de faire la différence entre un document qui couvre beaucoup de sujets et un autre document qui ne couvre que le sujet de la requête, donc l'aspect spécificité du document. Cependant, on peut facilement voir que si les termes qui n'apparaissent pas dans la requête sont nombreux (ce qui est le cas en général), le calcul devient très complexe.

3.1.2. Modèle de Hiemstra et al. Est de Miller et al.

Hiemstra et al. [Hiemstra98] utilisent le même principe de génération de la requête par le document. La formule que Hiemstra propose est la suivante :

Score (D,Q)= P(D|Q)= P(D|t1t2……tn)

$$= P(D)\frac{P(t_1 t_2 \ldots\ldots t_n|D)}{P(t_1 t_2 \ldots\ldots t_n)} \qquad (2.16)$$

Il suppose que P(t1t2……tn) est une constante 1/c, et que les mots dans la requête sont indépendants. On a donc :

$$score(D,Q) = c\ P(D) \prod_{t\in Q} P(t_i|D) \qquad (2.17)$$

Pour P(ti|D), Hiemstra utilise une approche d'interpolation :

PDF created with pdfFactory Pro trial version www.pdffactory.com

$$P(t_i|D) = \alpha\, P_{ML}(t_i|D) + (1-\alpha) P_{ML}(t_i|C)$$

(2.18)

L'estimation de vraisemblance maximale de $P_{ML}(t_i|D)$ et $P_{ML}(t_i|C)$ sont respectivement

$$\frac{tf(t_i,D)}{|D|} \; et \; \frac{df(t_i)}{\sum_{t_j \in V} df(t_j)}$$

où df(ti) est le nombre de documents contenant ti, et V est le vocabulaire d'indexes.
Une fois on prend le logarithme, on obtient :

$$\log(P(t_1\, t_2 \ldots\ldots\ldots\ldots t_n|D) = \log \prod_{t\in Q} P(t_i|D)$$

$$= \sum_{i=1}^{n} \log(\alpha\, P_{ML}(t_i|D) = \log \prod_{t\in Q} P(t_i|D)$$

$$= \sum_{i=1}^{n} \log\left(1 + \frac{\alpha\, P_{ML}(t_i|D)}{(1-\alpha) P_{ML}(t_i|C)}\right) + \sum_{i=1}^{n} (1-\alpha) P_{ML}(t_i|C)$$

(2.19)

On remarque que le dernier composant de cette formule ne dépend pas de document, mais seulement de la requête et le corpus C. Ainsi, c'est une constante que l'on peut ignorer pour enfin classer les documents.
En ce qui concerne la valeur de α, la façon la plus simple consiste à déterminer une constante pour α. Mais en principe, ce paramètre peut bien dépendre du document ou de la requête. Ainsi, une façon plus sophistiquée consiste à estimer une valeur de α en utilisant un processus d'optimisation automatique telle la maximisation de l'espérance (EM).
La probabilité a priori d'un document D est estimée comme suit :

$$P(D) = \frac{|D|}{|C|}$$

En somme, on a :

$$\text{logScore}(Q, D) = \sum_{tQ} \log\left(1 + \frac{\alpha * tf(ti,D) \sum df(ti)}{(1-\alpha)|D|\, df(t)}\right) + \log\frac{|\boldsymbol{D}|}{|\boldsymbol{C}|} \qquad (2.20)$$

On pourrait estimer la probabilité a priori conditionnée sur autres propriétés des documents, par exemple la forme de URL ou nombre des liens. Cette approche s'est avérée très efficace dans une application RI particulière - la recherche des pages d'entrée (entry page) [Kraaij02].
Miller et al. [Miller98, Miller99] utilise une formulation similaire, à quelques détails près. La plus grande différence entre le modèle de Miller et celui de Hiemstra est la façon d'implanter le modèle : Miller utilise un modèle de Markov caché à deux états comme suit :

PDF created with pdfFactory Pro trial version www.pdffactory.com

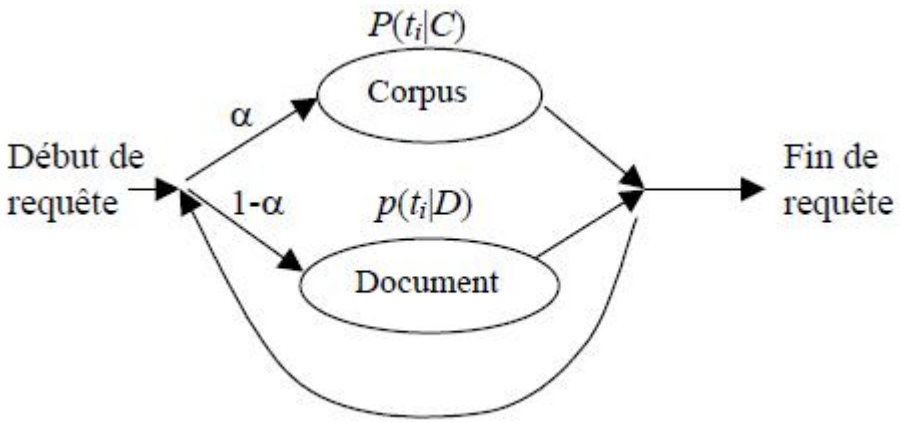


Figure2.2: Modèle de Markov caché à deux états

Ce modèle correspond à la formule suivante :

$$P\left(Q\middle|D_{est\ pertinent}\right) = \prod_{t_i \in Q} \alpha\, P_{ML}\,(t_i|D) + (1-\alpha) P_{ML}(t_i|C) \qquad (2.21)$$

Ce qui est similaire au modèle de Hiemstra (équation 2.18).
Bien que Hiemstra et Miller aient tous utilisé le lissage par interpolation, leur intention initiale n'était pas pour traiter la claire semence de données, mais pour mieux modéliser la requête, dans laquelle certains mots peuvent être non-pertinents. Ainsi, dans la formulation de Hiemstra, un autre élément tf(t, Q) – la fréquence du terme t dans la requête Q - est inséré pour tenir compte de ce fait. Dans notre présentation, nous n'en avons pas décrit les détails.

3.2. Génération de documents par le modèle de la requête

Dans cette catégorie, un modèle de requête est construit soit : M_Q, ensuite on essaye de trouver la probabilité qu'un document D qu'il soit générer par le modèle de la requête soit :

$$P\,(D|\,M_Q)$$

Nous allons illustrer un modèle représentatif de cette catégorie ci-dessous :

Modèle de Relevance :

On désigne par le modèle de Relevance (La Relevance Feedback) la réinjection de la pertinence. Ce modèle repose sur l'hypothèse suivante :

La requête initiale de l'utilisateur n'est pas la requête idéale pour obtenir les documents qu'il cherche.

Delà, le modèle de Relevance a été conçu dans le but de déplacer le vecteur de la requête pour la rapprocher des documents pertinents.

PDF created with pdfFactory Pro trial version www.pdffactory.com

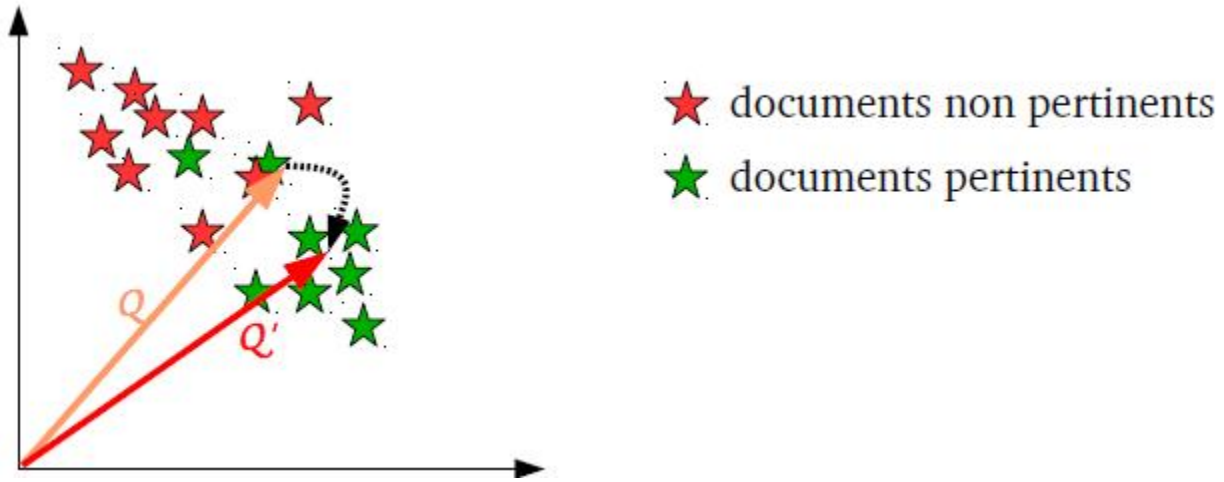


Figure 2.3 : Exemple du modèle de Relevance.

Formule de Rocchio :

$$\vec{Q}' = \alpha \vec{Q} + \beta \vec{P} + \gamma \vec{N} \vec{P}$$

- $\vec{Q}'$: nouveau vecteur requête
- α : valeur positive supérieure aux autres (ex : 1)
- $\vec{Q}$: vecteur requête initial
- β : valeur positive (ex : *0.5*)
- $\vec{P}$: moyenne des vecteurs des documents pertinents
- γ : valeur négative (ex : *-0,25*)
- $\vec{N}\vec{P}$: moyenne des vecteurs des documents non pertinents

3.3. Comparaison entre le modèle de requête et le modèle de document

Pour cette dernière catégorie de modèle, on construit un modèle de langue pour la requête et un autre modèle pour les documents. Par la suite, on effectue la similarité ou bien la comparaison entre les deux modèles.

PDF created with pdfFactory Pro trial version www.pdffactory.com

On retrouve ci-dessous la mesure utilisé dans cette catégorie :

Modèle de l'Entropie Croisée :

Une variante du modèle, basé sur le ratio de vraisemblance, est de représenter la RI comme une entropie croisée (ou écart d'entropie). Nous allons montrer comment nous pouvons effectivement passer de l'équation du ratio vers une forme entropique.
On suppose comme d'habitude, l'indépendance des termes, en considérant une fonction logarithmique et après quelques transformations, cette équation peut s'écrire de la façon suivante :

$$LR(Q|D) = \log \frac{P(Q|M_D)}{P(Q|M_C)} \tag{2.22}$$

Supposons une distribution multinomiale de termes dans le document et dans le corpus, nous avons :

$$P(Q|M_D) = \frac{|Q|!}{\prod_{t_i \in Q} tf(t_i, Q)!} \prod_{t_i \in Q} P(t|D)^{tf(t_i,Q)} \tag{2.23}$$

$$P(Q|M_C) = \frac{|Q|!}{\prod_{t_i \in Q} tf(t_i, Q)} \prod_{t_i \in Q} P(t|C)^{tf(t_i,Q)}$$

Où |Q| est la taille de la requête (le nombre d'occurrences de mots). Ainsi, nous avons:

$$\mathrm{LR(Q|D)}=\sum_{i=1}^{n} tf(t_i, Q) \times \log \frac{\alpha\, P(t_i|M_D)+(1-\alpha)P(t_i|M_C)}{P(t_i\,|M_C)} \tag{2.24}$$

où P(Q|Mc) est la probabilité générative de la requête étant donné un modèle de langage estimé sur un corpus C et tf(ti, Q) est la fréquence du terme ti dans a requête.
Pour chaque terme de la requête, le ratio de vraisemblance mesure le rapport entre la probabilité d'observer une requête donnée étant donné un modèle de document sur la probabilité d'observer cette requête étant donné le modèle de la collection.
Les scores dans l'équation dépendent de la longueur de la requête, ils peuvent être facilement normalisés en les divisant par la longueur de requête (comme la longueur est constante, elle n'intervient pas dans le score). La forme normalisée de l'équation 31 peut donc s'exprimer comme suit :

$$\mathrm{NLR(Q|D)}=\log \frac{P(Q|M_D)}{P(Q|M_C)}$$

PDF created with pdfFactory Pro trial version www.pdffactory.com

$$= \sum \frac{tf(t_i,Q)}{\sum_i tf(t_i,Q)} \times \log \frac{\alpha P(t_i|M_D)+(1-\alpha)P(t_i|M_C)}{P(t_i|M_C)} \quad (2.25)$$

L'étape suivante est de considérer le rapport $\frac{tf(t_i,Q)}{\sum tf(t_i,Q)}$ comme une estimation du maximum de

Vraisemblance de la distribution de probabilité représentant la requête P(ti|MQ). L'équation (2.25) peut être réinterprétée comme la relation entre les deux distributions de probabilité P(t|MD) et P(t|MQ), normalisées par la troisième distribution P(t|MC) :

$$NLR(Q|D) = \sum_{i=1}^{n} P(t_i|Q) \times \frac{P(t_i|M_D)}{P(t_i|M_C)} \quad (2.26)$$

Le modèle mesure de combien le modèle de document pourrait coder des événements du modèle de requête mieux que le modèle de corpus. En théorie de l'information, ceci est interprété comme une différence entre deux entropies croisées, exprimée comme suit :

$$NLR(Q|D) = H(X|C) - H(X|D) \quad (2.27)$$

où X est une variable aléatoire avec la distribution de probabilité P(ti)=P(ti|MQ) et C et D sont des fonctions de masse de probabilité représentant respectivement la distribution marginale (du corpus) et le modèle du document.

L'entropie croisée permet donc de mesurer en moyenne l'écart entropique sur le fait que le modèle de document correspond (suit) bien la distribution probabiliste de la requête. D'autres formes entropie croisées ont été étudiées, on y trouve notamment celle basée sur l'information de Kullback-Leibler dans [lavrenko01].

Remarquons que dans l'entropie croisée, on voit apparaître implicitement le modèle de langue de la requête MQ. Si on l'utilise seul (ou pour remplacer le modèle de langue du document), l'approche souffrira encore plus du problème de l'information limitée pour la construction du modèle. Dans le cas de RI pour des requête ad hoc (comme sur le Web), typiquement, la requête est très courte. Cette approche est donc impraticable. Dans le contexte de « topic tracking » où la « requête » - un texte – est plus longue, on peut davantage utiliser cette approche. Cependant, sa performance n'est pas très bonne. Dans l'approche d'entropie croisée, comme celle utilisée dans le modèle de pertinence [Lavrenko01], le modèle de langue de la requête n'est pas utilisé seul, mais ensemble avec le modèle du corpus et du document, dans un cadre qui tente de modéliser la pertinence.
Remarquons aussi que le principe de comparaison du modèle du document et le modèle de la requête est aussi utilisé dans le cadre de « minimisation de risque » [Lefferty01, Zhai02]. La décision pour retrouver un document est basée sur une fonction de perte comparant les deux modèles. Ils ont ensuite proposé un modèle de langue à deux étapes : une première étape pour

PDF created with pdfFactory Pro trial version www.pdffactory.com

lisser le modèle de document (pour le problème de clairesemence de données), et une deuxième étape pour tenir compte de la pertinence des termes dans la requête (le modèle de requête). Les techniques spécifiques utilisées pour les deux étapes sont respectivement le lissage Dirichlet et le lissage par interpolation.

4. Prise en compte des informations à priori dans les modèles de langues

Selon les approches qu'on vient de présenter, les propriétés (taille de document, nombre de liens entrants, etc) indépendantes des requêtes peuvent être utilisées pour conditionner la probabilité à priori de pertinence d'un document. Si la probabilité à priori de pertinence P(D) n'est pas conditionnée par l'une de ces propriétés alors cette probabilité représente la probabilité de prélever un document de la collection, sinon si elle est conditionnée par l'une de ces caractéristiques alors les documents de la collections n'ont pas la même probabilité à priori et les documents ne sont pas équiprobables.

Plusieurs caractéristiques ont été utilisées pour estimer la probabilité à priori d'un document comme : la longueur du document, la structure des liens, le rapport Information/Bruit, type d'URL, le facteur temps. Elles expriment qu'un document est plus probable parc qu'il est plus long, plus populaire, plus récent, ou contient plus d'informations que de bruits.

4.1. Formules Générale

La formule que Hiemstra proposée est la suivante :

$$P(Q|D) = P(D) \prod_{i=1}^{n} p(t_i|D) \tag{2.28}$$

Où : P(D) est la probabilité a priori de pertinence du document D.

$\prod_{i=1}^{n} p(t_i|D)$ est le score par le contenu.

$$score\ (D_i Q) = \prod_{i=1}^{n} (t_i, d_1) \cong \frac{t_i}{|d_i|} \tag{2.29}$$

4.2. Information à priori

4.2.1. Taille du document

La probabilité à priori est proportionnelle à la taille du document, telle que :

$$\boldsymbol{P(D) = \frac{|D|}{|C|}}$$

Où : |D| est la taille du document et |C| la taille de la collection.

Un document plus long tend à contenir plus d'informations et par conséquent il est plus probable d'être pertinent. Les résultats obtenus avec l'utilisation de cette caractéristique ont été mixtes selon la collection utilisée [Kraaij, Westerveld & Hiemstra, 2002], [Miller, Leek & Schwartz, 1999].

PDF created with pdfFactory Pro trial version www.pdffactory.com

4.2.2. La structure des liens

Les documents populaires ou les plus cités tendent à être plus pertinents. La méthode utilise généralement le nombre de liens entrants, i.e :

$$\mathbf{P(D)} = \frac{\mathbf{n(l,D)}}{\sum_{\mathbf{D'}} \mathbf{n(l,D')}} \quad (2.30)$$

Où : *n (l,D)* est le nombre de liens entrants.

4.2.2.1. Algorithme HITS (Hyperlinked Induced Topic Search)

Kleinberg fut l'un des premiers à s'interesser aux propriétés de connectivité du graphe représentatif du web et de son apport dans la détection de la pertinence d'une page à une requête [Kleinberg, 1999]. Quelques constatations simples sont à l'origine de ses travaux dans ce domaine.

Dans cette approche, deux types de pages sont identifiés en fonction de la nature de leurs connexions avec les autres documents. On retrouve d'une part les pages qui semblent être très importantes qui jouent le rôle d'autorité sur un sujet donné et d'autre part les documents possédant un grand nombre de liens vers des pages faisant autorité sur un sujet. On distingue ainsi les pages autorités ayant un grand nombre de liens entrants et les pages hubs ayant un grand nombre de liens sortant et regroupant les autorités d'un même sujet. Le but de l'algorithme HITS est de déterminer les hubs et les autorités qui renforcent leurs relations mutuellement sur un sujet donné. Ainsi, Kleinberg dénombre les bons hubs comme des pages pointant vers beaucoup de bonnes autorités et les bonnes autorités comme des pages pointées par beaucoup de bons hubs.

Cette dénotation de bons hubts et de bonnes autorités fait apparaître une troisième catégorie de pages ayant un grand nombre de liens entrants provenant de documents n'ayant aucune particularité. Ces pages, que nous nommons pages indépendantes, sont considérées comme universellement populaires et n'apportent que peu ou pas d'intérêt [Chakrabarti et al.,1999]. Par exemple, la page de Google est extrêmement référencée mais n'apporte que peu d'intérêt sur la plupart des sujets rencontrés où une publicité aura de nombreux liens vers elle.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Cette situation est illustrée dans la figure suivante :

Hubs Autorités Page indépendante

Figure 2.4: Répartition des pages en bon Hubs, bonnes autorités et pages indépendantes.

Une justification intuitive de l'autorité conférée à une page en fonction de la structure des liens l'entourant peut être donnée en considérant qu'un fort taux de jugement humain entoure l'ajout d'un lien hypertexte dans un document. En quelque sorte, l'auteur du document estime que la page vers laquelle il construit un lien évoque un sujet similaire à son souhait et paraît intéressante.

Pour déterminer les hubs et les autorités d'un sujet σ donné, l'algorithme HITS se base sur un sous-graphe du web S_σ qui doit répondre aux conditions suivantes :

- **ü** S_σ est relativement petit
- **ü** S_σ est riche en pages pertinentes
- **ü** S_σ contient la totalité (ou la plupart) des plus importantes autorités

En gardant S_σ petit, l'application d'algorithmes non triviaux peut s'effectuer sans s'occuper du temps de calcul nécessaire à la réalisation de la tâche. Les deux derniers points nous permettent de nous assurer d'avoir de bonnes chances de déterminer les bonnes autorités correspondant à la requête σ.

La mesure « Authority » est calculée comme suit :

$$a(p) = \sum_{q \in S_\sigma, p \to q} h(q)$$

La mesure « hub » est calculée comme suit :

$$h(p) = \sum_{q \in S_\sigma, q \to p} h(q)$$

Où : p ∈ S σ et q- >p est la condition « q contient un lien pointant vers p ».

PDF created with pdfFactory Pro trial version www.pdffactory.com

4.2.2.2. L'Algorithme PageRank

L'idée principale de cette méthode est de simuler le comportement d'un internaute naviguant de manière aléatoire sur le web. La probabilité qu'il visite une page donnée est d'autant plus grande que cette page soit pointée par beaucoup d'autres pages au travers de leurs liens hypertextes. En considérant qu'une page confère une certaine autorité à une autre page en établissant un lien vers elle, la probabilité de passage de cet internaute aléatoire sur une page indique le degré de pertinence de ce document.

Il existe deux manières d'accéder à une page Web. On peut d'une part l'atteindre directement en connaissant son adresse et d'autre part suivre un lien hypertexte d'un autre document. Le calcul du PageRank – et donc de la pertinence – d'une page intègre ces deux éléments au travers d'une probabilité d. d représente en quelque sorte la probabilité que l'internaute aléatoire s'ennuie sur une page et décide de choisir une autre page au hasard.

L'équation suivante permet de calculer le PageRank pour tous les nœuds du graphe représentant le web.

$$PR(p_j)_t = (1-d) + d \sum_{\substack{i=1 \\ pi \to pj}}^{n} \frac{PR(p_i)_{t-1}}{C(p_i)} \qquad (2.31)$$

Où :

$PR(p_j)_t$ *:*représente la valeur du PageRank à l'itération *t* pour la page p_j.

$C(p_i)$ *:*est défini comme le nombre de liens sortants de la page p_i.

Le paramètre d prend ses valeurs dans l'intervalle [0-1] et est généralement placé à d=085 d'après des études statistiques menées par Lary Page dans [Page et al., 1998].

Ce dernier paramètr*e* permet de faire converger l'algorithme de manière plus ou moins rapide. En effet, plus *d* est élevé, plus l'effet de l'ajout d'un lien entrant vers une page est accru et plus celui-ci se propagera dans toutes les pages d'un même site.

Le PageRank forme ainsi une distribution de probabilités des pages Web. Le calcul peut s'effectuer de manière itérative et converge vers une valeur asymptotique de manière assez rapide. En effet, le calcul effectué dans [Page et al.,1998] sur un graphe de 26 millions de nœuds en considérant d=0.85, converge en seulement 52 itérations.

4.2.2.3. Rapport Information/Bruit

Il est définit comme le rapport entre la taille de document après prétraitement (élimination des motsvides et des balises Html) et la taille de document sans prétraitement [Zhu & Gauch, 2000].

$$\boldsymbol{P(D) = \frac{L_{token}}{L_{document}}} \qquad (2.32)$$

PDF created with pdfFactory Pro trial version www.pdffactory.com

Tel que : L_{token} est la taille de document après le prétraitement et $L_{document}$ est la taille de document avant le prétraitement. Ainsi, un document avec moins de mots vides peu de balises Html produit un haut rapport Information/Bruit, ce qui signifie que le document est de « Bonne » qualité.

4.2.2.4. Type d'URL du document

Kraaij, Westerveld et Hiemstra dans [KKraaiji, Westerveld & Hiemstra, 2002] ont utilise e le type d'URL pour estimer la probabilité qu'une page soit une page d'entrée.

$$\boldsymbol{P(D) = P\left(PE \middle| URL_{type}(D) = t_i\right) = \frac{c(PE,t_i)}{c(t_i)}} \quad (2.32)$$

Où :

URL_{type} est le type d'URL de document D,

$c(PE, t_i)$ est le nombre de documents de type d'URL « t_i » qui sont des pages d'entrée « PE » pour un site web obtenu à partir des évaluations de pertinence, c (t_i) est le nombre de documents de type d'URL « t_i ».

Quatre types d'URL ont été définis :

- Racine : elle contient le nom de domaine seulement, par exemple : www.sigir.org
- Sous-racine : elle contient le nom de domaine suivi d'un seul répertoire, par exemple : www.sigir.org/sirgirlist/.
- Chemin (répertoire) : il contient le nom de domaine suivi d'un ou de plusieurs répertoires, par exemple : www.sigir.org/sirgirlist/issues.
- Fichier : tout URL se terminant avec un nom de fichier autre qu'index.html.

PDF created with pdfFactory Pro trial version www.pdffactory.com

5. Conclusion

Dans ce chapitre, on a pu voir comment s'effectue la modélisation de langue en Recherche d'Information de part et ses principes et ses approches, du modèle initialement proposé dans ce domaine [Ponte &Croft, 1998] jusqu'aux modèles développés récemment. Cette étude nous a permis de constater l'engouement de la Recherche pour les modèles de langue.

Elle nous a permis aussi de constater que les approches de modélisation de langue présentées dans ce chapitre, permettent d'intégrer des informations pour conditionner la probabilité à priori de pertinence d'un document.

Dans le chapitre qui suit, on va voir les différents algorithmes de Clustering qui existe, et nous illustrerons chacun d'eux par un exemple.

PDF created with pdfFactory Pro trial version www.pdffactory.com

1. Introduction

L'accroissement du volume de données dans notre environnement et leurs rôles de plus en plus significatifs, nous poussent à utiliser d'avantage les méthodes de traitements automatisés. Cette partie fournit un état de l'art d'une branche spécifique, souvent mentionnée sous le nom de *Clustering* qui s'attache à étudier les problématiques liées au classement de ces données.

L'utilisation du *clustering* est vaste et varie selon le besoin. Mais l'objectif général reste identique : révéler par le *regroupement ou la segmentation* des données une connaissance qui peut être exploitée ultérieurement.

L'utilisation d'une méthode de clustering n'est pas définie comme une approche monolithique mais plutôt comme une série d'étapes. Ainsi, [JD88, And04] propose une liste définissant ces étapes. Nous proposons une liste similaire mais réduite et plus adaptée à la taille de notre problème.

- **Extraction :** extraire les données et en distinguer des objets spécifiques (individus). C'est ici que les données sont préparées, nettoyées.
- **Représentation des données :** obtenir une représentation des données adaptée à l'algorithme, choisir une fonction de similarité suivant la nature de cette représentation, appréhender les différents critères sur ces données.
- **Stratégie de classement :** choisir l'algorithme suivant les exigences (performance, représentation des résultats, type de données) et fixer les éventuels paramètres.
- **Validation :** estimer la qualité de la construction produite (par un expert ou en utilisant une métrique adaptée).
- **Interprétation et utilisation :** juger si la classification répond bien aux besoins exprimés en utilisant la classification avec les données réelles si cela n'a pas déjà été fait auparavant.

PDF created with pdfFactory Pro trial version www.pdffactory.com

2. L'apprentissage Automatique

L'apprentissage automatique (machine-learning en anglais) se trouve au carrefour de nombreux domaines : intelligence artificielle, statistiques, sciences cognitives, théorie des probabilités, de l'optimisation, du signal et de le recherche d'information... Il est donc bien difficile de donner une taxinomie des techniques d'apprentissages.

L'apprentissage automatique consiste à tirer des règles générales à partir d'observations particulières. L'apprentissage automatique a pour objectif d'extraire et d'exploiter automatiquement l'information présente dans un jeu de données. En cela il couvre un vaste champ d'objectifs comme la fouille de données, la classification, la sélection de variables, la discrimination, la régression, la sélection de modèle, la génération et l'inférence de règles, etc. Il s'avère également être fortement pluridisciplinaire puisque selon les données et les objectifs.

Alors on peut dire que l'apprentissage automatique c'est la Capacité d'un système à améliorer ses performances via des interactions avec son environnement.

On distingue trois types d'apprentissages automatiques :

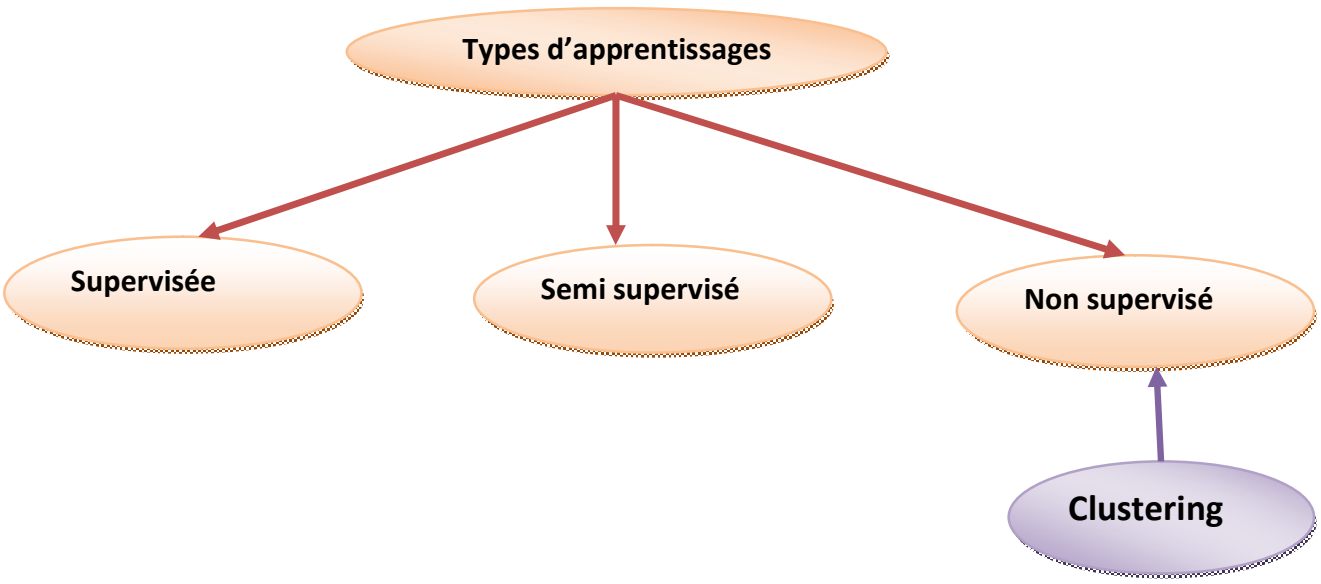


Figure 3.1 : les types d'apprentissage automatique.

2.1. Apprentissage supervisé

L'apprentissage supervisé est une technique d'apprentissage automatique où l'on cherche à produire automatiquement des règles à partir d'une base de données d'apprentissage contenant des *« exemples »* (en général des cas déjà traités et validés).

L'apprentissage supervisé consiste à inférer un modèle de prédiction à partir d'un ensemble d'apprentissage, c'est-à-dire plusieurs couples de la forme {observation, étiquette}, où chaque étiquette dépend de l'observation à laquelle elle est associée.

PDF created with pdfFactory Pro trial version www.pdffactory.com

L'apprentissage supervisé suppose qu'un oracle fournit les étiquettes de chaque donnée d'apprentissage. On distingue en général trois types de problèmes auxquels l'apprentissage supervisé est appliqué : la classification supervisée, la régression, et les séries temporelles. Ces trois types de problèmes se différencient en fonction du type d'étiquettes fournit par l'oracle. Dans notre chapitre, nous ne nous intéresserons qu'à la classification. Pour ce problème, les étiquettes sont des classes.

Quelques algorithmes d'apprentissage supervisé :

La plupart des algorithmes d'apprentissage supervisés tentent de trouver un modèle (une fonction mathématique) qui explique le lien entre des données d'entrée et les classes de sortie. Ces jeux d'exemples sont donc utilisés par l'algorithme.

Il existe de nombreuses méthodes d'apprentissage supervisé :

- Méthode des k plus proches voisins.
- Machine à vecteurs de support (SVM) .
- Mélanges de lois.
- Réseau de neurones.
- Arbre de décision.
- Classification naïve bayesienne.
- Inférence grammaticale.

2.2. Apprentissage semi-supervisé

L'apprentissage semi-supervisé est une classe de techniques d'apprentissage automatique qui utilise un ensemble de données étiquetées et non-étiquetés. Il se situe ainsi entre l'apprentissage supervisé qui n'utilise que des données étiquetées et l'apprentissage non-supervisé qui n'utilise que des données non-étiquetées [Chapelle, O. et al, 2006]. Il a été démontré que l'utilisation de données non-étiquetées, en combinaison avec des données étiquetées, permet d'améliorer significativement la qualité de l'apprentissage. Un autre intérêt provient du fait que l'étiquetage de données nécessite l'intervention d'un utilisateur humain. Lorsque les jeux de données deviennent très grands, cette opération peut s'avérer fastidieuse. Dans ce cas, l'apprentissage semi-supervisé, qui ne nécessite que quelques étiquettes, revêt un intérêt pratique évident.

PDF created with pdfFactory Pro trial version www.pdffactory.com

2.3. Apprentissage non supervisé

Apprentissage non supervisé est un outil très important en analyse exploratoire de données non étiquetées. Il est utilisé pour la détection de regroupement, lorsque l'on n'a pas d'informations a priori sur la structure interne de ces données. Un problème de regroupement peut être défini comme le partitionnement d'un ensemble d'éléments en plusieurs sous groupes pertinents, en général mutuellement disjoints, *les clusters*.

Les méthodes de classification non supervisée jouent un rôle très important dans la compréhension de phénomènes variés décrits par des bases de données
Dans notre mémoire on s'intéresse à l'apprentissage non supervisé.

3. Quelques bonnes raisons de s'intéresser à l'apprentissage non supervisé

L'apprentissage non supervisé peut être intéressant de découvrir de l'information sur un grand nombre de données non annotées, et d'utiliser ensuite les méthodes supervisées seulement sur les *clusters* trouvés, on trouve d'autre raisons comme :

- Profusion d'enregistrements et de variables.
- Constituer des échantillons d'apprentissage étiquetés peut être très couteux.
- Découvertes sur la structure et la nature des données à travers l'analyse exploratoire.
- Utile pour l'étude de caractéristiques pertinentes.
- Prétraitement avant l'application d'une autre technique de fouille de données.
- de meilleurs résultats peuvent être obtenus à l'aide d'une méthode non-supervisée dans le cas où les motifs changent doucement avec le temps,
- les méthodes non-supervisées permettent de découvrir des informations de nature et de structure des données utiles.

4. Les différentes approches à la classification non supervisé

Devant un problème défini de façon aussi imparfaite, il était naturel de voir apparaitre un grand nombre de technique, souvent à fort parfum heuristique. On peut aujourd'hui les regrouper en deux grandes familles : la classification par partition, et la classification hiérarchique.

4.1. Classification par partition

Ø Partitionnement « dur »

L'idée générale est de découper l'espace des observations en un certain nombre de régions disjointes, définies par des frontières, et de décréter que toutes les observations situées dans une même région de l'espace appartiennent à une même classe. Chaque classe est représentée par un ''prototype'', observation virtuelle sensée être la plus représentative de la population de la classe. Le prototype d'une classe sera le plus souvent le barycentre des observations de la classe.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Ces prototypes sont positionnés de façon itérative dans les zones à forte densité, et les observations sont affectées aux classes sur la base d'un critère de proximité aux différents prototypes.

Ø Partitionnement « doux »

L'idée selon laquelle chacune des classes réelles, sous-jacentes, occupe une région limitée de l'espace peut paraître irréaliste. En particulier, l'Analyse

Discriminante nous a habitués à penser en termes de classes ayant des distributions multi-normales, et donc se chevauchant nécessairement. Il est donc naturel de considérer la possibilité que les classes empiètent les unes sur les autres.

4.2. Classifications hiérarchiques

La " classification hiérarchique" est une famille de techniques qui génèrent des suites de partitions emboîtées les unes dans les autres, et allant depuis la partition triviale à une seule classe (contenant toutes les observations) jusqu'à la partition triviale où chaque observation est une classe. Entre ces deux extrêmes figurent de nombreuses partitions plus réalistes entre lesquelles l'analyste devra choisir.

- **ü** Méthodes descendantes -*divisive*-
- **ü** Méthodes ascendantes -*agglomerative*-

5. Clustering

5.1. Définition d'un clustering

Le clustering, en français *regroupement* ou *partitionnement*, est une tâche dont l'objectif est de trouver des groupes au sein d'un ensemble d'éléments (appelés par la suite objets). Ces objets sont décrits par des caractéristiques, encore appelées attributs, qui décrivent les propriétés des objets. Les groupes recherchés, communément appelés des clusters, forment des ensembles homogènes d'objets du jeu de données partageant des caractéristiques communes.

Clustering est une action de découper un ensemble d'objets en groupes (clusters) de telle sorte que les caractéristiques des objets dans un même cluster soient similaires et que les caractéristiques des objets dans des clusters différents soient distinctes.

Les classes de la classification regroupent les objets ayant des caractéristiques similaires et séparent les objets ayant des caractéristiques différents (homogénéité interne et hétérogénéité externe) c'est un apprentissage non supervisé.

Le clustering est une problématique de recherche étudiée depuis de nombreuses années dans différentes communautés**:** machine learning, data mining, pattern recognition, statistiques, …etc. Son objectif, très général, consiste à séparer un ensemble d'objets en

PDF created with pdfFactory Pro trial version www.pdffactory.com

différents groupes (ou clusters) en fonction d'une certaine notion de similarité. Les objets qui sont considérés comme similaires sont ainsi associés au même cluster alors que ceux qui sont considérés comme différents sont associés à des clusters distincts.

5.2. But de Clustering

Le clustering (regroupement) des documents vise à mettre les documents similaires ensemble. En ce faisant, on veut atteindre un des buts suivants:

- Le nombre de clusters, par rapport au nombre de documents, est beaucoup plus petit. Ainsi, on peut accélérer le processus de recherche.
- Si on document est pertinent à une requête, alors les documents similaires ont plus de chance à être pertinents aussi. Ainsi, les clustering peuvent être aussi vu comme un moyen d'expansion.
- Finalement, les réponses du système peut être regroupées, plutôt qu'être mises dans une liste individuellement. L'avantage de cette présentation de résultats est que l'utilisateur peut avoir une idée globale des résultats que le système a trouvés assez rapidement.

Avec le progrès rapide sur les matériels d'informatique, le premier avantage semble beaucoup moins important maintenant. Les deux autres restent toujours d'actualité.

5.3. L'évaluation d'un système de clustering

L'évaluation d'un système de clustering n'est pas simple. En effet, comment évaluer le fait qu'un ensemble de documents a été découpé en clusters pertinents. La méthode d'évaluation la plus objective fait appel à un expert qui s'occupe de juger la pertinence du clustering *a posteriori*. C'est une méthode très coûteuse qui ne peut-être utilisée que pour des applications spécifiques. Dans le cas de l'utilisation du clustering pour la RD, l'évaluation courante consiste à mesurer la précision du système sans utilisation des clusters puis avec l'utilisation des clusters. Si la précision est plus importante avec les clusters, on peut considérer que le clustering est alors de bonne qualité. D'autres critères peuvent être utilisés comme par exemple la vitesse de réponse du système de RD. Cependant ce type d'évaluation est dépendant de la tâche fixée.

Parmi les mesures indépendantes d'une tâche donnée, on peut citer les mesures d'homogénéité comme par exemple la dispersion intra cluster au sens d'une mesure de similarité (ou d'une distance) donnée. Nous allons présenter ici trois critères usuels indépendants d'une application spécifique : l'entropie, la pureté et l'information mutuelle.

Pour être calculés, ces critères nécessitent d'avoir à disposition un corpus étiqueté en classe. Ils vont alors évaluer dans quelle mesure le système de clustering est capable de retrouver des clusters « en accord » avec l'étiquetage du corpus.

PDF created with pdfFactory Pro trial version www.pdffactory.com

5.3.1. Pureté

La pureté d'un cluster permet de mesurer le pourcentage de documents du cluster appartenant à la classe majoritaire :

$$pureté(cl) = \frac{nombre\ de\ documents\ du\ cluster\ de\ la\ classe\ c^{cl}}{nombre\ de\ documents\ du\ cluster}$$

Ainsi, la pureté d'un cluster est égale à 1 si tous les documents du cluster sont issus de la même classe. Dans le cas extrême où l'on a un cluster par document, la pureté est alors de 100%. C'est pour cela que, pour comparer des systèmes de clustering, il faut comparer la mesure de pureté (entropie, information mutuelle) à nombres de clusters égaux. Nous calculerons comme mesure globale la pureté *micro moyenne* et la pureté *macro moyenne.*

5.3.2. Entropie

L'entropie est une autre mesure de la qualité d'un cluster. Elle mesure comment les différentes classes de documents sont reparties dans un cluster.

$$entropie(cl) = -\frac{1}{\log/C/}\sum_{i=1}^{/C/}\frac{n_i^{cl}}{n^{cl}}\log\frac{n_i^{cl}}{n^{cl}} \qquad (3.1)$$

Où n^{cl} est le nombre de documents du cluster *cl*
et n_i^{cl} est le nombre de documents de la classe i dans le cluster cl.
La synthèse sur l'ensemble des clusters sera obtenue par le calcul de l'entropie *micro moyenne* et de l'entropie *macro moyenne.* L'entropie idéale est une entropie nulle. Plus l'entropie est faible, meilleur est le clustering. Tout comme la pureté, les entropies de deux systèmes doivent être comparées à nombres de clusters égaux.

5.3.3. Information mutuelle

Enfin, la troisième mesure utilisée sera l'information mutuelle. Elle mesure l'indépendance entre deux variables. Dans le cas du clustering, elle mesure l'indépendance entre les variables de classe et les variables de cluster. Elle se calcule de la manière suivante :

$$I(C, CL) = \sum_{\substack{c \in C \\ cl \in CL}} p(c, cl)\log\frac{P(c,cl)}{P(c)P(cl)} \qquad (3.2)$$

Un clustering de bonne qualité présentera une forte information mutuelle. Cette information mutuelle est aussi directement corrélée au nombre des clusters.

5.4. Applications possibles

Algorithmes de classification peuvent être appliqués dans de nombreux domaines, par exemple:

PDF created with pdfFactory Pro trial version www.pdffactory.com

- *Marketing:* trouver des groupes de clients avec un comportement similaire donné une grande base de données clients contenant leurs propriétés et les dossiers d'achat antérieurs;
- *Biologie:* classification des plantes et des animaux compte tenu de leurs caractéristiques;
- *Bibliothèques:* Commande de livres;
- *Assurance:* identifier les groupes de détenteurs de polices d'assurance automobile avec un coût de réclamation moyenne élevée; fraudes identification;
- *Urbanisme:* l'identification des groupes de maisons en fonction de leur type de maison, la valeur et l'emplacement géographique;
- *études sismiques:* regroupement des épicentres des séismes observés pour identifier les zones dangereuses;
- *WWW:* la classification des documents, des données de weblog de clustering de découvrir des groupes de modes d'accès similaires.

5.5. Types de données pour l'analyse de clusters

Cinq types différents de variables nécessitent des traitements différents :

- ü **Numériques linéaires** (continue sur un intervalle)

Ex : poids, taille, longitude, latitude, etc.

- ü **Binaires** : une valeur parmi deux possibles

Ex : 0 : la variable est absente, 1 : la variable est présente

- ü **Nominale** : valeur prise dans une liste finie

Ex : couleur : « vert, bleu, rouge, jaune, noir »

- ü **Ordinales** : l'ordre des valeurs est plus important

Ex : résultat d'un concours

- ü **Ratios** (à échelle variable) : variables numériques sur une échelle exponentielle

Ex : croissance optimale des bactéries, durée de la radioactivité.

5.6. Notion de similarité

5.6.1. Vocabulaire

- Ø Mesure de dissimilarité DM : plus la mesure est faible plus les points sont similaires.
- Ø Mesure de similarité SM : plus la mesure est grande, plus les points sont similaires
- Ø DM = borne – SM

PDF created with pdfFactory Pro trial version www.pdffactory.com

5.6.2. Notations

Observation : $x_i \in \mathbb{R}^d$

Avec $x_i = \begin{pmatrix} x_{i,1} \\ x_{i,2} \\ \vdots \\ x_{i,d} \end{pmatrix}$

5.6.3. Mesure de la distance $d(x_1, x_2)$ entre 2 points x_1 et x_2

- Ø Distance de Minkoswski :

$$d(x_1, x_2) = (\sum_{j=1}^{d} |x_{1,j} - x_{2,j}|^q)^{\frac{1}{q}}$$

- Ø Distance Euclidienne correspond à $q = 2$:

$$d(x_1, x_2) = \sqrt{\sum_{j=1}^{d}(x_{1,j} - x_{2,j})^2} = \sqrt{(x_1 - x_2)^t(x_1 - x_2)}$$

- Ø Distance de Manhattan $(q = 1)$: $d(x_1, x_2) = Pd$

$$d(x_1, x_2) = \sum_{j=1}^{d} |x_{1,j} - x_{2,j}|$$

- Ø distance de Sebestyen :

$$d^2(x_1, x_2) = (x_1 - x_2)^t W (x_1 - x_2)$$

- Ø distance de Mahalanobis :

$$d^2(x_1, x_2) = (x_1 - x_2)^t C^{-1} (x_1 - x_2)$$

5.6.4. Mesure de la distance $D(C_1, C_2)$ entre 2 classes C_1 et C_2

- v plus proche voisin :

$$D_{min}(C_1, C_2) = min\{d(x_i, x_j), x_i \in C_1, x_j \in C_2\}$$

- v diamètre maximum :

$$D_{max}(C_1, C_2) = max\{d(x_i, x_j), x_i \in C_1, x_j \in C_2\}$$

- v distance moyenne :

$$D_{moy} = (C_1, C_2) = \frac{\sum_{x_i \in C_1} \sum_{x_j \in C_2} d(x_i, x_j)}{n_1 n_2}$$

- v distance des centres de gravité :

$$D_{cg} = d(\mu_1, \mu_2)$$

PDF created with pdfFactory Pro trial version www.pdffactory.com

- distance de Ward :

$$D_{Ward}(C_1, C_2) = \sqrt{\frac{n_1 n_2}{n_1 + n_2}}\, d(\mu_1, \mu_2)$$

5.7. Mesures de similarités

Une bonne méthode de clustering est une méthode qui maximise la ressemblance entre les données à l'intérieur de chaque cluster, et minimise la ressemblance entre les données des clusters différents. C'est pourquoi les résultats d'une technique de clustering dépendent fortement de la mesure de similarité choisie par son concepteur, qui doit la choisir avec prudence. En effet la mesure de similarité repose sur le calcul de la distance entre deux données, sachant que chaque donnée est composée d'un ensemble d'attributs numériques et/ou catégoriels. Plus la distance est importante, moins similaires sont les données et vice versa. Soit x_i et x_j deux données différentes dont on veut calculer la distance. Cette distance est composée d'une part de la distance entre les valeurs des attributs numériques et d'une autre part de la distance entre les valeurs des attributs catégoriels ou symboliques, en prenant bien sur en considération le poids (le nombre) de chaque type d'attributs.

5.7.1. Attributs numériques

Pour mesurer la distance entre les données à attributs numériques, plusieurs formules existent:

- La distance Euclidienne :

$$D_n(x_i, x_j) = \sqrt{\sum_{k=1}^{n_n} (x_{ik} - x_{jk})^2} \qquad (3.3)$$

- La distance City blocs :

$$D_n(x_i, x_j) = \sum_{k=1}^{n_n} |x_{ik} - x_{jk}| \qquad (3.4)$$

- La distance de Minkowksi :

$$D_{np}(x_i, x_j) = \left(\sum_{k=1}^{n_n} (x_{ik} - x_{jk})^2\right)^2 \qquad (3.5)$$

Il faut faire attention lors du calcul de ces distances à la normalisation des attributs, puisque les intervalles de variances des attributs peuvent être très différents, ce qui peut entraîner la dominance d'un ou de quelques attributs sur le résultat. Il est conseillé donc, de normaliser tous les attributs sur le même intervalle puis calculer la distance.

5.7.2. Attributs catégoriels

Le problème qui se pose lors du calcul de la distance entre les attributs catégoriels, c'est qu'on ne dispose pas d'une mesure de différence. La seule mesure qui existe, dans l'absence de toute information sur la signification des valeurs, c'est l'égalité ou l'inégalité.
La distance utilisée est alors :

PDF created with pdfFactory Pro trial version www.pdffactory.com

—

(3.6)

Il faut en fin la normaliser avec les attributs numériques et le nombre d'attributs catégoriels. La distance entre deux données et , composées d'attributs numériques et catégoriels, est donc :

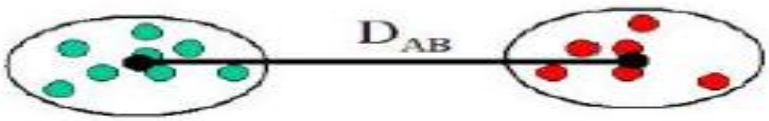
En se basant sur la distance entre deux attributs, plusieurs distances peuvent être calculées :
– Distance entre deux clusters : permet de mesurer la distance entre deux clusters pour une éventuelle fusion en cas où ils soient trop proches. Cette distance peut être prise entres les centres des deux clusters, entre les deux données les plus éloignées (ou plus proches) des deux clusters ou la distance moyenne de leurs données.

complete linkage :

- plus petite similarité/plus grande distance entre toutes les paires de gènes entre 2 clusters.

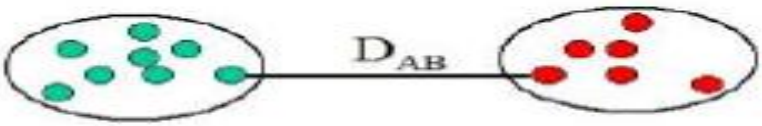
average linkage :

- similarité moyenne entre les paires de gènes.



single linkage

- plus grande similarité/plus petite distance entre 2 gènes de 2 clusters.



5.8. Méthodes de clustering

La classification non supervisée (ang.Clustering) consiste à diviser un ensemble de données D en sous-ensembles, appelés classes (ang.Cluster), de sorte que les classes soient les plus homogènes possible suivant un critère défini. Les critères les plus couramment utilisé sont la similarité entre objets, la densité des classes ou des mesures probabilistes. Les objets sont regroupés selon le critère qu'utilise la méthode classification employée.
Cette technique apporte deux avantages majeurs :

- Une réduction significative de la complexité temporelle de la recherche sur de grandes bases documentaires.
- Les documents retournés à l'utilisateur sont tous pertinents du fait qu'ils possèdent des caractéristiques similaires voir même identiques pour une requête donnée.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Quelques méthodes parmi les plus utilisées dans ce type de classification :

- **Algorithmes hiérarchiques :** décomposition/composition hiérarchique de clusters (CURE, BIRCH)
- **Algorithmes par partitionnement :** regroupement itérative avec amélioration par replacement des objets (k-means, kmediodes)
- **Fonctions de densité** (cluster avec forme arbitraire) : clusters grandissent aussi longtemps que la densité des objets dans leur voisinage est supérieure a une borne (DBSCAN, OPTICS)
- **Grilles :** l'espace est divise en cellules qui forment une grille (STING, CLIQUE)
- **Modèles :** chaque cluster est suppose suivre un modèle : trouver la meilleure correspondance entre les modèles et les données (COBWEB).

5.8.1. Méthode Hiérarchiques

Le principe de base des méthodes de classification hiérarchique est extrêmement simple, à chaque fois de créer une nouvelle partition en regroupant les deux éléments les plus proches en une seule classe.

Ce type de clustering essaie de créer une hiérarchie des clusters, les documents les plus similaires sont regroupés dans des clusters aux plus bas niveaux, selon comment la hiérarchie est créée,

Dans ce type de clustering le nombre de clusters ne peut être connu à l'avance. Le système prend en entrée l'ensemble de données et fournit en sortie une arborescence de clusters.

On distingue alors deux types de hiérarchie

- les algorithmes divisibles qui commencent à partir d'un ensemble de données et le subdivisent en sous ensembles puis subdivisent chaque sous ensemble en d'autres plus petits, et ainsi de suite, pour générer en fin une séquence de clusters ordonnée du plus général au plus fin.
- La deuxième classe est celle des algorithmes agglomiratifs qui considèrent chaque enregistrement comme étant un cluster indépendant puis rassemblent les plus proches en des clusters plus importants, et ainsi de suite jusqu'à atteindre un seul cluster contenant toutes les données.

Le schéma suivant illustre ces deux types de hiérarchie :

PDF created with pdfFactory Pro trial version www.pdffactory.com

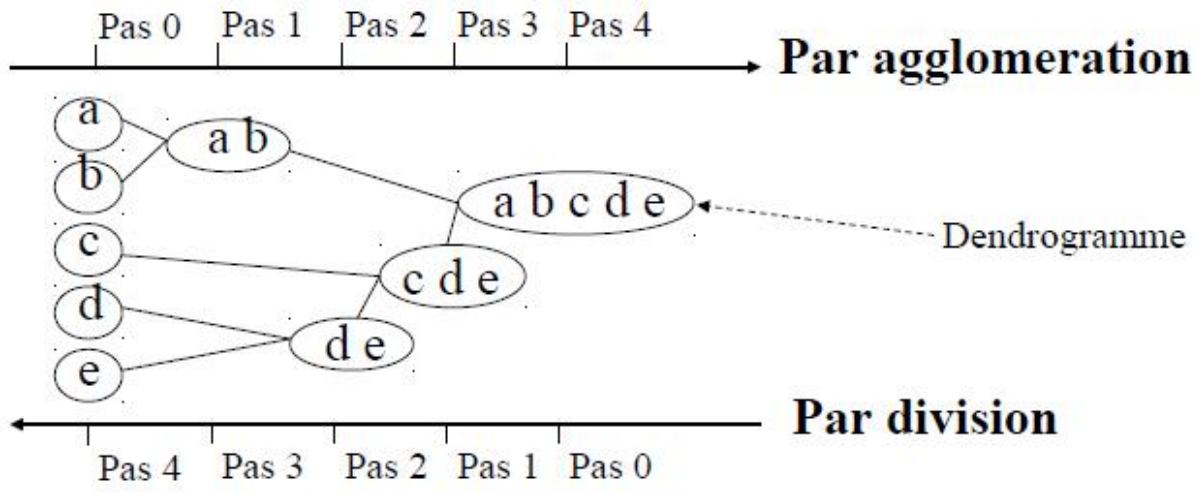


Figure 3.2: Classification hiérarchique.

- Utilisation d'une matrice de distance : ne nécessite pas de spécifier le nombre de clusters
- Paramètre : condition de terminaison (nombre de clusters souhaites, …)

5.8.1.1. Classification hiérarchique Ascendante (CHA)

Le but de la CHA est d'obtenir une classification automatique de l'ensemble d'individus, elle commence par déterminer parmi les n individus, quels sont les 2 individus qui se ressemblent le plus par rapport à l'ensemble des p variables spécifiées pour former une classe. Il existe donc à ce niveau (n-1) classes, une étant formé des 2 individus regroupés précédemment, les autres ne contenant qu'un unique individu. Le processus se poursuit en déterminant quelles sont les 2 classes qui se ressemblent le plus, et on les regroupe. Cette opération est répétée jusqu'à l'obtention d'une unique classe regroupant l'ensemble des individus.

5.8.1.1.1. Les critères d'agrégations

De nombreux critères d'agrégation on été proposés, parmi les plus connus on cite :

Ø Le critère du saut minimal :

La distance entre 2 classes C1 et C2 est définie par la plus courte distance séparant un individu de C1 et un individu C2.

Ø Le critère du saut maximal :

La distance entre 2 classes C1 et C2 est définie par la plus grande distance séparant un individu de C1 et un individu de C2.

Ø Le critère de la moyenne :

PDF created with pdfFactory Pro trial version www.pdffactory.com

Ce critère consiste à calculer la distance moyenne entre tous les éléments du C1 et tous les éléments du C2.

$$D(C1, C2) = \frac{1}{n_{c1}\, n_{c2}} \sum_{x \epsilon C1} \sum_{y \epsilon C2} d(x, y) \qquad (3.7)$$

Avec : n_{c1} et n_{c2} : le cardinal de C1 et C2 respectivement.

5.8.1.1.2. L'Algorithme

L'initialisation :

- Chaque individu est placé dans son propre cluster.
- Calcul de la matrice de ressemblance M entre chaque couple de cluster (ou bien entre chaque point).

Répéter :

- Sélection dans M des deux clusters les plus proches C_i et C_j.
- Fusion $C_i$de et C_j pour créer un cluster C_G.
- Mise à jour de M en calculant la ressemblance entre C_G et les clusters existants.

Jusqu'à :

- La fusion des deux derniers clusters.

5.8.1.1.3. Exemple

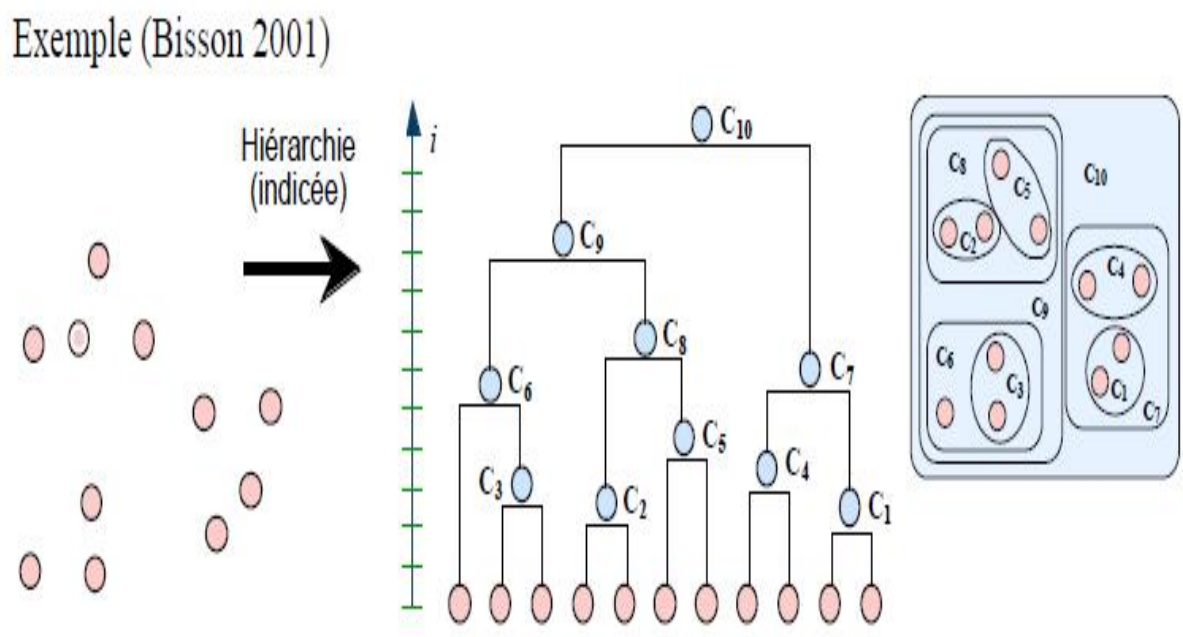


Figure 3.3: exemple de CHA.

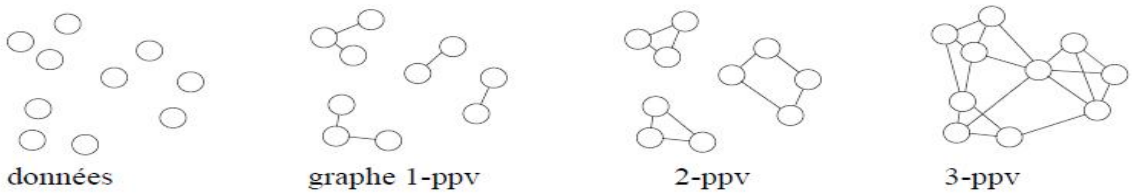
PDF created with pdfFactory Pro trial version www.pdffactory.com

5.8.1.1.4. Méthode Chameleon

C'est une méthode de classification hiérarchique ascendante plus évoluée.

a) Principe : c'est estimer la densité intra-clsuter, et inter-cluster à partir des graphes des K plus proches voisins.

b) Exemple :



c) Algorithme : l'Algorithme de Chameleon se déroule selon deux phases :

- Trouver des sous-clusters initiaux.
- Fusionner dynamiquement les sous-clusters.

d) Similarité entre deux clusters (Chameleon)

1. **Inter-connectivité :**

- **Inter-connectivité absolue entre 2 clusters :**

$EC(C_i, C_j)$ = ensemble des points qui relient des noeuds de C_i et de C_j

- **Inter-connectivité interne d'un cluster :**

$EC(C_i)$ = plus petit ensemble qui *partionne* C_i *en 2 sous-clusters de tailleproche.*

- **Inter-connectivité relative :**

$$RI(CI,CJ) = \frac{2*|EC(Ci,Cj)|}{|EC(Ci)|+|EC(Cj)|} \qquad (3.8)$$

2. **Proximité :**

- **Proximité absolue entre 2 clusters :**

$EC(C_i, C_j)$ = distance moyenne entre les points de $EC(C_i, C_j)$.

- **Proximité interne d'un cluster :**

$EC(C_i)$ = distance moyenne entre les points de $EC(C_i)$.

- **Proximité relative :**

$$RC(C_i, C_j) = \frac{(|C_i|+|C_j|)EC(C_i,C_j)}{|C_i|EC(C_i)+|C_j|EC(C_j)} \qquad (3.9)$$

PDF created with pdfFactory Pro trial version www.pdffactory.com

e) Résultat de Chameleon

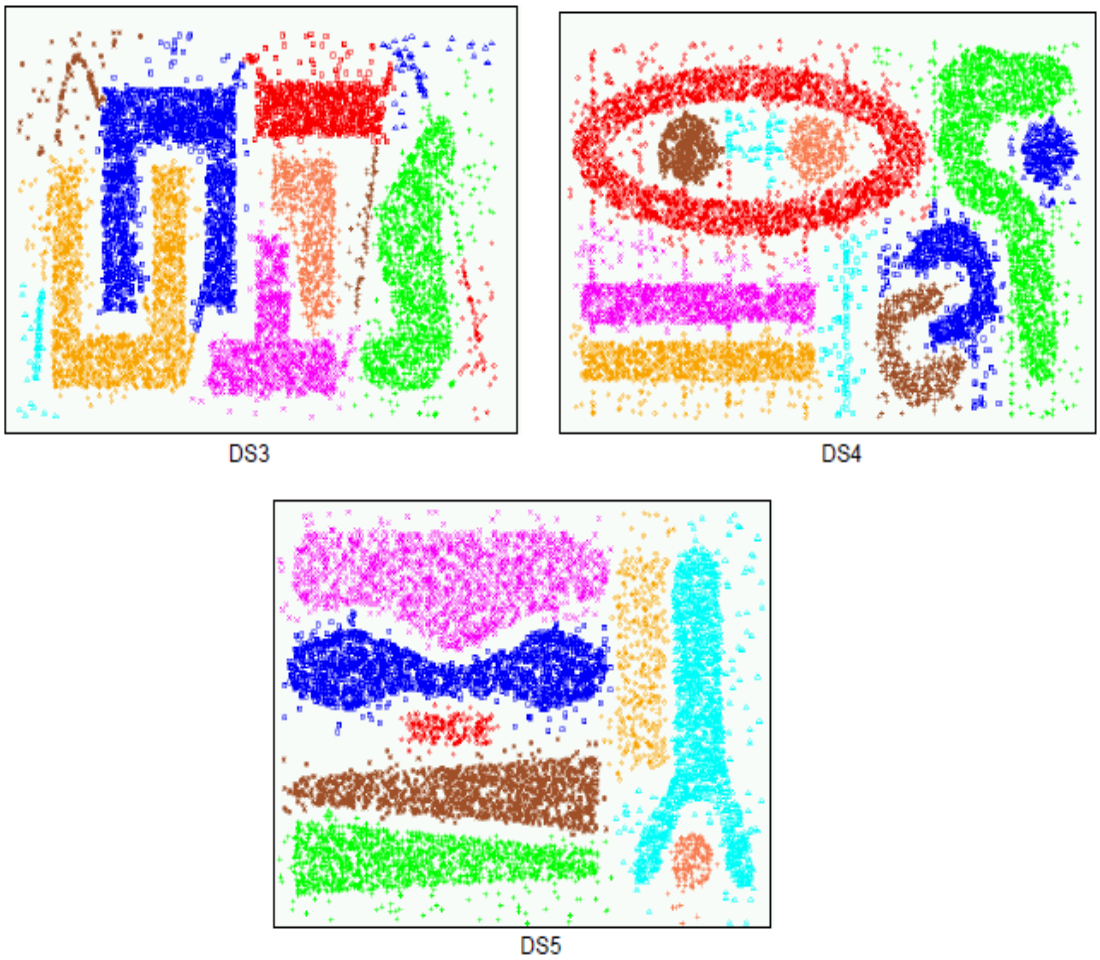


Figure 3.4: Résultat de l'algorithme de Chameleon.

5.8.1.2. Classification Hiérarchique Descendant (CHD)

Les méthodes de classifications hiérarchiques descendante partent d'un ensemble d'individus I et construisent, de manière itérative une partition de l'ensemble d'individus.

a) Principe

- Sélection du cluster le moins cohérant.
- Subdivision du cluster le moins cohérant.

b) Problèmes

- Pour réaliser la subdivision, il faut souvent faire une classification hiérarchique ascendante pour savoir quelle est la meilleure façon de séparer les points.
- Algorithme beaucoup moins attractif que l'autre.

PDF created with pdfFactory Pro trial version www.pdffactory.com

5.8.1.3. Avantages et Inconvénients de la classification hiérarchique

Avantage

- Flexibilité concernant le niveau de granularité.
- Facilité de manipuler toute forme de similitude ou de distance.
- Application à tout type d'attribut.

Inconvénients

- La difficulté de choisir la droite arrêtant des critères.
- La plupart des algorithmes hiérarchique ne révisent pas des classes (d'intermédiaire) une fois ils sont construits.

5.8.2. Classification par partitionnement

Ces méthodes visent à partitionner un ensemble de X de N objets $\{x_1, x_2, \dots, x_n\}$ en K classes (K fixé par l'utilisateur), de tel sorte que chaque classe C_k ayant N_k membre est définie par un prototype appelé centre de la classe (cluster) avec :

$$m_k = \frac{1}{N} \sum_{x \epsilon C_k} x.$$

5.8.2.1. Méthode K-means

L'Algorithme K-means [Pedrycz, W. 2007]., créé par MacQueen [Celeux et al,.1989] en 1967 est l'algorithme de Clustering le plus connu et le plus utilisé car il s'avère être très simple à mettre en œuvre et efficace. Il suit une procédure simple de classification d'un ensemble d'objets en un certain nombre K de clusters, K fixé à priori. Dans cette algorithme, chaque cluster est caractérisé par son centre qui se trouve être la moyenne des éléments composant le cluster.

5.8.2.1.1. L'Algorithme de K-means s'exécute en 5 étapes

- Choisir au hasard le centre de chacune des K classes
- Affecter chaque élément à la classe dont le centre lui est le plus proche (en utilisant par exemple une distance euclidienne)
- déplacer chaque centre vers la moyenne des éléments de la classe
- répéter de 2 à 3 jusqu'à convergence

PDF created with pdfFactory Pro trial version www.pdffactory.com

5.8.2.1.2. Organigramme de l'algorithme de k-means

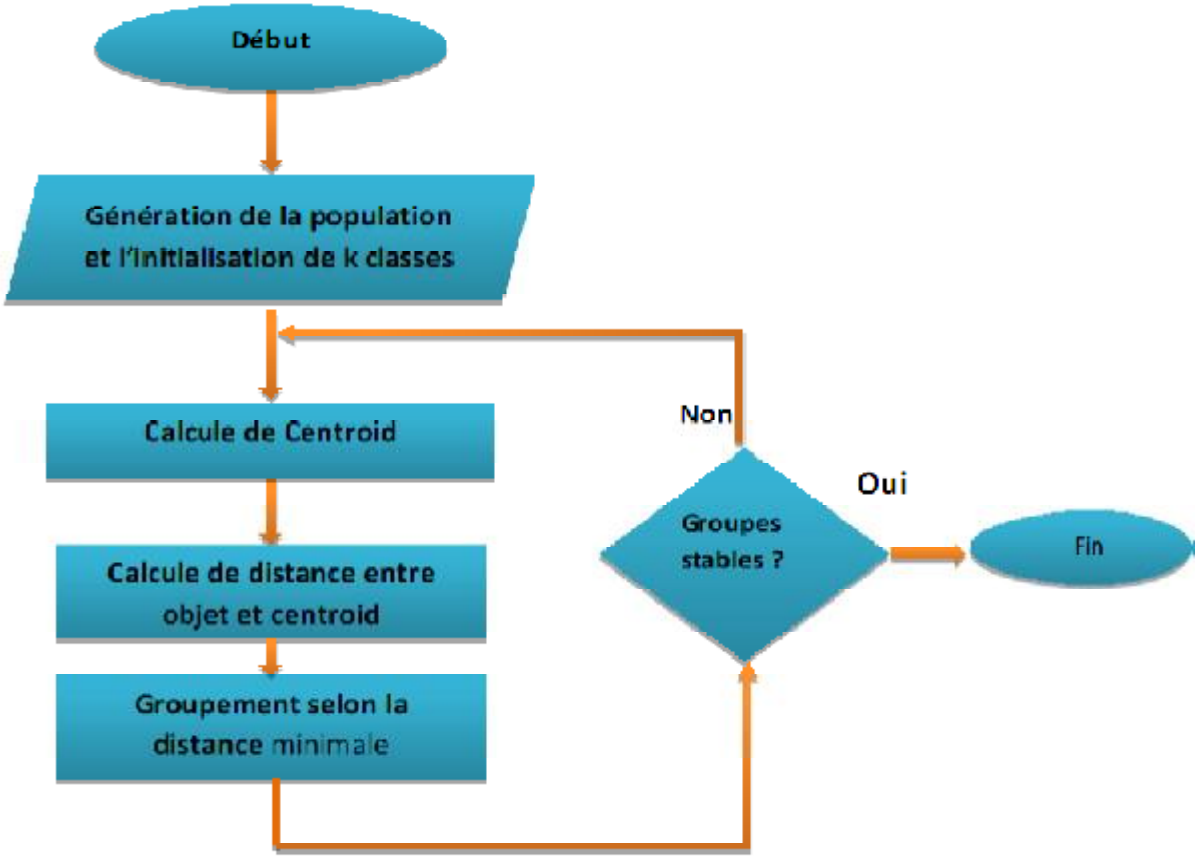
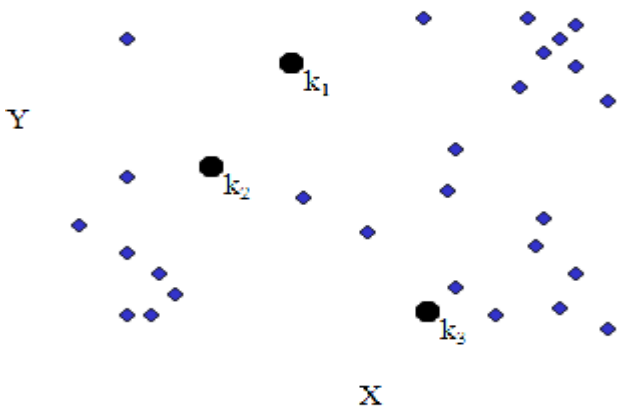


Figure 3.5 : Organigramme de l'algorithme de k-means

5.8.2.1.3. K-means: illustration

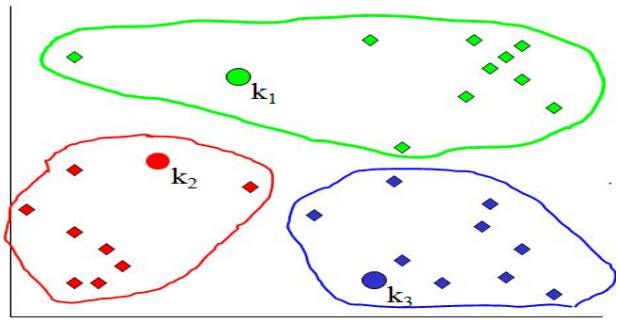
Etape 1:



Choisir 3 centres de classes (au hasard)

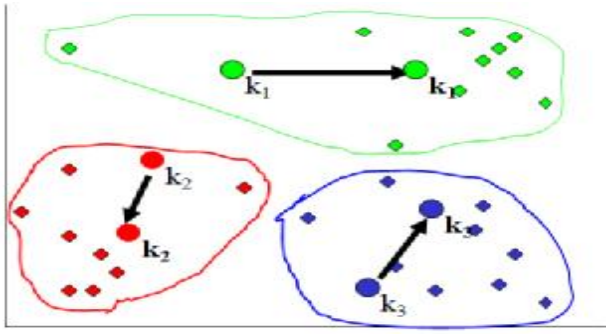
PDF created with pdfFactory Pro trial version www.pdffactory.com

Etape 2 :



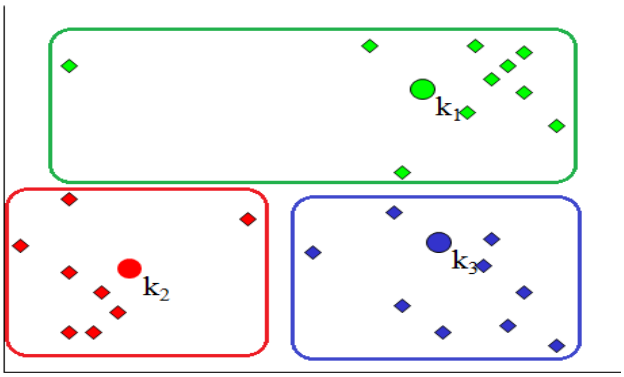
Affecter chaque point à la classe dont le centre est le plus proche

Etape3 :



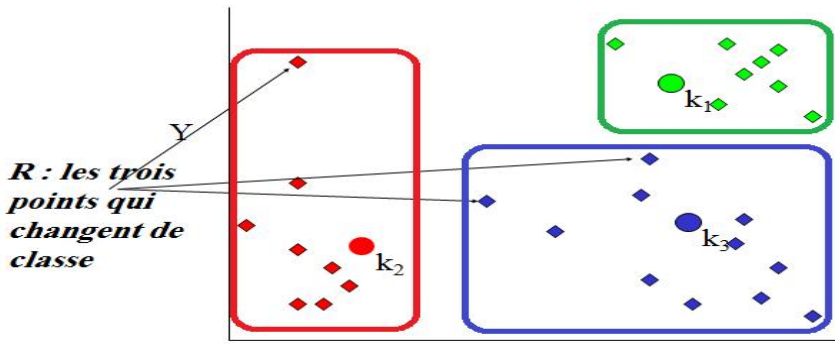
Déplacer chaque centre de classe vers la moyenne de chaque classe

Etape 4 :



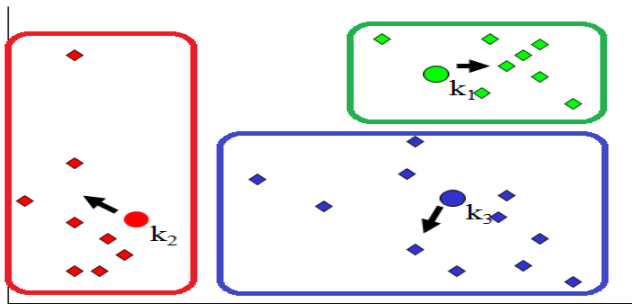
Réaffecter les points qui sont plus proches du centre d'une autre classe

Q : Quels sont les points qui changent de classe?



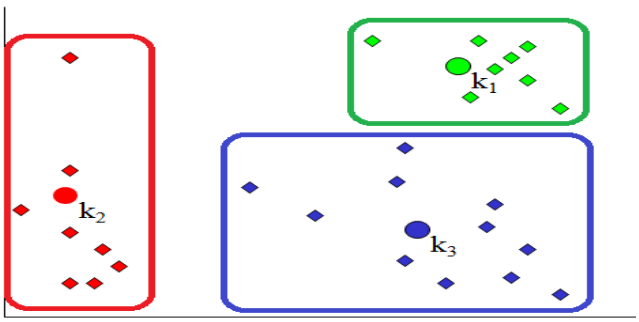
PDF created with pdfFactory Pro trial version www.pdffactory.com

Etape 4 :



Re-calculer les moyennes des classes

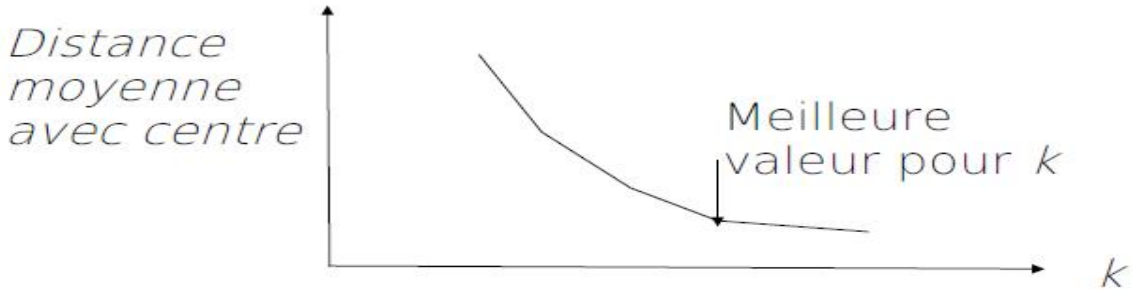
Etape 5 :



Déplacer les centres des classes vers les moyennes

Comment choisir k?

Essayer avec des *k* differents et mesurer les changements de la distance moyenne avec le centre.



5.8.2.1.4. Les avantages et les inconvénients

Avantage :

- Ø L'algorithme de k-means est très populaire du fait qu'il est très facile à comprendre et à mettre en œuvre.
- Ø La méthode résolve une tâche non supervisée, donc elle ne nécessite aucune information sur les données.
- Ø Sa simplicité conceptuelle.
- Ø Sa rapidité et ses faibles exigences en taille mémoire.
- Ø La méthode est applicable à tout type de données (mêmes textuelles), en choisissant une bonne notion de distance.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Les inconvénients :

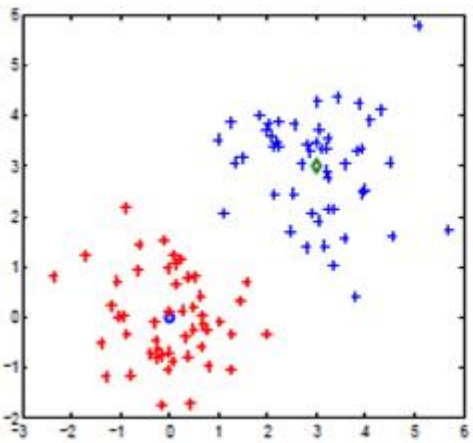
- Ø La partition finale dépend de la partition initiale. Le calcul des centroïdes, après chaque affectation d'un individu, influence le résultat de la partition finale. En effet, ce résultat dépend de l'ordre d'affectation des documents.
- Ø Le nombre de classes est un paramètre de l'algorithme. Un bon choix du nombre k est nécessaire, car un mauvais choix de k produit de mauvais résultats.
- Ø Besoin de spécifier k à l'avance
- Ø Ne gère pas le bruit et les exceptions
- Ø Ne trouve que des clusters de forme convexe.

5.8.3. Classement par modélisation (Modélisation statique)

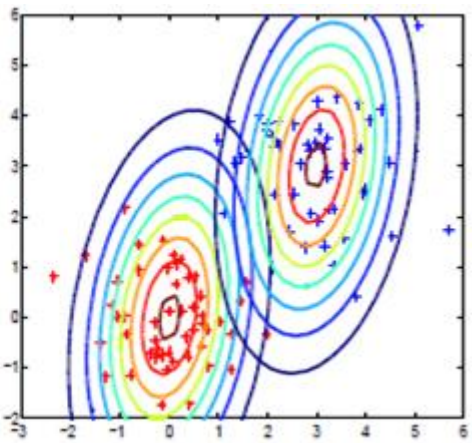
5.8.3.1. Notion de modèles de mélanges

Considérons le modèle de données suivant : N données formant deux classes

Données de deux classes



Données de deux classes et contours des classes



- On veut trouver le modèle statique des données, on constate que pour modéliser les données il faut deux distributions gaussiennes.
- On remarque que dans la classe 1 les données suivent une loi : , et dans la classe 2 les données suivent une loi : .

Loi marginale de X :

d'après Bayes.

Où :

désignent la probabilité à priori.

et désignent la densité conditionnelle de X respectivement à et .

PDF created with pdfFactory Pro trial version www.pdffactory.com

$f(X)$ on l'appelle modèle de mélange de densité, et est déterminé par les connaissances : $\pi_i, \pi_j, \sum i, j \in \{1,2\}$.

Si on connait le modèle de mélange, on connait les probabilités à priori, d'après bayes, on déduit les probabilités à post-priori.

$$\Pr(Z = 1|X = x) = \frac{\pi_1 f(X=x|Z=1)}{f(X=x)} \qquad (3.10)$$

$$\Pr(Z = 2|X = x) = \frac{\pi_2 f(X=x|Z=2)}{f(X=x)} \qquad (3.11)$$

Remarque :

$\Pr(Z = 1|X = x) + \Pr(Z = 2|X = x) = 1$ et donc : le point x est affecté à la classe de plus grande probabilité à post-priori.

$C1$ **ß** x si $\Pr(Z = 1|X = x) > \Pr(Z = 2|X = x)$.

C2 **ß** x sinon.

	K-means	Approche modélisation
Modèle	-	Modèle de mélanges $f(X) = \sum_{k=1}^{K} \pi_k f(X/Z = k)$
Paramètres à estimer	Centres des clusters $\mu_k, k = 1, \cdots, K$	Proba a priori π_k Paramètres des lois $f(X/Z = k)$
Critère optimisé	Variance intra-classe	Log-Vraisemblance
Indice d'affectation	$d(x, \mu_k)$	Proba a posteriori $\Pr(Z = k/X = x)$
Règle d'affectation de x	Cluster dont le centre est le plus proche	Cluster de plus grande proba a posteriori

Tableau 3 : Tableau de comparaison avec K-means.

5.8.3.2. L'Algorithme EM

L'algorithme se fait comme suit :

- Initialiser les paramètres $\pi_1 \mu_1 \sum 1$ et $\pi_2 \mu_2 \sum 2$ $(Rq: \pi_1 + \pi_2) = 1$
- Répéter :

Etape E (Expectation) : calcul des probabilités à post priori.

$$\gamma_i^{(1)} = \Pr\left(Z_i = \frac{1}{x_i}\right) = \frac{\pi_1 N(x_i; \mu_1, \sum_1 \quad)}{\pi_1 N(x_i; \mu_1, \sum_1 \quad) + \pi_2 N(x_i; \mu_2, \sum_2 \quad)} \qquad i = 1 \ldots N$$

Remarque :

$\gamma_i^{(2)} = \Pr\left(Z_i = \frac{0}{x_i}\right) = 1 - \gamma_i^{(1)}$ Probabilité à posteriori que x_i C_2.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Etape M (Maximisation) : calcul des paramètres.

$$\mu_j = \frac{\sum_{i=1}^{N} \gamma_i^{(j)} x_i}{\sum_{i=1}^{N} \gamma_i^{(j)}},$$

$$\sum j = \frac{\sum_{i=1}^{N} \gamma_i^{(j)} (x_i - \mu_j)(x_i - \mu_j)^T}{\sum_{i=1}^{N} \gamma_i^{(j)}},$$

$$\pi_j = \frac{\sum_{i=1}^{N} \gamma_i^{(j)}}{N} \quad j \ \{1,2\}.$$

Ø Jusqu'à convergence.

Après convergence, on dispose des paramètres. On affecte le point x_i au cluster C_j tel que :

$\gamma_i^{(j)}$ Soit maximal.

Exemple :

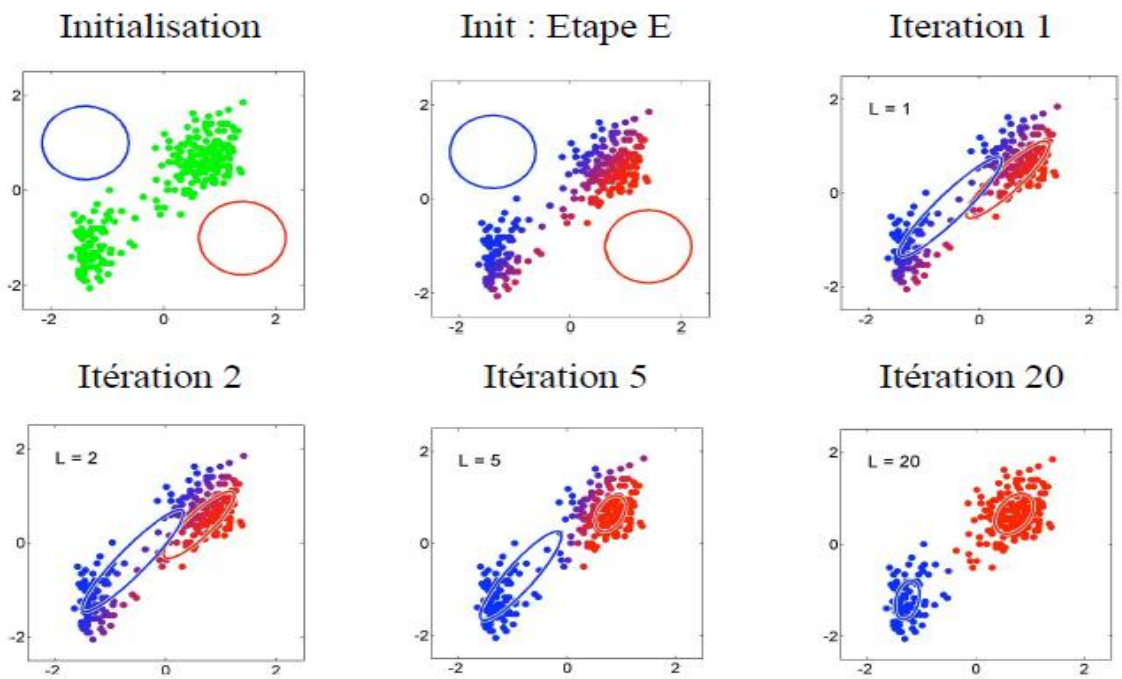


Figure 3.6: Exemple d'illustration de [S.Sankararam].

5.8.3.3. Remarque sur l'Algorithme EM

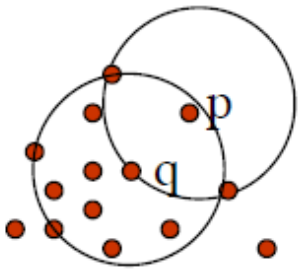
- Convergence vers un extremum local (convergence globale non garantie)
- Critère de convergence possible : variation de la log-vraisemblance
- On montre qu'à chaque étape, on améliore la log-vraisemblance
- Initialisation : aléatoire ou utilisation de résultats a priori (comme par l'exemple faire K-means sur les données et utiliser ces résultats pour initialiser l'algorithme)
- Des initialisations différentes peuvent donner des paramètres différents

PDF created with pdfFactory Pro trial version www.pdffactory.com

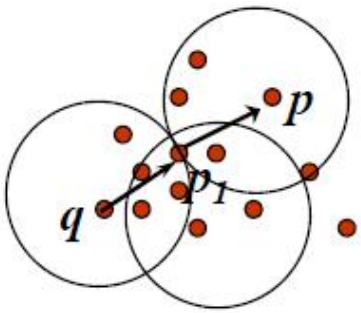
- Il existe des variantes de EM pour le clustering dont le CEM (Classification - Expeectation - Maximisation).

5.8.4. Classement basé sur la densité

- Principales caractéristiques
  - Cluster de forme arbitraire
  - Gestion du bruit
  - Besoin d'un paramètre de densité comme critère d'arrêt
- 2 paramètres
  - Eps : rayon maximal de voisinage
  - MinPts : nombre minimal de points dans le voisinage défini par Eps



-
- un point *p* est directement atteignable d'un point *q* si
  - appartient à
  -

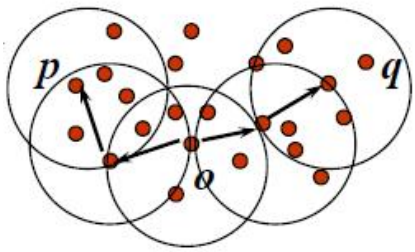


- un point *p* est atteignable d'un point *q* si :

il existe une chaîne de points telle que et et que les sont directement atteignables des

- un point est connecté à un point si *:*

il existe un point *o* tel que et sont atteignables depuis *o*



5.8.5. Classement basé sur une grille

- Utilisation d'une grille des résolutions multiples comme structure de données
- L'espace est divisé en cellules rectangulaires
- Chaque cellule de niveau i est divisée en un certain nombre de cellules plus petites au niveau i+1

PDF created with pdfFactory Pro trial version www.pdffactory.com

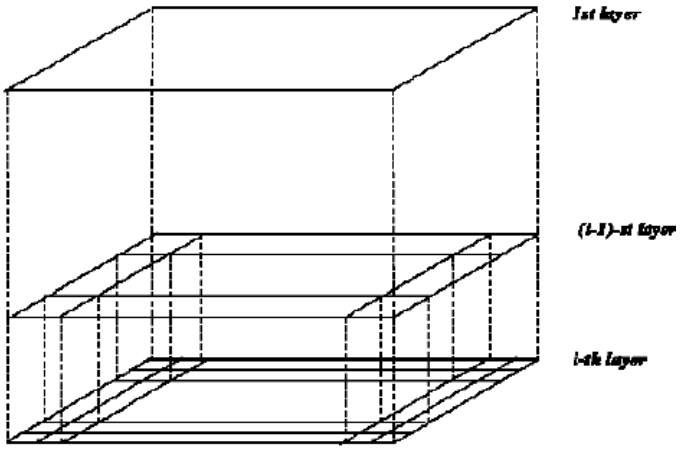


Figure 3.7: Présentation d'une grille

- ü Informations statistiques calculées et stockées à chaque niveau
- ü Approche descendante
- ü Suppression des cellules non pertinentes pour les itérations suivantes
- ü Répéter le processus jusqu'à atteindre le niveau le plus bas.

Avantage :

- parallélisable, mise à jour incrémentale
- , où est le nombre de cellule au plus bas niveau.

Faiblesse :

- Les bords des clusters sont soit horizontaux soit verticaux, pas de diagonale !

6. Conclusion

Nous avons présenté dans ce chapitre les différents algorithmes de Clustering, et nous avons vu aussi le déroulement de chacun d'eux. On peut dire que chacun passe par une initialisation, une boucle et une fin.

Certains algorithmes, contiennent des formules mathématiques (Probabilités) pour calculer la distance entre chaque point (cluster).

Dans le chapitre suivant, nous allons implémenter l'algorithme de k-means.

PDF created with pdfFactory Pro trial version www.pdffactory.com

1. Introduction

Nous avons abordé dans les précédents chapitres les notions de base de la Recherche d'Information ainsi que le modèle de langue. Nous avons aussi présenté dans le troisième chapitre la classification non supervisé (Clustering). Dans ce chapitre, nous allons exposer l'approche proposée ainsi que son implémentation, et évaluation.

Le chapitre est organisé comme suit :

En premier lieu, nous présenterons notre approche, suivie d'un exemple illustrant son fonctionnement.

En second lieu, nous présenterons les données utilisées pour évaluer notre approche, ainsi que le résultat obtenue par cette dernière.

Nous conclurons notre chapitre par une conclusion qui fait la synthèse de notre étude.

2. Description de l'approche implémentée

Généralement la recherche d'information privilégie la minimisation du temps de réponse par rapport à la qualité des documents retournés à l'utilisateur. Ceci vient du faite que ce n'est pas toutes les dimensions d'un document dans le processus d'indexation et de recherche qui sont pris en compte.

2.1 Hypothèse de l'approche

Notre approche consiste à implémenter une méthode de Clustering pour le calcul de la pertinence à priori d'un document dans une collection de documents.
Notre approche est basée sur l'hypothèse suivante : « Un document qui se rapproche de plusieurs thématiques est a priori pas pertinent ». On suppose qu'une thématique est représentée par un cluster.

On peut expliquer cette hypothèse par le fait qu'un document qui traite plusieurs thématiques « tend à être proche à beaucoup de cluster », un tel document peut être qualifié de générique et il ne repend pas à un besoin précis d'un utilisateur.

2.2 Fonction d'appariement utilisée

Pour calculer la pertinence d'un document vis-à-vis d'une requête, on utilise le modèle de lange. La fonction d'appariement de ce modèle est donnée comme suit :

$$P(Q|D) = P(D) \prod_{i=1}^{n} p(t_i|D) \qquad (4.1)$$

Où : *P(D)* est la probabilité à priori de pertinence du document D.

PDF created with pdfFactory Pro trial version www.pdffactory.com

$\prod_{i=1}^{n} \boldsymbol{p}(\boldsymbol{t_i}|\boldsymbol{D})$ Est le score par le contenu.

$$score\ (Di, Q) = \prod_{i=1}^{n}(t_i\ , d_1) \cong \frac{t_i}{|d_i|} \qquad (4.2)$$

Cette formule fait ressortir deux parties la première, P(D) permet d'estimer la probabilité (pertinence) a priori d'un document D.

La seconde $\prod_{i=1}^{n}(\boldsymbol{t_i}\ , \boldsymbol{d_1})$ **,** concerne le score du contenue du document vis-à-vis de la requête.

Afin d'estimer la probabilité a priori P(D), nous nous sommes basés sur l'hypothèse énoncée précédemment, que nous formaliserons de la manière suivante :

$$P(D) = \frac{\sum_k dist(D, C_k)}{\sum_{dj}(\sum_k(d_j, C_k)} \qquad (4.3)$$

Où :

$dist(d_i, C_k)$ Est la distance entre un document et le centroide du cluster C_i.

Afin de calculer cette distance, nous avons utilisé la mesure Euclidienne exprimée ainsi :

$$Dist(d_i, C_n) = \sqrt{\sum_{i=1}^{k}(d_i - C_n)^2} \qquad (4.4)$$

Avant d'effectuer cette tâche, une étape de Clustering de la collection a été réalisée.

Nous avons optée pour l'Algorithme de Clustering K-means.

Nous comparerons notre approche à une autre approche sur basant sur le l'hypothèse suivante : « un document long tend à être générique, dons moins pertinent qu'un document de petite taille ».

La formule matérialisant cette approche est la suivante :

$$P(D) = \frac{|D|}{|C|}$$

|D| est la taille du document D.

|C| est la taille de la collection C.

L'objectif de notre projet est l'utilisation d'une méthode de Clustering pour le calcul de la probabilité à priori de la pertinence d'un document.

La phase la plus importante dans ce projet est l'application de l'algorithme K-Means sur un ensemble de document tout en garantissant un niveau de Clustering intéréssant.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Afin d'arriver au but de ce projet, son implémentation a respectés les phases de la fouille de données et a subit des transformations au fur et à mesure afin d'améliorer le résultat du Clustering.

Voici ci-dessous un exemple illustratif.

Exemple :

Prenons 10 documents et chaque document contient les fréquences des termes.

Si on prend document 1 Doc1 contient :

- terme T1 avec la fréquence de 9
- terme T2 avec la fréquence de 1
- terme T3 avec la fréquence de 7
- terme T4 avec la fréquence de 0
- terme T5 avec la fréquence de 4
- terme T6 avec la fréquence de 2
- terme T7 avec la fréquence de 1
- terme T8 avec la fréquence de 3

Technique :

1er étape : créer les clusters (les thématiques).

Effectuation initiale C1 = {Doc 1, Doc 5, Doc 6}

C2 = {Doc 2, Doc 7}

C3 = {Doc 3, Doc 8, Doc 9}

C4 = {Doc 4, Doc 10}

PDF created with pdfFactory Pro trial version www.pdffactory.com

2ème étape : appliquer l'algorithme de K-means :

Algorithme K-means

Input : E={ e_1,e_2,e_3, ... , e_n} (Set of entités to be clustered)
K (number of clusters)
MaxIters (limit of iterators)

Output: C= {c_1,c_2,c_3, …….. , c_n} (Set of cluster centroids)
L= { *l*(e) | e = 1,2,3, ……. , n} (Set of custer labels of E)
foreach C_i ∈ C do
C_i ß e_j ∈ E (e.g random selection)
end
foreach e_i ∈ E do
l(e_i) ß argminDistance (ei, cj) j∈ {1,2,3, …… , K}
end
changed ß false;
iter ß 0;

repeat
foreach C_i ∈ C do
UpdateCluster (C_i);
end
foreach e_i ∈ E do
minDist ß argminDistance (e_i,e_j) j∈ 1{ …….. k}
if minDist ≠ *l*(e_i) then
l(ei) ß minDist;
changed ß true;
end;
end;
inter ++;
until changed = true and iter ≤ $MaxIters$;

Figure4.1: Algorithme K-means.

Apres avoir appliqué l'algorithme de clustering on obtiendra les résultats suivant :

Cluster 1 : C1 = {Doc1, Doc2, Doc3}

Cluster 2: C2 = {Doc4, Doc5}

Cluster 3: C3 = {Doc6, Doc7, Doc8}

Cluster 4: C4 = {Doc9, Doc10}

PDF created with pdfFactory Pro trial version www.pdffactory.com

		T1	T2	T3	T4	T5	T6	T7	T8	Total
C1	Doc 1	0	1	3	0	0	2	1	1	27
	Doc 2	3	5	1	1	1	1	0	4	25
	Doc 3	3	1	2	3	1	2	7	3	34
Total de fréquence de chaque terme d'un cluster 1 « Tf »		15	7	19	4	14	8	8	11	
C2	Doc 4	2	0	7	3	3	0	2	3	26
	Doc 5	3	2	1	2	1	5	3	0	31
Total de fréquence de chaque terme d'un cluster 2 « Tf »		6	2	12	10	4	13	5	5	
C3	Doc 6	0	2	7	4	2	0	2	6	30
	Doc 7	1	1	0	5	3	3	4	2	32
	Doc 8	5	4	9	0	1	5	0	4	39
Total de fréquence de chaque terme d'un cluster 3 « Tf »		6	22	16	9	14	9	7	18	
C4	Doc 9	6	0	3	10	3	5	8	11	46
	Doc 10	8	9	4	5	6	0	6	3	41
Total de fréquence de chaque terme d'un cluster 4 « Tf »		14	9	7	15	9	5	14	14	

Le score de la similarité qu'on emploie, dans notre cas, est issu de la formule du Cosinus :

$$Sim\ (d_i, C_i) = \frac{\sum_i w_{Cj} \times w_{Ci}}{\sqrt{\sum_i w_{Cj}^2 \times \sum_i w_{Ci}^2}} \qquad (4.5)$$

L'objectif ici est de montrer qu'un document proche des clusters doit avoir un score de pertinence a priori petit(4.3)

$$P(D) = \frac{\sum_k dist(D, C_k)}{\sum_{dj}(\sum_k (d_j, C_k)}$$

PDF created with pdfFactory Pro trial version www.pdffactory.com

Document	Score-Approche	Score-Taille
Doc1	0.00373	0.04145
Doc2	0.00100	0.08290
Doc3	0.00060	0.11398
Doc4	0.00423	0.103626
Doc5	0.00141	0.08808
Doc6	0.00373	0.11917
Doc7	0.00141	0.09844
Doc8	0.00121	0.14507
Doc9	0.00373	0.13989
Doc10	0.00141	0.06735

3. Présentation de l'architecture du système développé

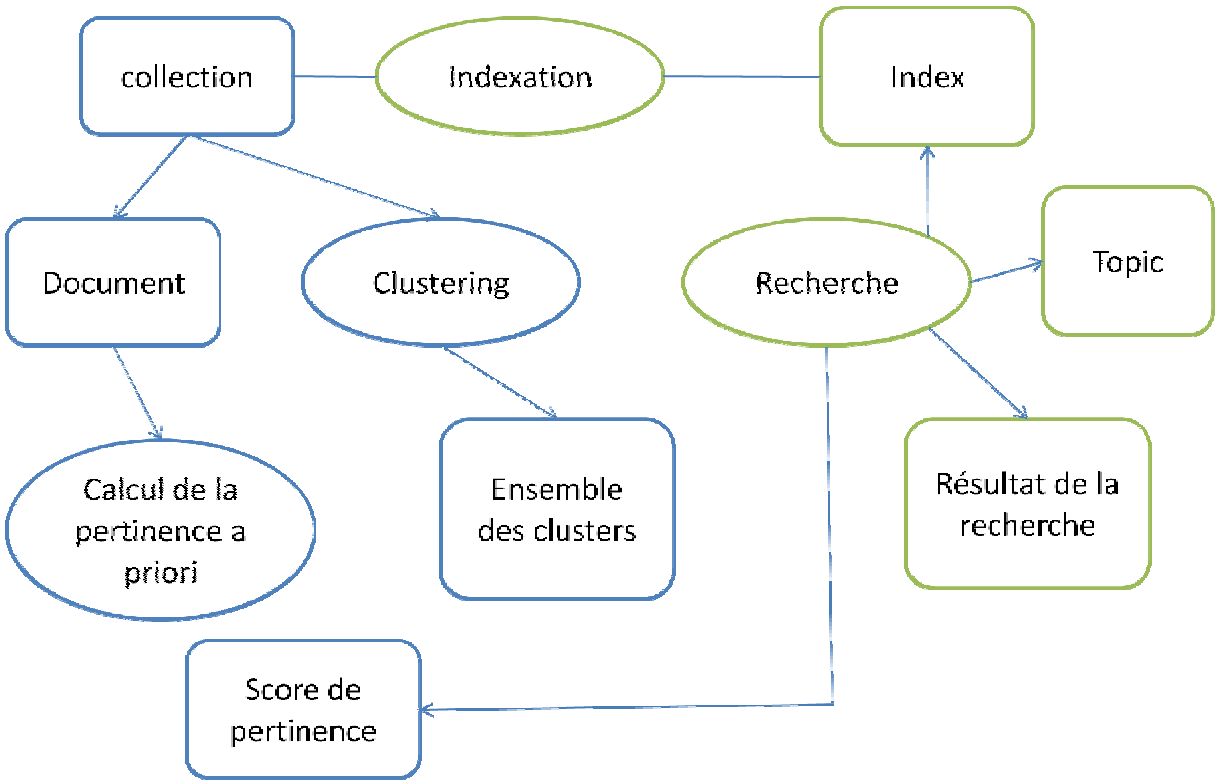


Figure4.2: Architecture générale de notre système.

Test effectué sous Lemur 4.12.

Test effectué sous Terrier 2.1.

PDF created with pdfFactory Pro trial version www.pdffactory.com

3.1. Présentation de la plate forme Terrier 2.1

Terrier (TerabyteRetrieval) est un logiciel libre et open source, classé dans la catégorie des SRI, Sa création fut par le département d'informatique de l'université de Glascow au Royaum-Uni.
Son fonctionnement repose sur différentes méthodes d'indexation, plusieurs modèles de pondération et il permet la reformulation des requêtes.
Deux principales étapes caractérisent terrier :

- L'Indexation
- La recherche
- Evaluation

3.1.1 Processus d'indexation

Le processus d'indexation de terrier consiste à analyser tous les documents de la collection spécifiée afin de produire un ensemble de termes d'indexation « index » et de les ranger dans la structure « index data structure »

3.1.2 Construction d'index

Le but de cette étape est d'effectuer plusieurs traitements afin de parvenir à la création de la structure d'index. La base de ces fichiers est le lexicon qui est composé de terme et de sa fréquence d'occurrence. Le stockage des différentes statistiques se fait sur les quatre principaux types de fichiers :

- **Document index** : contient des informations à propos des documents tels que leurs identificateurs, leurs tailles, etc.
- **Lexicon et Lexicon index** : sauvegarde le vocabulaire.
- **Fichier inverse** : index inverse
- **Fichier directe** : index directe.

PDF created with pdfFactory Pro trial version www.pdffactory.com

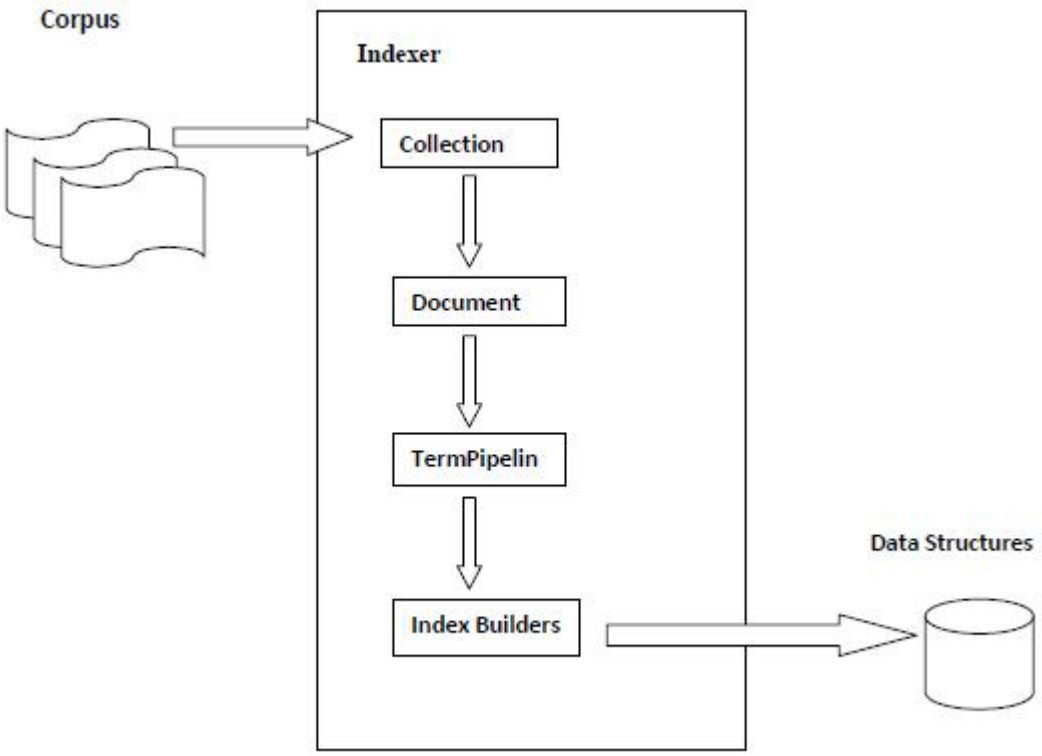


Figure4.3 : Présentation du processus d'indexation de terrier.

3.1.3 Le langage de programmation utilisé (JAVA)

Le choix de JAVA comme langage de programmation s'est imposé vue que l'Open Source de Terrier est entièrement écrit en JAVA. De plus Java est un langage de programmation récent (les dernières versions datent de 1995) développé par Sun MicroSystems récemment racheté par Oracle. Il est fortement inspiré des langages C et C++.

Le langage Java trouve son origine dans les années 1990. A cette époque, James Grosling (Sun MicroSystems) développe un premier langage, Oak, permettant de programmer dans un environnement indépendant du Hardware. En 1992, Bill Joy (cofondateur de Sun MicroSystems) face à une puissante montée d'internet, insiste sur l'élaboration d'un langage indépendant des plates formes et des environnements (et ceci viens des problèmes d'hétérogénéité Matériel et Logiciel utilisé par Internet), des lors, Oak est renommé JAVA en 1995. Une machine virtuelle, un compilateur ainsi que de nombreuse spécification sont données gratuitement et JAVA entame une conquête fulgurante.

Aujourd'hui, après de nombreuses améliorations n'est plus uniquement une solution liée à internet. De plus en plus de sociétés font appelle a ce langage de programmation pour l'ensemble de leurs développements (applications classiques, interface homme-machine, application réseaux, …etc).

PDF created with pdfFactory Pro trial version www.pdffactory.com

3.1.4 Présentation de netbeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun en juin 2000 sous licence CDDL et GPLv2 (Common Development ans Distribution Licence). En plus de java, NetBeans permet également, comme python, C, C++, javaScript, XML, Ruby, PHP et HTML. Il comprend toutes les caractéristiques d'un EDI moderne (éditeur en couleur, projet multi-langage, refactoring, éditeur graphique d'interfaces et de page Web).

Conçu en java, NetBeans est disponible sous Windows, linux, Solaris (sur X86 et SPARC), Mac OS X ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle java JVM). Un environnement Java Development Kit JDK est requis pour les développements en Java. NetBeans constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing(Java)). L'IDE NetBeans s'appuie sur cette plateforme.

Dans notre travail on a utilisé la version NetBeans 6.5, la figure suivante illustre son interface Principale.

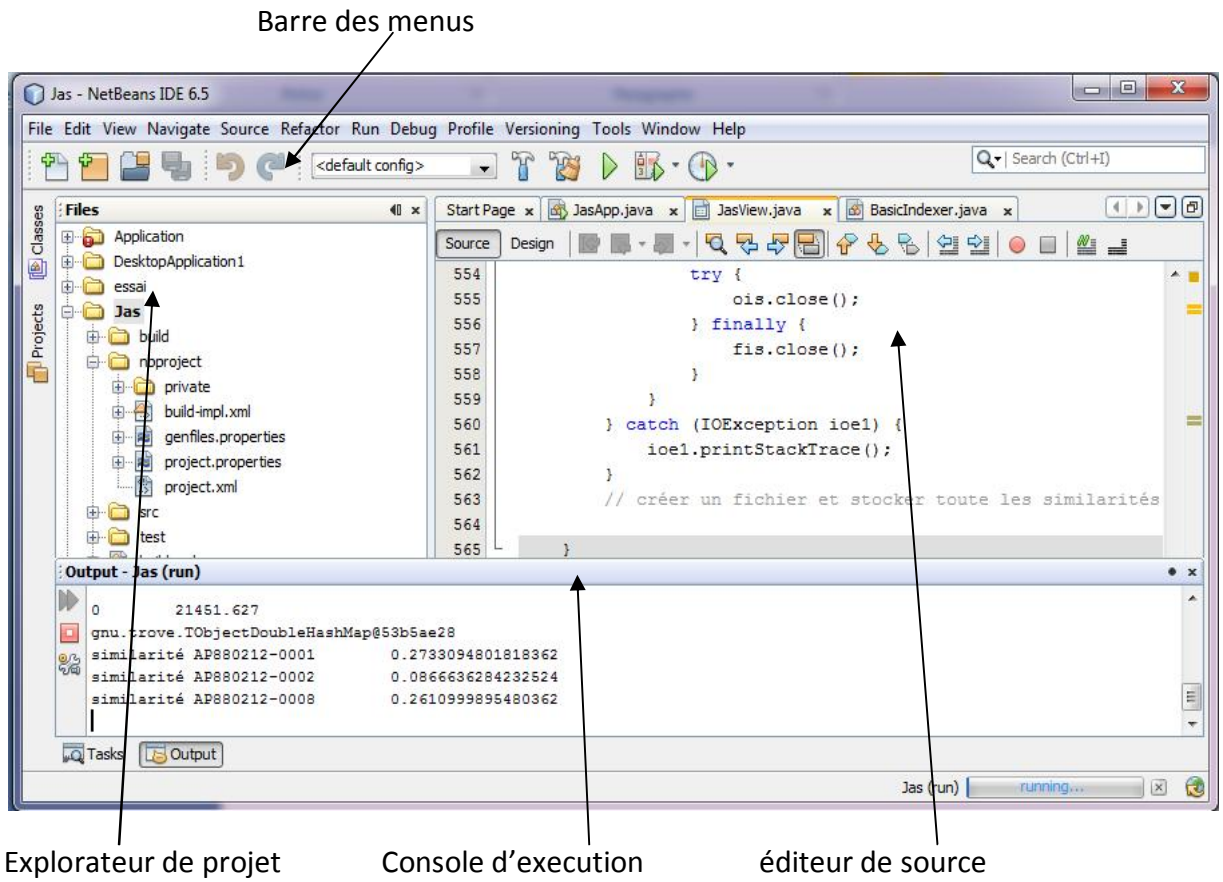


Figure 4.4: Environnement de développement NetBeans.

PDF created with pdfFactory Pro trial version www.pdffactory.com

3.2 Présentation de la plate forme Lemur 4.12

3.2.1 Présentation de la plate forme La recherche d'informations (RI) fournit des techniques et des outils pour trouver les documents contenant l'information pertinente aux besoins des utilisateurs. Et parmi les outils on trouve lemur qui permet de faciliter la recherche dans la modélisation du langage et de la recherche d'information. Il permet de créer de nouveaux systèmes de recherche en créant de nouvelles méthodes et algorithmes.

Lemur est un système de RI "open source". Il contient différentes fonctionnalités pour l'indexation et la recherche.

Le développement de Lemur a été réalisé grâce au projet « The Lemur Project ». Il a été développé en C++ et en C pour une utilisation sous UNIX et Windows XP. Il permet à la fois, d'indexer les données et de faire de la recherche d'information.

- Ø Lemur a été écrit principalement en C + +
- Ø L'interface graphique est écrite en Java
- Ø Il est compatible avec Windows (XP,NT), Linux

Architecture de la plate forme Lemur : Lemur fonctionne avec un système de descripteurs en parallèle, où chaque index reflète une représentation linguistique particulière des documents et requêtes.

Les documents et requêtes passent tout d'abord par un module d'analyse linguistique qui permet d'obtenir 12 représentations différentes d'un même document (ou requête). De manière plus précise, l'architecture proposée peut être synthétisée de la façon suivante:

PDF created with pdfFactory Pro trial version www.pdffactory.com

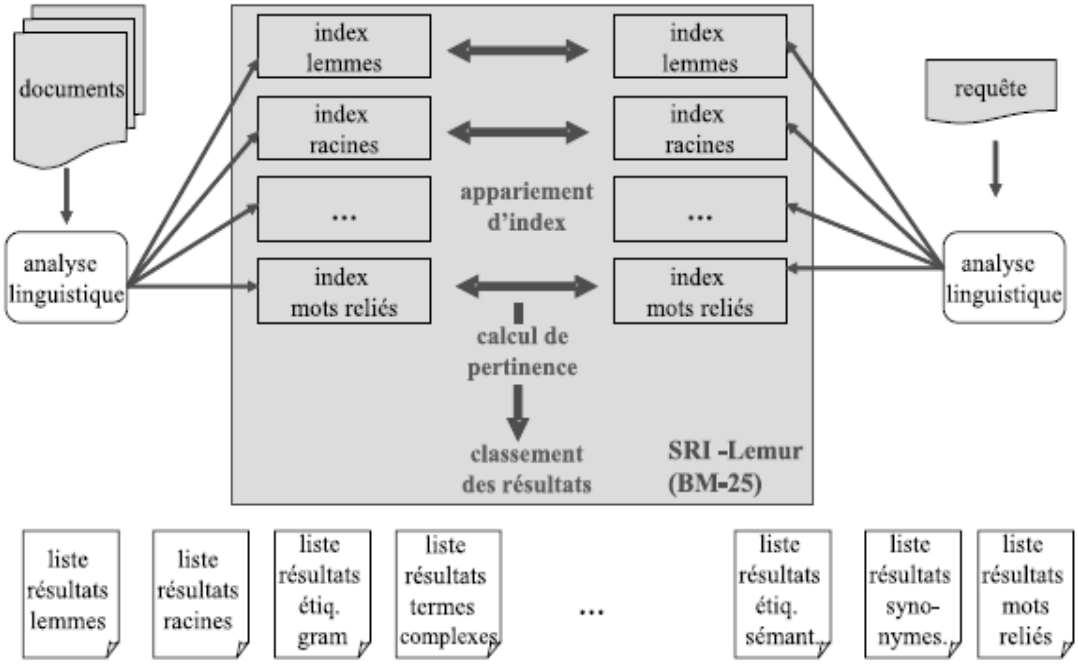


Figure4.5 : Architecture de la plate forme Lemur.

4. Description des collections de tests utilisées

Les collections de tests utilisées dans notre projet sont : la collection AP88 (Associated Press newswire, 1988) et la collection WSJ90-92 (Wall Street Journal, de 1990 à 1992), les statistiques sur ces deux collections sont décrites dans le tableau suivant :

collection	Nombre de documents dans la collection	Nombre de termes dans la collection	Taille moyenne d'un document
AP88	79919	110868	144
WSJ90-92	74520	101428	146

Tableau4 : Description des collections de tests AP88 et WSJ90-92

5. Tests et résultats de notre approche

Nous avons tout d'abord commencer notre travail par l'indexation de document avec la plate forme Lemur et on a obtenue le fichier de la figure 7 (fichier-AP88),par là nous avons créer un fichier parametre.txt qui contient exactement ce code :

PDF created with pdfFactory Pro trial version www.pdffactory.com

<parameters>

<index>E:/indexation-lemur</index> // E:index => c'est le schéma de l'indexation de la collection des documents.

<clusterIndex>mycluster</clusterIndex> //I assume this is the output. So I give an arbitrary name.

<clusterDBType>flatfile</clusterDBType>

<clusterType> centroid </clusterType>

<simType> Cos</simType>

<seuil>0.25</seuil>

<numParts>10</numParts>

<numIters>253</numIters>

<docMode>max</docMode>

<threshold>0.25</threshold>

</parameters>

Une fois le fichier enregistré, on l'a mis dans : C:/Program Files/ Lemur/ Lemur 4.12 /Bin et par la nous avons executer les deux commandes suivante :

Ø ClusterApp parametre.txt

Qui nous retourne le fichier de la figure4.8.

Ø OfflineCluster parametre.txt

Qui nous retourne le fichier de la figure4.9.

5.1 L'indexation avec lemur

L'indexation avec Lemur se fait de la façon suivante :

- Index name : c'est le dossier ou va être stocké l'indexation
- Data file : c'est la collection pour laquel on veut appliquer l'indexation
- Ensuite on clique sur le bouton : Build Index et le dossier E:\indexation-lemur contiendra toute l'indexation des collections sélectionné.

PDF created with pdfFactory Pro trial version www.pdffactory.com

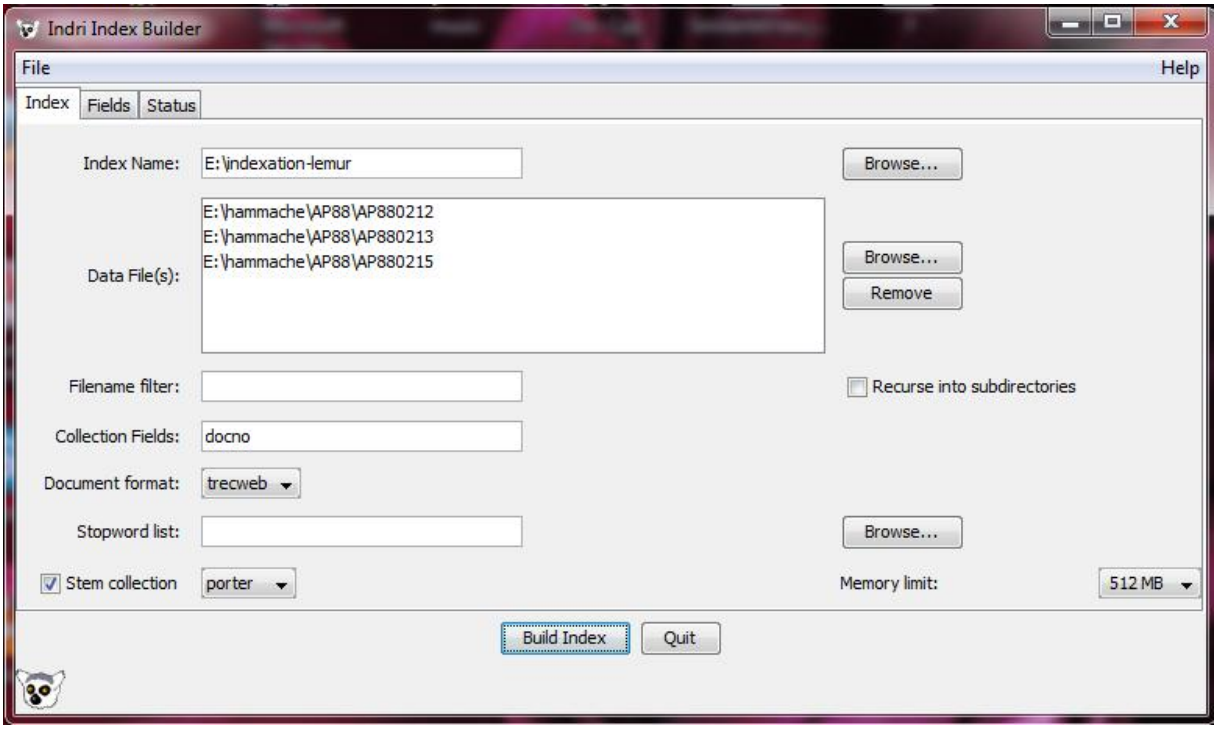


Figure4.6: indexation avec Lemur 4.1

5.2 Fichier-AP88 ce fichier se créera avec la commande suivante sous MS DOS :
C:\Program Files\lemur\lemur 4.12\Bin> ClusterApp parametre.txt> E:\f88.txt

PDF created with pdfFactory Pro trial version www.pdffactory.com

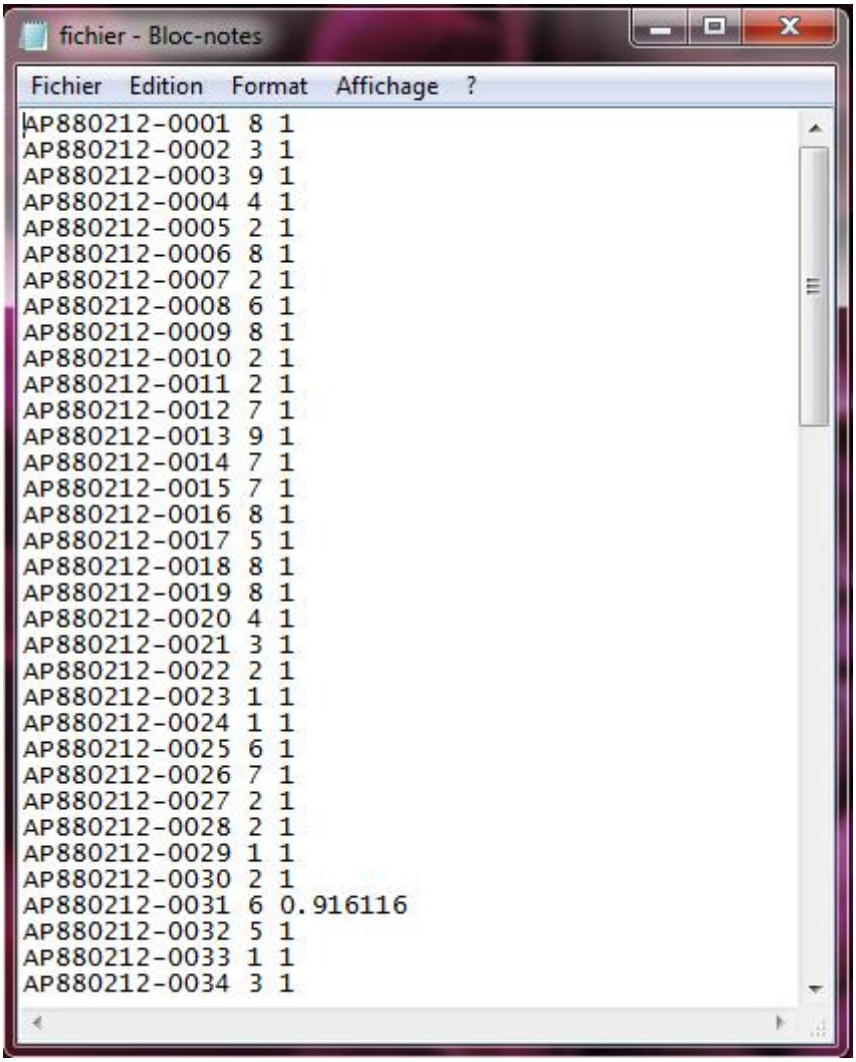


Figure4.7: fichier-AP88 (id-doc / num-cluster / score)

PDF created with pdfFactory Pro trial version www.pdffactory.com

5.3 L'application de Kmeans

Ce fichier se créera avec la commande suivante sous MS DOS :

C:\Program Files\lemur\lemur 4.12\Bin> OfflineCluster parametre.txt> E:\f3.txt

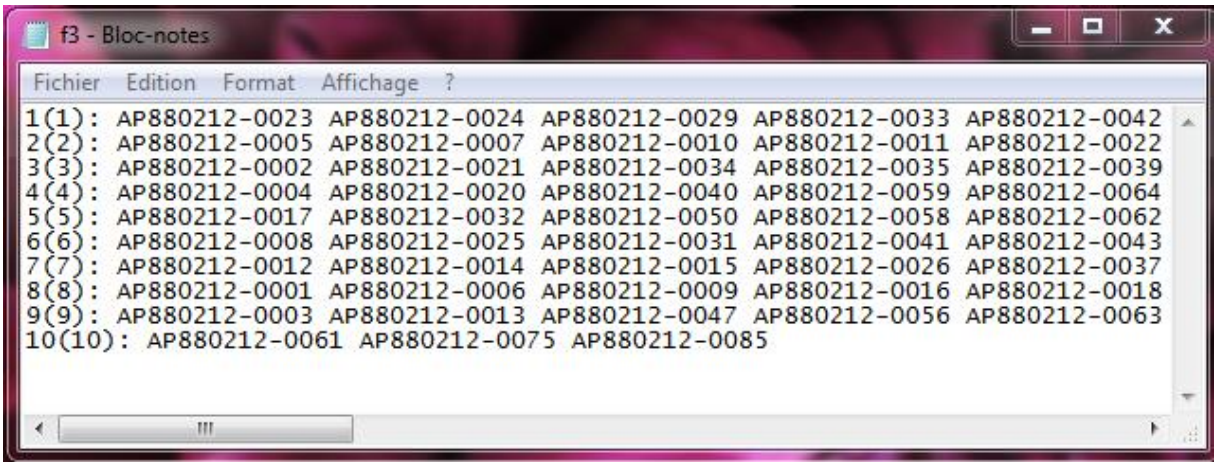


Figure4.8: l'application de l'algorithme Kmeans sur les documents

5.4 L'interface d'acceuil

L'interface d'acceuil possède qutres bouttons chaqu'un sa fonctionnalité :

- Calcul de la similarité entre clusters
- Indexation d'un cluster
- Construction du fichier prior

PDF created with pdfFactory Pro trial version www.pdffactory.com

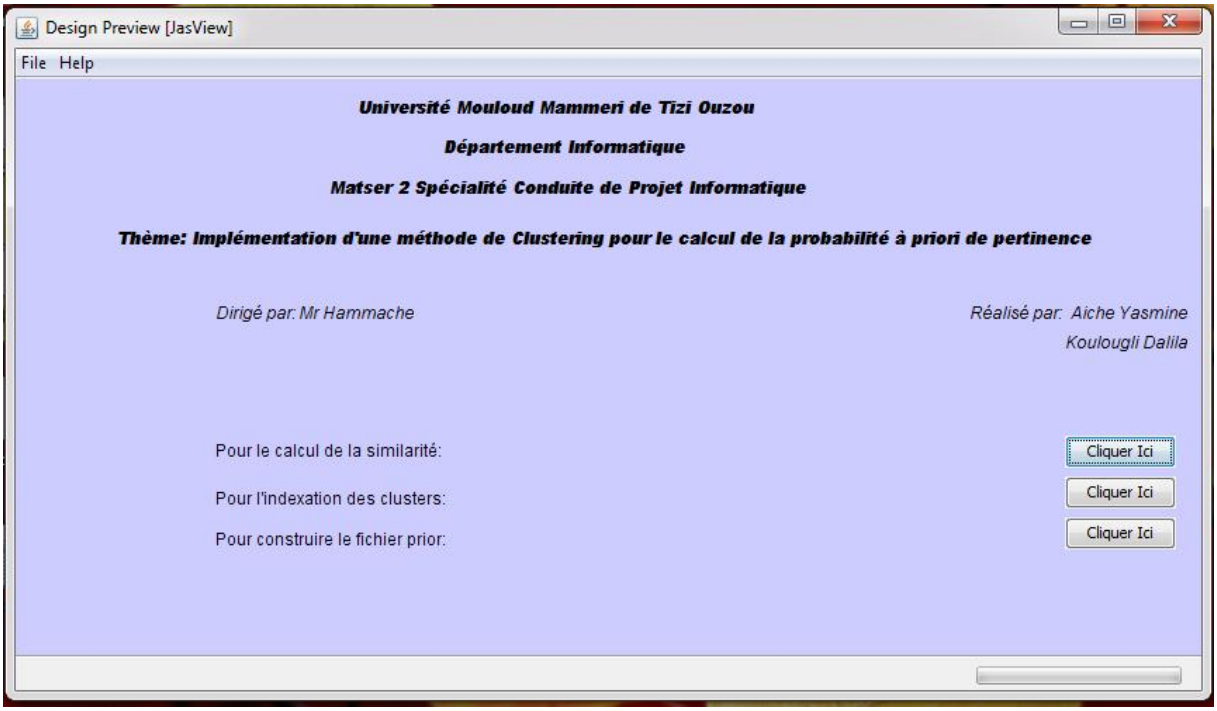


Figure4.9: Page d'Acceuil de notre application.

5.5 Le fichier prior :

Ce fichier est obtenue grace a un programme java, ou on a implémenter la formule suivante : la probabilité a priori de pertinence.

$$P(D) = \frac{\sum_k dist(D, C_k)}{\sum_{dj}(\sum_k (d_j, C_k)}$$

PDF created with pdfFactory Pro trial version www.pdffactory.com

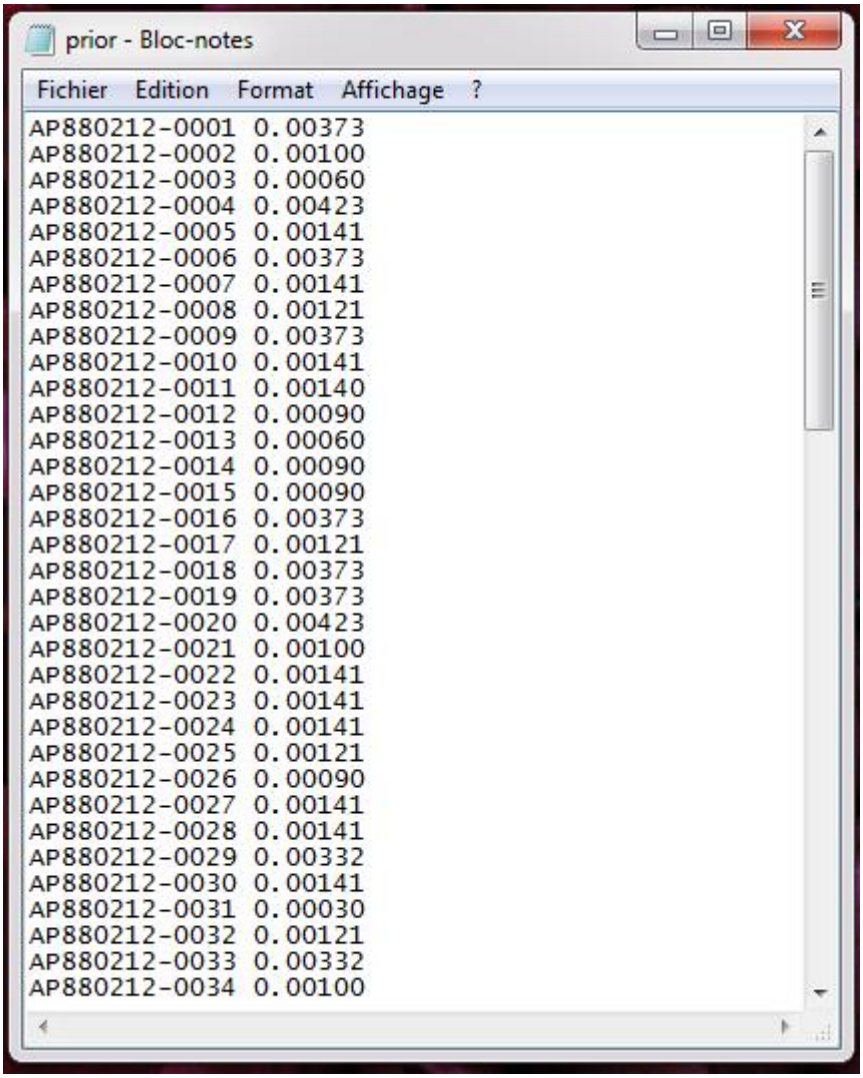


Figure4.10 : Fichier prior.

6. Conclusion

Nous avons vue dans ce chapitre l'approche à utiliser pour l'implémentation de notre méthode.

Nous avons donc optée pour l'algorithme de Clustering K-means, et nous l'avons illustré via un exemple.

Nous avons aussi vue la réalisation de notre approche et son résultat finale.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Conclusion générale et perspectives

1. Conclusion générale

Notre mémoire s'inscrit dans le domaine de la Recherche d'Information, et plus précisément dans l'étude du problème de la probabilité a priori de pertinence d'un document.

Nous avons décris dans ce mémoire une approche qui consiste à un implémenter une méthode de Clustering pour le calcul de la probabilité a priori de pertinence. Cette approche repose sur l'hypothèse qu'un document qui se rapproche de plusieurs thématiques est a priori pas pertinent.

Lors de la réalisation de notre approche, nous avons d'abord indexé la collection de document pour ensuite, appliquer l'algorithme de Clustering K-Means afin de partitionner les données de cette collection, et enfin on a pu calculer la probabilité a priori de pertinente d'un document.

Concernant nos acquis, durant la période de développement de notre travail, nous avons approfondis nos connaissances dans le domaine de recherche d'information classique, nous avons amélioré nos compétences de programmation JAVA dans l'environnement de développement NETBEANS, et aussi nous nous somme familiarisé avec les plateformes Terrier et Lemur qui sont des logiciels libres et open source, les éléments les plus important durant la concrétisation de notre projet.

2. Perspectives

Nous envisageons plusieurs suites à nos travaux

- comparer les performances des autres algorithmes par rapport à k-means qui est une méthode de type *hard Clustering*. Cela signifie qu'un document peut appartenir à un seul cluster et qu'une probabilité unique est calculée pour l'appartenance de chaque document à ce cluster, contrairement à cette approche, l'algorithme d'EM (Expectation Maximization) est une méthode de type soft Clustering. Cela signifie qu'un document appartient toujours à plusieurs clusters et qu'une probabilité est calculée pour chaque combinaison point de données/cluster. Il utile de noter que l'algorithme k-means est très performant en termes de temps d'exécution, mais il souffre du problème de dépendance des résultats aux choix effectués lors de l'initialisation.
- Tester la méthode de Clustering implémenté sur une collection TREC et la comparer à d'autres méthodes telle que : la méthode basé sur la taille du document.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[Adamson et al.,74] G. W. Adamson and J. Boreham. The use of an association measure based on character structure to identify semantically related pairs of words and document titles. Information Storage and Retrieval, 10(7-8):253–260, 1974.

[Amati et al., 99]: G. Amati and F. Crestani. Probabilistic Learning for Selective Dissemination of Information. Information Processing and Management, 35(5):633-654, 1999.

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

[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.

[Belkin et al 1992] N. J. Belkin, W. B. Croft, *Information retrieval and information filtering: Two sides of the same coin?*. CACM, pages: 29-38, 1992.

[Belkin et al., 97] N. J. Belkin, R. N. Oddy, and H. M. Brooks. Ask for information retrieval: part : background and theory. Readings in information retrieval, pages 299–304, 1997.

[Berry & al, 1995] Berry M.W., Dumais T. & O'Brian G., "Using linear algebra for intelligent information retrieval", SIAM Review, Vol 37, 1995

[Berry et al., 99] : Berry, M. W. and Browne, M. 1999 Understanding search engines: mathematical modeling and text retrieval. Society for Industrial and Applied Mathematics.

[Borlund 1998] P. Borlund. Measures of relative relevance and ranked halflife: performance indicators for interactive ir. International ACM-SIGIR conference, pages 24–28, 1998.

[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 et al., 92] : Mohand Boughanem, C. Soulé-Dupuy: A Connexionist Model for Information Retrieval. DEXA 1992: 260-265.

[Boughanem et al., 94] : M. Boughanem, C. Soulé-Dupuy: *Query Modification Based on Relevance Back-Propagation in an Ad hoc Environment*. Inf. Proceeding of RIAO'94, pp. 124-131, 1994.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[Boughanem et al.,99]. Boughanem M., Chrisment C. and Soule-Dupuy C. . Query modification based on relevance backpropagation in adhoc environment. Information Processing and Management, 35 :pages 121-139 .

[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.

[Bordogna et al., 00]: Flexible Querying of Structured Documents. Proceedings of the Fourth International Conference on Flexible Query Answering Systems(FQAS), 2000.

[Boubekeur, 08] Boubekeur F. "Contribution à la définition de modèles de recherche d'information flexibles basés sur les CP-Nets", thèse de doctorat en informatique, Université Toulouse III - Paul Sabatier. 2008

[Celeux et al,.1989].Celeux G,.Diday E., Govaert G., Lechevallier Y., Ralam-Bondrainy H. Classification Automatique des Données. Bordas, Paris, 1989.

[Chapelle, O. et al, 2006].Chapelle, O., Schölkopf, B., & Zien, A. (eds). 2006. Semi-Supervised Learning. Cambridge, MA : MIT Press.

[CLEVERDON 70] C. CLEVERDON. Progress in documentation. evaluation of information retrieval systems. Journal of Documentation, 26:55–67, 1970.

[cooper 70**]:** cooper WS. The potential usefulness of catalog acces points ather than outhor, title, and subject. Journal of the American society for Information Science. 1970:112-127.
[Cooper et al.,88] Cooper WS.Getting beyond Boole. Information Processing and Management. 1988.

[Crestani, 95] : Fabio Crestani: Implementation and Evaluation of a Relevance Feedback Device Based on Neural Networks. IWANN 1995: 597-604

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

[Croft,1991] :W.Bruce Croft :Editorial.ACM Trans.Inf.Syst.9(3) :185(1991).

[Crouch et al., 89] : Crouch, D. B., Crouch, C. J., and Andreas, G. 1989. The use of cluster hierarchies in hypertext information retrieval. In Proceedings of the Second Annual ACM Conference on Hypertext (Pittsburgh, Pennsylvania, United States). HYPERTEXT '89. ACM Press, New York, NY, 225-237.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[Crouch et al., 92] : Carolyn J. Crouch, Bokyung Yang: Experiments in Automatic Statistical Thesaurus Construction. SIGIR 1992: 77-88

[Daniel et al.,2005] Daniel T. Larose. Discovering Knowledge in Data: An Introduction to Data Mining, John Wiley & Sons, Inc., Hoboken, New Jersey. 2005

[Deerwester et al., 1990] Deerwester, S., Dumais, S., Furnas, G., Landauer, T., and Harshman, R. (1990). Indexing by latex semantic analysis. Journal of the American Society for Information Science, 41(6) :391–407.

[Deerwester et al., 90] : Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas and Richard A. Harshman, 1990. "Indexing by Latent Semantic Analysis". In Journal of the American Society of Information Science, Vol. 41:6, 391-407.

[Dumais, 95] : S. Dumais, Latent Semantic Indexing (LSI). Proceeding of TREC-3, 1994.

[Dumais et al., 1996] Dumais S., Landauer t. & Littman M., "Automatic cross-linguistic information retrieval using latent semantic indexing", In SIGIR'96 – Workshop on Cross-Linguistic Information Retrieval, pages 16–23, 1996.

[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.

[Fluhr et al 1985] C. Flhur, F.Debili *Interrogation en langue Naturelle de données textuelles et factuelles* Intelligent Multimadia Information System and Management (RIAO), Grenoble (France), pages : 548-556, 1985.

[Foltz, 90] : P. W. Foltz, *Using Latent Semantic Indexing for information filtering.* CACM, pp. 40-47, 1990.

[Foltz & Dumais, 1992] Foltz P. & Dumais S., "Personalized information delivery: An analysis of information filtering methods", Communications of the ACM, 35(12): 51–60, 1992.

[Fox 92] C. Fox. Lexical analysis and stoplists, pages 102–130. Frakes WB, Baeza- Yates R (eds) Prentice Hall, New jersey, 1992.

[Frakes et al., 92] : William B. Frakes and Ricardo Baeza-Yates (eds.), 1992. Informa-tion Retrieval Data Structures & Algorithms. Prentice-Hall. ISBN

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[Frakes et al.,92**]:** Frakes WB, Baeza-Yates R. Information Retrieval: Data Structures and Algorithms. Englewood Cliffs, NJ: Prentice Hall; 1992.

[Fuhr et al, 91] : N. Fuhr and C. Buckley : A Probabilistic Learning Approach for Document Indexing. In ACM Transactions on Information Systems, (9) (3). Pp 223-248, 1991.

[Furnas et al, 88] : Furnas, G.W., Landauer, T.K., Gomez, L.M., and S.T. Dumais.: The Vocabulary Problem in Human-System Communication, Communications of the ACM 30 (1987) 964-971.

[Grossman et al.,98] Grossman DA, Friender O. Information Retrieval: Algorithms and Heuristics. Norwell, MA: Kluwer Academic publishers; 1998.

[Grossman et al., 98] : David Grossman and Ophir Frieder, Ad Hoc Information Retrieval: Algorithms and Heuristics, Kluwer Academic Publishers, 1998.

[G.Salton, 1970] the SMART retrieval system: Experiments in automatic document processing. Prentice Hall 1970.

[Harter 92] S. Harter. Psychological relevance and information science. Journal of the American Society for Information Science (JASIS), 43:602–615, 1992.

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

[Hlaoua 07]. Hlaoua L. Reformulation de Requêtes par Réinjection de Pertinence dans les Documents Semi-Structurées, thèse en informatique , l'Université Toulouse III -Paul Sabatier

[Hiesmtra, 1998]:Djored Hiesmtra, A linguistically motive probabilistic model of information retrieval ,dans Christos N and Stephanides C. (eds), Proc.European Conference of Digital Library (ECDL98), Sept.1998,Springer Verlag.

[H. TEBRI, 2004] : « formalisation et spécification d'un système de filtrage incrémental d'information », 2004.

[Jacquemin et al., 02] : Jacquemin, C., Daille, B., Royanté, J., and Polanco, X. 2002. In vitro evaluation of a program for machine-aided indexing. *Inf. Process. Manage.* 38, 6 (Nov. 2002), 765-792.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[K. Amrouche, 2008] : Karima AMROUCHE, thèse de Doctorat en Informatique. Option : Système d'information. Thème : Passage à l'échelle en Recherche d'Information Méthode d'élagage pour la réduction de l'espace de recherche. Institut National de la formation en Informatique (I.N.I) Oued smar Alger (2007 /2008).

[Kraaij, Westerveld & Hiemstra, 2002]: The importance of prior probabilities for entry page search. ACM SIGIR Conference on Research and Bibliography and Development in Information Retrieval. Tampere, Finland, (2002) 27-34.

[Koczy et al., 98] : Baranyi, P.; Gedeon, T.D.; Koczy, L.T.; Intelligent information retrieval using fuzzy approach. Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on Volume 2, 11-14 Oct. 1998 Page(s):1984 - 1989 vol.2 Digital Object Identifier 10.1109/ICSMC.1998.728188.

[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.

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

[Lagus et al., 96] : Krista Lagus, Timo Honkela, Samuel Kaski, and Teuvo Kohonen : WEBSOM - A Status Report. In Proceedings of STeP'96. Jarmo Alander, Timo Honkela and Matti Jakobsson (eds.). Publications of the Finnish Artificial Intelligence Society, pp. 73-78.

[Lafferty & Zhai, 2001]: Lafferty, J. and Zhai, C(2001). Document langague models, query models,and risk minimization for information retrieval. In Proceedings of the 2001 ACM SIGIR Conference on Research and Development in information Retrieval, pages 111-119.

[Lelu et al 1992] A. Lelu, C. François, *Information retrieval based on a neural unsupervised extraction of thematic fuzzy clusters*. Les réseaux Neuromimétiques et leurs applications (Neuro Nîmes), pages : 93-104, 1992.

[Loiseau et al, 2004] : Mohand Boughanem, Yannick Loiseau, Henri Prade. Rank-ordering documents according to their relevance in information retrieval using refinements of ordered-weighted aggregations. Dans : AMR05, 3rd International Workshop on Adaptive Multimedia Retrieval, Glasgow, UK, 28 juillet 29 juillet 2004. Springer.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[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.

[MdG 91] J. Maniez and E. de Grolier. A decade of research in classification. 1991.
[Miller, 1998]: David R.H Miller, Tim Leek and Richard M.Schwartz, BBN at TREC-7 using Hidden Markov Models for Information Retrieval, TREC7, pp.133-142, 1998.

[Miller, Leek & Schwartz, 1999]: David R. H. Miller and Tim Leek and Richard M. Schawartz, A Hidden Markov Model Information Retrieval System, Research and Development in Information Retrieval Proc. ACM-SIGIR, pp.214-221,1999.

[Mizzaro 97] S. Mizzaro. Relevance, the whole (hi) story. Journal of the American Society for Information Science, 48:810–832, 1997.

[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.

[Nie et al., 99] : Fuji Ren, Lixin Fan, Jian-Yun Nie, SAAK Approach: How to Acquire Knowledge in an Actual Application System, IASTED International Conference on Artificial Intelligence and Soft Computing, Honolulu, 1999, pp.136-140.

[Page et .al,1998]: Lawrence Page, Sergey Brin, Rajeev Motwani, et Terry Winograd. The pagerank citation ranking : Bringing ordb.er to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

Pedrycz, W. (2007). Collaborative and knowledge-based fuzzy clustering. International Journal of Innovative, Computing, Information and Control 1(3), 1–12.

[Ponte & al, 98]: Jay M. Ponte and W.bruce Croft."A Language Modeling Approach to Information Retrieval". Research and Development in Information Retrieval, Proc.ACM SIGIR, pp.275-281,1998.

[Porter 80] M. F. Porter. An algorithm for suffix stripping. Program, 14(3):130–137, 1980.
[Qui et al., 93] : Y. Qui and H. P. Feri, "Concept Based Query Expansion," in Proc. of the Sixteenth Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pp. 160-169, 1993.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[Ricardo et al.,99]. Ricardo B Y., Berthier R N. Modern information retrieval, ACM (Association for Computing Machinery) .

[Rijsbergen, 79] V. Rijsbergen. “Information Retrieval”. Butterworths & Co, Ltd, London, 1979.

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

[Robertson et al 1976] S. Robertson, EA. Sparck Jones, *Relevance weighting of search terms*, Journal of American Society of Information Science, JASIS, 27(3), pages : 129-146, 1976

[Robertson, 77] : Robertson, S. E. (1977). The probability ranking principle in IR. Journal of Documentation, 33 (4), 294-304.

[Rorbertson et al., 1997]. Robertson S. E., Walker S. 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.

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

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

[Salton, 83] : Salton, G., & McGill, M.. Introduction to Modern Information Retrieval. McGraw-Hill, New York, 1983.

[Salton et al., 86] G. Salton and M. J. McGill. Introduction to Modern Information Retrieval. McGraw-Hill, Inc., New York, NY, USA, 1986.

[Salton & Buckley, 1988] G.Salton & C.Buckley : *Term Weighting Approaches in Automatic Text Retrival,* Information Processing and Management, pp 513-523, 1988

[Salton 1989] Salton G. *Automatic text processing: the transformation, analysis, and retrieval of information by computer*. Boston, MA: Addison-Wesley Longman Publishing Co., Inc.; pages : 85-92, 1989.

[Saracevic 75] T. Saracevic. Relevance : A review of and a framework for the thinking on the notion in information science. Journal of the American Society for Information Science (JASIS), 2:321–343, 1975.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[Saracevic 96] T. Saracevic. Relevance reconsidered. International Conference on Conceptions of Library and Information Science (COLIS), 39(3):201–218, 1996.

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

[S. FELLAG 2006] : Mémoire de magister « recherche d’information dans les documents semi-structurés XML », 2006.

[Signore et al 1992] O. Signore, A.M.Garibald, M.Greco *Proteus: a concept browsing interface towards conventional Information Retrieval System* Database and Expert System Applications (DEXA), pages : 149-154, 1992.

[Singhal et al.,96] Singhal A, Buckley C, Mitra M, Pivoted document length normalization. SIGIR. 1996: 21 29.

[Singhal et al., 1997]. Singhal A., Mitra M., Buckely C. Learning routing queries in a query zone. In Proceedings of the 20th Annual international ACM SIGIR Conference on Research and Development in information Retrieval (Philadelphia, Pennsylvania, United States, July 27 - 31, 1997).

[Soule Dupuy, 90] : Chantal Soule-Dupuy. Systèmes de recherche d'information : le système Videotex Infodiab. Mecanismes d'indexation et d'interrogation. Thèse de Doctorat, Université Paul Sabatier, Toulouse, France.

[SparckJones, 79] : Karen Sparck Jones: Experiments in relevance weighting of search terms. Inf. Process. Manage. 15(3): 133-144, 1979.

[Steinbach, 2000] Steinbach, Karypis, Kumar: *A comparison of document clustering techniques*, ACM SIGKDD, 6th World Text Mining Conference, 2000

[Tamine 00] : Tamine L. Optimisation de requêtes dans un système de recherche d’information. Thèse de doctorat. Toulouse : Université Paul Sabatier ; 2000.

[Turtle et al.,91] Turtle H, Croft WB, Evaluation of an inference network-based retrieval. ACM transactions on Information Systems. 1991: 187-222.

[Walker 97] S. Waller, S. E. Robertson, M. Boughanem, G. J. F. Jones, K. Sparck Jones. *Okapi at TREC-6 automatic and ad hoc*, VLC routing, filtring and QSDR. Proceeding of TREC-6, 1997.

PDF created with pdfFactory Pro trial version www.pdffactory.com

Bibliographie

[Wong et al.,85]: Wong SKM,Ziarko W, Wong PCN, Generalized vector spaces model in Information retrieval. In: Proceedongs of the 8th annual international ACM SIGIR conference on Research and development in information retrieval. New York, NY: ACM; 1985. P. 18-25

[Wong et al.,87]: Wong SKM, Ziarko W, Raghavan VV, Wong PCN, On modeling of information retrieval concepts in vector spaces. ACM Transactions on Information Systems. 1987:229:321

[Yates et al 1999] R. B. Yates, R. Neto, *Modern Information Retrieval*. ACM Press, Addison Wesley, pages: 70-77, 1999.

[Zadeh, 1965] L.A. Zadeh, "Fuzzy sets", Information and control, p338-353, 1965.

[Zhu & Gauche, 2000]: Incorporating quality metrics in centralized / distributed information retrieval on the World Wide Web. The 23th Annual Internatonal ACM SIGR Conference on Research and Development in informayion Retrieval, Athens, Greece, (2000) 288-295.

PDF created with pdfFactory Pro trial version www.pdffactory.com