Fusion de systèmes et analyse des caractéristiques linguistiques des requêtes: vers un processus de RI adaptatif

Nongdo Désiré Y. Kompaoré

THÈSE

présentée

devant l'Université Paul Sabatier de Toulouse

pour obtenir

le grade de : *Docteur de l'Université Paul Sabatier*
Mention INFORMATIQUE

par

Nongdo Désiré Yawbsom KOMPAORÉ

Équipe d'accueil : SIG/EVI
École Doctorale : MITT
Composante universitaire : IRIT

Titre de la thèse :

Fusion de systèmes et analyse des caractéristiques linguistiques des requêtes :
vers un processus de RI adaptatif

présentée le 26 Juin 2008 devant la commission d'examen

Florence	SÈDES	Professeur Université Paul Sabatier Toulouse	Président
Josiane	MOTHE	Professeur IUFM Toulouse	Directeur
Béatrice	DAILLE	Professeur Université de Nantes	Rapporteurs
Marc	EL-BÈZE	Professeur Université Avignon	
Iadh	OUNIS	Associate Professor University of Glasgow	
Alain	BACCINI	Professeur Université Paul Sabatier Toulouse (Invité)	Examinateurs
Claude	CHRISMENT	Professeur Université Paul Sabatier Toulouse	
Ludovic	TANGUY	Maître de conférence Université Mirail Toulouse (Invité)	

Remerciements

Je me souviens il y'a plus de 4 ans déjà mon arrivée à Toulouse pour le DEA 2IL. Un an plus tard, je me souviens aussi de mes débuts en thèse sous la direction de Josiane Mothe. Tout au long de ce parcours, les personnes qui m'ont côtoyé, rencontré, supporté et soutenu ont participé de près ou de loin à l'aboutissement de ces années d'études. Je tiens à les remercier et à leur signifier toute ma gratitude.

Merci à Monsieur Claude Chrisment de m'avoir donné des conseils avisés lors de ces moments difficiles de la thèse, et de m'avoir fait l'honneur de participer au jury de ma thèse. Je tiens à vous exprimer mes sincères remerciements.

Je tiens à exprimer ma reconnaissance à Madame Josiane Mothe qui a dirigé mes recherches depuis mon DEA. Merci de votre disponibilité et de votre rigueur dans le travail. Merci de m'avoir fait découvrir le monde de la recherche dans toute sa complexité... Soyez assurée de ma profonde gratitude.

Je remercie Madame Béatrice Daille, Monsieur Marc El-Bèze et Monsieur Iadh Ounis pour leur précieux conseils et leurs remarques qui ont permis d'améliorer la qualité de ce mémoire, ainsi que leur participation au jury de ma thèse.

Merci à Madame Florence Sèdes de m'avoir fait l'honneur de participer au jury. J'ai vraiment apprécié votre bonne humeur et votre disponibilité. Je remercie également Ludovic Tanguy d'avoir consacré un peu de son temps à la relecture de ce mémoire. Merci à Alain Baccini d'avoir accepté recevoir un pauvre étudiant à la recherche de *méthodes statistiques* pour avancer dans ses travaux. Je suis très fier de notre collaboration et je tiens à vous exprimer toute ma reconnaissance.

Merci à vous tous de m'avoir fait l'honneur de participer à mon jury de thèse et d'avoir porté un intérêt à mes travaux.

Merci à Sébastien de m'avoir permis d'avoir un peu d'R (on se comprend...). Merci pour ta disponibilité et toutes tes réponses.

Merci à Daniel Marre pour notre collaboration à L'INSA de Toulouse. J'ai vraiment apprécié travailler avec vous pendant ces années.

Merci à Nesrine, Karen, Estella, Eloise pour tous ces moments passé ensemble lors des repas et toutes ces discussions. Je suis très heureux de vous connaître. Pour mes "colocataires de bureau" (Bouchra, Dana, Estella, Bachelin), je tiens à vous dire merci pour l'ambiance trèèèèès studieuse qui régnait dans le bureau ... Merci de ces moments de fou rire et merci de m'avoir supporté pendant ces quelques années. A tous les autres membres de l'équipe SIG (Max, Gilles, Olivier, Ronan, Guillaume, Moultazem, Chantal, Bernard, Boughanem...) je tiens à vous dire merci pour tout.

Je remercie également le personnel de l'Irit (Agathe, Jean-Pierre, Jean-Claude, Jean-Philippe, Françoise, et tous les autres). Merci pour votre sympathie et votre jovialité.

Comme on le dit souvent, *chaque pièce a son revers*. De *l'autre côté de la thèse*, il y a tous ces moments ludiques qui m'ont permis de m'épanouir à Toulouse. Je tiens à remercier mes "maîtres" de guitare (Timothee, Remy, Bachelin) d'avoir eu la patience de me faire découvrir cet instrument de musique qui fait partie intégrante de mon nouvel environnement musical. Merci à mes compagnons du basket (Sylvain, Mattieu, Greg, Jérémie, Cyril, ...) d'avoir essayé d'être à mon niveau. Je reconnais avoir été un peu fort pour vous mais nous avons passé (et nous continuerons) de bons moments ensemble. Je remercie pour leur amitié tous mes amis du FJT avec lesquels j'ai gardé de très bons contacts (Aurelie, Sophie, Jean-Michel, ...).

Ahinama... Salsa. Merci à Eric, Mike, Isaac de m'avoir fait découvrir le monde de la musique. J'y ai rencontré des personnes que j'apprécie beaucoup. Merci à vous d'avoir partagé cette passion avec moi (Arnaud, Marc, Pierre, Sylvia, Iza, et tous les autres).

Que ferais-je sans mes amis de tous les jours, qui sont toujours présents à mes côtés depuis notre plus jeune enfance ? Joel, Zouzbe, Ricki, Puff. Merci de votre soutien sans faille et de votre amitié.

Il y a aussi des personnes que l'on rencontre en cours de route et qui nous marquent profondément. Merci à Maikomi, Yoyo, Maeva, Julie, Chantal, Guy et Barbara. Vous êtes comme ma deuxième famille et je suis très heureux d'avoir établi ces liens avec vous.

Merci à Séra, Valé, Maxime, Céline, Kouti, Sylvie et, Jean-Jacques. Merci pour tout. Merci de votre présence, votre disponibilité et votre amitié, les soirées passées, à venir, la salsa, etc .

Judith, je tiens à te dire MERCI du fond du coeur. Merci pour tout.

Je ne peux m'empêcher d'avoir une grande émotion lorsque je pense aux personnes qui me sont les plus chères : mes PARENTS. Papa, maman ; merci d'avoir toujours été là pour moi et de m'avoir donné envie d'étudier. Merci pour toutes ces valeurs que vous nous avez inculquées. Vous avez toujours été des modèles pour moi et je tiens à vous dédier le fruit de toutes ces années d'études. Merci de votre amour et de la confiance que vous avez en moi. Je pense aussi à vous, Nènè, Olivier, Aimé, Claudine. Bark Bark Bark...

Comme le dit l'adage, "après un temps il en vient un autre". La suite au prochain épisode...

Résumé

La recherche d'information (RI) est un domaine de recherche qui est de plus en plus visible, surtout avec la profusion de données (textes, images, vidéos, etc) sur Internet. Nous nous intéressons dans cette thèse à la RI à partir de documents textuels non structurés.

Trois éléments sont essentiels dans un processus de RI : un besoin d'information (généralement exprimé sous la forme d'une requête), un système de recherche d'information (SRI), et une collection de documents. Ainsi, la requête est soumise au SRI qui recherche dans la collection les documents les plus pertinents pour la requête. La variabilité relative à l'expression de la requête, la relation entre la requête et les documents, ainsi que celle liée aux caractéristiques des SRI utilisés conduisent à des variabilités dans les réponses obtenues (Buckley et al., 2004). Ainsi, le système A peut être très performant pour une requête donnée et être très médiocre pour une autre requête, alors que le système B conduira à des résultats inversés.

Notre thèse se situe dans ce contexte. Notre objectif est de proposer des méthodes de recherche pouvant s'intégrer dans un modèle de recherche capable de s'adapter à différents contextes. Nous considérons par exemple que les caractéristiques linguistiques (CL) des requêtes, les performances locales des systèmes ainsi que leurs caractéristiques sont des éléments définissant différents contextes. Nous proposons plusieurs processus afin d'atteindre cet objectif. D'une part, nous utilisons un profil linguistique des requêtes (Mothe et Tanguy, 2005) qui nous permet d'établir une classification des requêtes à base de leurs CL. Nous utilisons à cet effet des techniques statistiques d'analyse de données telles que la classification ascendante hiérarchique (CAH) et les k-means. Les requêtes ne sont plus alors considérées de manière isolée, mais sont vues comme des groupes possédant des CL similaires. L'hypothèse sous-jacente que nous faisons est qu'il existe des contextes dans lesquels certains SRI sont plus adaptés que d'autres. Nous étudions alors les performances des systèmes sur les classes de requêtes obtenues (contextes). Nous proposons quatre méthodes de fusion afin de combiner les résultats obtenus pour une requête donnée, par différents SRI. Une série d'expérimentations valide nos propositions.

L'ensemble de ces travaux s'appuie sur l'évaluation au travers des campagnes d'évaluation de TREC.

Mots-clés : Recherche d'information, fusion de données, classification de requêtes, caractérisation linguistique, analyse canonique

Abstract

Today, accessing wide volumes of information is reality. Information retrieval (IR) techniques are more and more used by a huge number of users on the Internet to retrieve relevant information (data, video, pictures, etc.). We are interested in this work in textual IR.

Three elements are necessary during an IR process : an information need (more often a query of few words), an IR system and a set of documents. The query is submitted to the system which tries to return relevant documents from the set of document as an answer to the user inquiry. Variability in the expression of the query lead to variation in the performances of the systems (Buckley et al., 2004). For instance, system A can be very efficient for a given query and very bad for an other one, whereas system B gets opposite results.

Or thesis is done in this context of variabilities. The main objective of our work is to propose retrieval techniques that can adapt to different contexts. We consider for example that the linguistic features of queries, the performance of the systems and their characteristics are contextual elements of the retrieval process. Many propositions are done in this thesis. Queries are clustered according to their linguistic features (Mothe et Tanguy, 2005) with technics like Agglomerative clustering methods and k-means. Queries are then analysed by the *linguistic profile* of their belonging cluster. The underlying hypothesis is that some IR systems are more suitable than other for different clusters of queries. We analyse the performance of the systems for each of the determined cluster of queries (query context). Four fusion methods are proposed and tested with a set of experiments.

This work is done in the context of TREC campain.

Key-words : Information Retrieval, data fusion, query clustering, linguistique features, canonical analysis

Table des matières

Table des figures

Liste des tableaux

Introduction

L'information permet de répondre à un besoin ponctuel ou continu et représente un certain nombre de connaissances que nous souhaitons acquérir, transmettre ou partager. Cette notion d'information est définie de plusieurs manières différentes et concerne plusieurs aspects de notre vie. Par exemple, d'après le dictionnaire TLFi (Trésor de la Langue Française Informatisé)[1], l'information correspond à "l'action de s'informer, de recueillir des renseignements sur quelqu'un, sur quelque chose". D'après la théorie de l'information de Claude Shannon [Sha48], la conception la plus répandue de l'information est liée au couple <message + récepteur>, le dernier possédant des connaissances implicites valorisant le message et lui permettant de l'interpréter. L'information est une donnée que l'on recherche, manipule et même que l'on protège. Plusieurs supports véhiculent de l'information. L'information peut être orale ou visuelle, et même multimédia. La présence de tous ces médias facilite l'accès et le traitement de l'information. Toutefois, l'information n'a de sens que si elle peut être exploitée par la personne qui en a besoin.

Nos travaux de recherche se situent dans le cadre de la recherche d'information (RI). Gérard Salton [Sal68] définit la recherche d'information comme étant "un domaine de la recherche qui s'intéresse à la structure, à l'analyse, à l'organisation, au stockage, à la recherche et à la découverte de l'information".

Une RI est déclenchée par l'expression d'un besoin d'information qu'un utilisateur exprime à travers une requête. L'information recherchée se trouve dans des documents. Pour accéder aux documents répondant à son besoin, l'utilisateur passe par l'intermédiaire d'un système de recherche d'information (SRI), ou moteur de recherche, qui se charge d'effectuer la comparaison entre une collection de documents et la requête.

Problématique

Comme en témoignent de récentes études dans le domaine de la RI [BH04], les performances de différents SRI sont très variables. Par exemple, le SRI A peut être très performant pour une requête donnée et être très médiocre pour une autre requête, alors que le SRI B peut avoir un comportement inverse pour les mêmes requêtes. Lors du séminaire sur l'accès fiable à l'information (RIA workshop) en 2004, Buckley et ses collègues [Buc04] ont montré que, maîtriser la variabilité des résultats de la recherche

[1] http ://atilf.atilf.fr/tlfi.htm

est complexe à cause d'un certain nombre de paramètres tels que la formulation de la requête et les techniques de recherche utilisées par les SRI . Le but de leur étude était d'arriver à expliquer les échecs des SRI (les résultats obtenus par 6 différents SRI ont été analysés) sur certaines requêtes.

La problématique que nous traitons dans cette thèse concerne l'étude des variabilités qui interviennent aussi bien au niveau de la requête que des techniques de recherche utilisées par les SRI . L'objectif vise à plus long terme de proposer un processus de recherche adaptatif qui prenne en compte le contexte de la recherche, dans le but d'en améliorer l'efficacité. L'hypothèse que nous faisons est qu'il existe des contextes dans lesquels un SRI sera plus adapté qu'un autre pour répondre à la requête de l'utilisateur. Pour cela, il est nécessaire de détecter ces différents contextes par une analyse des variables pouvant influer dans les résultats d'un SRI . Nous nous intéressons dans nos travaux au contexte de la requête, des collections de documents et des différentes approches de recherche mises en œuvre par les SRI .

Trois grandes questions reprennent les principaux aspects abordés lors de la Recherche d'information (RI) : **Quoi chercher ? Où chercher ? Comment chercher ?**

Quoi chercher ?

À travers cette question, l'objectif est de modéliser le besoin d'information de l'utilisateur. Dans le cadre de la RI , le besoin d'information est spécifié par un utilisateur au moyen d'une requête généralement exprimée par quelques mots. Les requêtes exprimées sous forme d'une liste de mots clés sont les plus couramment utilisées en RI . L'utilisateur dispose d'une multitude de possibilités pour exprimer son besoin d'information, de même que les systèmes peuvent analyser une requête de plusieurs manières différentes. Cependant, les mots-clés sont une vue limitée du besoin de l'utilisateur, et introduisent certains problèmes liés à la langue dans laquelle la requête est exprimée. En effet, un besoin mal exprimé ne peut pas être correctement satisfait. La compréhension du besoin d'information, que l'utilisateur exprime à travers une requête, est donc une tâche très difficile en RI . En effet, la requête n'est qu'une expression partielle d'un besoin mental de l'utilisateur. Belkin [BC92] parle dans ses travaux d'un état anormal de connaissances (*Anomalous State of Knowledge*) pour l'utilisateur, lorsque celui ci est confronté à un manque de connaissance sur un sujet. Cet état est l'élément déclencheur de la recherche de l'utilisateur. Plus la requête de l'utilisateur est précise, plus la recherche est efficace. D'après Bates [Bat86], "la probabilité que deux personnes utilisent le même terme pour désigner la même chose équivaut à moins de 20% ". Furnas quant à lui [FDD+88] stipule que "... la probabilité que deux personnes choisissent le même terme est comprise entre 7 et 18%". L'expression de la requête est donc garante de la rapidité de la recherche, et a des impacts sur son déroulement.

La principale difficulté de la recherche est que les SRI ne comprennent pas le langage humain, et n'ont pas une connaissance parfaite du contenu (en termes de sémantique) de la collection de documents qu'ils utilisent. De plus, une majorité de SRI traitent

les requêtes de la même manière (ensemble de mots-clés). Un de nos objectifs est de proposer un ensemble de traitements adaptés au type de requête, afin de prendre en compte leur variabilité.

Nous proposons dans nos travaux de modéliser les requêtes en faisant appel à des techniques de Traitement Automatique des Langues (TAL). Le TAL est une branche pluridisciplinaire, qui met en œuvre des modèles et des outils informatiques pour traiter le langage humain dans toute sa complexité, en appliquant des traitements linguistiques sur les textes. Cette complexité nécessite des mécanismes d'appariement plus adaptés afin de d'améliorer l'efficacité de la recherche. En effet, le fait qu'un document contienne les termes de la requête ne signifie pas que ce document soit pertinent pour l'utilisateur et doit être restitué car la même idée peut être exprimée de plusieurs manières différentes. Les travaux présentés dans [JM00] donnent une description assez complète du traitement automatique des langues.

Afin de proposer des traitements adaptés aux requêtes, nous proposons de mettre en place une typologie des requêtes. Cette typologie est effectuée en analysant un ensemble de caractéristiques linguistiques des requêtes (CL). Nous associons donc à chaque requête un certain nombre de CL qui sont extraites automatiquement des textes des requêtes. Ces CL sont regroupées en 3 grandes familles linguistiques [Lid98] qui sont la morphologie, la syntaxe, et la sémantique. La morphologie s'intéresse à la structure des termes. La syntaxe étudie les relations grammaticales qui existent entre les termes, et la sémantique étudie le sens des termes.

Où chercher ?

Dans le cadre de nos travaux, les documents textuels non structurés sont les supports de l'information que recherche l'utilisateur. Une collection de documents regroupe un très grand nombre de documents. Les collections de documents peuvent être statiques ou non. Les collections statiques contiennent un nombre fixe de documents et sont utilisées en général lors de l'évaluation de la recherche. On peut citer par exemple les collections adhoc de la campagne d'évaluation TREC (Text REtrieval Conference)[2] qui contiennent un certain nombre de documents fixes pendant toute la session d'évaluation. La RI se base sur les collections de documents pour retrouver les documents qui correspondent au mieux au besoin de l'utilisateur. Pour résumer, l'on peut dire que les collections de documents sont une source d'information "globale" dans laquelle l'utilisateur peut trouver satisfaction par rapport à son besoin d'information. Les documents constituent l'unité d'information retournée à l'utilisateur à l'issue du processus de RI . Les collections de documents permettent de centraliser un ensemble de documents afin de faciliter le traitement des SRI .

Un des atouts des campagnes d'évaluation est la masse considérable de documents qu'elles contiennent. Différents SRI testent leurs techniques de recherche en utilisant les collections des campagnes d'évaluation qui contiennent un ensemble de documents et un ensemble de requêtes prédéfinies. Ce contexte d'évaluation est favorable à la

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

comparaison de plusieurs SRI . Les masses de données des campagnes d'évaluation sont sous exploitées, et nous proposons d'exploiter les résultats obtenus par différents SRI lors de leur participation aux campagnes d'évaluation, en mettant en place des techniques de fusion de données.

Comment chercher ?

Un certain nombre de mécanismes sont mis en place à travers les SRI pour permettre la mise en correspondance entre les documents et les requêtes. Un SRI est composé de deux parties très liées. La première partie est visible et constitue une interface entre l'utilisateur et les collections de documents. C'est grâce à ce module que l'utilisateur spécifie sa requête, et reçoit en résultat une liste de documents ordonnés par degré de pertinence supposée.

La deuxième partie est "cachée" et renferme un module théorique permettant de faire la mise en correspondance des requêtes et de la collection. Il existe un certain nombre de modèles théoriques dans la littérature comme le modèle booléen [Sal71b] dans lequel les documents et requêtes sont représentés sous forme de termes reliés par des opérateurs booléens, le modèle vectoriel [SL68], [Sal71b] qui considère les documents et les requêtes comme des vecteurs pondérés, le modèle probabiliste [RSJ76] qui associe à chaque document et requête un score de probabilité, etc.

Dans cette thèse, nous proposons un processus de recherche capable de s'adapter à différents contextes. Ce processus fonctionne par analogie comme un méta-moteur de recherche en mettant en collaboration plusieurs SRI différents. Nous utilisons dans ce processus des techniques de fusion, afin de combiner les résultats des SRI utilisés. La fusion est une technique qui est utilisée en RI pour améliorer l'efficacité de la recherche, mais généralement les requêtes sont gérées de la même manière à travers un processus de fusion "global". Nous proposons une fusion locale prenant en compte les CL des requêtes ainsi que leur typologie. Retourner plus d'une liste de documents (provenant de plusieurs collections) à l'utilisateur rend la tâche de l'utilisateur plus difficile car ce dernier se limite en général aux 10 premiers résultats qui lui sont restitués. Lorsque la même requête est soumise à des systèmes différents, ceux ci restituent des listes différentes ayant une intersection vide ou non, d'où l'intérêt d'utiliser des techniques de fusion de données lors de la recherche. Les techniques de fusion quant à elles s'intéressent à la variabilité des résultats lorsque différentes techniques de recherche sont utilisées pour dans le cadre de la RI . L'utilisation de ces techniques ne permet cependant pas de comprendre ni d'expliquer les causes de la variabilité des résultats de la recherche, ainsi que leur impact sur les performances des systèmes.

Nous proposons de suivre le plan suivant, découpé en deux grandes parties :

La première partie présente le contexte dans lequel nos travaux se situent à travers un état de l'art. Elle se décompose en 3 chapitres. Dans la deuxième partie, nous présentons en détail notre contribution, à travers 4 chapitres.

Première partie

Chapitre 1

Dans ce chapitre nous présentons les différents principes de la RI . Nous donnons dans un premier temps les concepts de base de la RI (cf. section 1.2) que nous avons résumés à travers les trois questions que nous avons présentés ci-dessus. Nous abordons ensuite le principe de l' indexationn (cf. section 1.3) qui est une étape préliminaire importante en RI . Dans la section 1.4, nous indiquons quelques méthodes utilisées par les SRI pour calculer la similarité entre les documents et les requêtes. Afin de mesurer l'efficacité de la recherche, une phase d'évaluation est mise en œuvre. Nous donnons plus de détails sur cette évaluation dans la section 1.5. Nous terminons ce chapitre en présentant le principe des campagnes d'évaluation en RI , plus spécifiquement la campagne TREC qui est le cadre dans lequel nos travaux sont menés.

Chapitre 2

Ce chapitre traite de la linguistique en RI . Dans le cadre de nos travaux, nous appliquons certaines techniques linguistiques sur les requêtes, afin d'établir leur profil linguistique que nous utilisons dans nos expérimentations. Nous présentons dans un premier temps les traitements linguistiques tels qu'ils sont utilisés en RI lors du processus d' indexation (section 2.2) et lors de la reformulation des requêtes (section 2.3). Nous présentons ensuite les travaux en RI qui utilisent des traitements linguistiques afin de caractériser les requêtes (section 2.4). Cette dernière section justifie l'utilisation de caractéristiques linguistiques dans la représentation de requêtes dans le cadre de systèmes de RI adaptatifs (qui adapteraient les traitements effectués) à la requête, objectif à plus long terme auquel contribue cette thèse.

Chapitre 3

Ce chapitre introduit le principe de la fusion tel qu'utilisé en RI . Nous présentons les deux principaux types de fusion à savoir la fusion de collections (cf. section 3.2) et la fusion de données (cf. section 3.3). Nous abordons ensuite le couplage classification/ fusion en RI (section 3.4).

Deuxième partie

Chapitre 4

Dans ce chapitre, nous présentons les différentes collections de données dont nous disposons pour nos expérimentations. Dans la section 4.2, nous présentons les collections TREC que nous avons utilisées. Dans un deuxième temps, nous nous intéressons à l'analyse de la variabilité des requêtes. L'étude de cette variabilité est basée sur l'utilisation des CL des requêtes afin de les regrouper. Ces CL sont issues des travaux présentés dans [MT05]. Nous détaillons les différentes CL des requêtes ainsi que leur mode d'extraction dans la section 4.3. Ces CL nous permettent de classifier les requêtes en fonction de leur

profil linguistique. Nous terminons ce chapitre en présentant les expérimentations que nous avons effectuées afin de valider le principe de classification des requêtes sur la base de leur CL. Nous avons mis en œuvre des méthodes statistiques basées sur l'Analyse en Composante Principale (ACP), la classification ascendante hiérarchique (CAH), ainsi que la méthode des k-means. Cette approche de classification des requêtes en fonctions de leurs CL, et le contexte dans lequel elle est utilisée, la rend originale en RI. Nous supposons que si l'on peut regrouper les requêtes en fonctions de leurs traits linguistiques, alors il est possible de trouver la stratégie de recherche la plus adaptée pour chacune des classes. Ce principe a aussi été utilisé dans [HO04b], [HO03a] mais les auteurs utilisent 3 caractéristiques statistiques pour classifier les requêtes. Il est toutefois important de noter que notre méthode nécessite pour fonctionner que les requêtes soient exprimées dans un format permettant l'application de traitements linguistiques. Dans notre approche, les requêtes sont caractérisées par 13 CL et nous appliquons sur les classes de requêtes obtenues, des techniques de fusion probabiliste (cf. chapitre 6). La méthode que nous proposons a été expérimentée dans le cadre de la campagne d'évaluation TREC, dans laquelle un ensemble de requêtes et de SRI est disponible, ainsi qu'un ensemble de jugements de pertinence. Le format des requêtes de TREC est adapté à l'application de traitements linguistiques. Nous terminons ce chapitre en présentant nos expérimentations dans la section 4.4.

Chapitre 5

Ce chapitre présente une étude que nous avons menée sur la fusion de données. Dans un premier temps (section 5.2), nous nous intéressons à 2 techniques simples de fusion que sont l' UNION et l'INTERSECTION. Ces méthodes de fusion ont respectivement des impacts sur le rappel et la précision obtenus à l'issue de la fusion. Nous nous situons dans le cas où aucune information sur le degré de similarité entre la requête et les documents n'est disponible. Dans ce contexte, un document est soit pertinent, soit non pertinent. Notre objectif est de mesurer l'impact de telles stratégies de fusion (union et intersection) sur les résultats de la recherche. Nous proposons dans un deuxième temps une stratégie de fusion moins systématique que l' union et l' intersection afin de prendre en compte un certain nombre de variabilités telles que le taux de chevauchement [Lee97] et de la typologie des requêtes (faciles ou difficiles) (cf. section 5.3). Nous terminons ce chapitre en présentant les différentes expérimentations que nous avons menées ainsi que les résultats que nous avons obtenus (section 5.4.1).

Chapitre 6

Dans ce chapitre, notre proposition est relative à l'adaptation de la fusion de données aux requêtes. Nous présentons deux algorithmes de fusion probabilistes que nous avons proposés qui sont MaxProb et MaxProbSeg (section 6.2). Dans notre proposition, nous utilisons les CL pour classer les requêtes, et nous analysons les performances des systèmes pour chacune de ces classes de requêtes. L'idée de cette classification est d'une part d'arriver à classer automatiquement les requêtes en fonction de leurs CL, et d'autre

part d'analyser les variabilités des SRI afin de proposer la (les) meilleure(s) stratégie(s) de recherche à utiliser. Le choix du système à utilisé est effectué en appliquant les algorithmes MaxProb et MaxprobSeg sur les classes de requêtes. La section 6.4 présente nos expérimentations et les résultats que nous avons obtenus.

Chapitre 7

Dans ce chapitre, nous proposons d'analyser la masse d'information dont nous disposons dans le cadre de notre étude, afin de faire ressortir les différentes relations qui existent entre ces données. L'idée est d'appliquer la technique de l'analyse canonique (AC) à l'ensemble de données dont nous disposons (CL des requêtes, collections de documents, performances des SRI , etc). Ce couplage AC/RI a donné lieu à une collaboration entre notre équipe et des collègues mathématicien à travers un projet qui a été nommé ACRIC (Analyse Canonique et RI Contextuelle).

Première partie

Mise en contexte : Recherche d'Information, linguistique et fusion

Chapitre 1

Les principes de base de la RI

1.1 Introduction

Gérard Salton [SM83] définit la **recherche d'information** (RI) comme étant "*un domaine de la recherche qui s'intéresse à la structure, à l'analyse, à l'organisation, au stockage, à la recherche et à la découverte de l'information*". La RI est déclenchée par l'expression d'un besoin d'information de l'utilisateur, spécifié par l'intermédiaire d'une requête. L'information recherchée peut se trouver dans des documents, et pour y accéder, l'utilisateur a besoin de passer par un intermédiaire communément appelé **système de recherche d'information** (SRI) qui compare un ensemble de documents avec la requête.

Nous abordons dans ce chapitre les principes de base de la RI, et nous présentons dans un premier temps les principaux concepts (section 1.2) qui sont : la requête (section 1.2.1), le document et la *collection de documents* (section 1.2.2), et enfin les SRI (section 1.2.3).

Pour accéder aux *collections de documents*, les SRI utilisent une technique appelée *indexation* qui leur permet d'accéder plus rapidement aux différents documents dans la collection. Nous verrons cette technique en détail dans la section 1.3 à travers deux étapes principales : l'extraction des index (section 1.3.1), et les techniques de pondération et de normalisation des index (section 1.3.2).

Lorsque la requête de l'utilisateur est soumise au SRI, ce dernier met en correspondance les documents de la collection et la requête, grâce à des calculs de similarité. Ces calculs permettent d'affecter un score aux documents en fonction de leur ressemblance avec la requête. Nous présentons les principales techniques de calcul de similarité en section 1.4.

L'évaluation est une étape importante de la RI. Elle permet de mesurer l'adéquation entre le besoin de l'utilisateur et les documents qui lui sont retournés. La section 1.5 s'intéresse à cet aspect. Nous commençons par présenter la notion de pertinence (section 1.5.1), puis nous présentons les mesures qui sont utilisées en RI (section 1.5.2 et 1.5.2.6).

Les concepteurs des SRI ne peuvent pas prédire l'efficacité de leurs systèmes sur tout type de collections. Généralement, les techniques utilisées par les SRI sont évaluées sur une collection fixe. Les différentes campagnes d'évaluation en RI permettent d'évaluer les SRI sur des bases homogènes. Nous présentons dans la section 1.6 la principale campagne d'évaluation en RI : TREC [1]. Nous terminons ce chapitre en section 1.7 en présentant les limites des approches de RI actuelles et quelques pistes pour compenser ces lacunes ; nous introduisons notamment l'utilisation de la linguistique et les techniques de fusion.

1.2 Concepts de base de la RI

Quoi chercher ? *Où* chercher ? *Comment* chercher ? Il s'agit des trois questions fondamentales auxquelles un système de RI doit répondre. La première question introduit la notion de *requête* que nous présentons dans la section suivante. La requête permet à l'utilisateur de spécifier son besoin. Plusieurs réponses peuvent correspondre à la requête de l'utilisateur. La question *Où chercher ?* (section 1.2.2) s'attache quant à elle à la notion de document et de *collection de documents* qui sont utilisés par les *SRI* (section 1.2.3) pour retourner à l'utilisateur une liste de documents correspondant à sa requête. Nous terminons cette section en présentant les différentes stratégies de recherche (section 1.2.3) mises en place dans les SRI pour répondre à la troisième question et retrouver les documents pertinents répondant à la requête de l'utilisateur.

1.2.1 La requête

Dans le cadre de la RI, le besoin d'information est spécifié par un utilisateur au moyen d'une requête généralement exprimée par quelques mots. Les requêtes exprimées sous forme d'une liste de mots clés sont les plus couramment utilisées en RI ; les mots clés pouvant être reliés par des opérateurs booléens (ET, OU, NON,...). Les requêtes peuvent aussi être exprimées en langage naturel. C'est le cas par exemple des requêtes issues de la campagne d'évaluation TREC. Nous nous intéressons dans cette thèse à ce type de requêtes sur lesquelles nous appliquons un certain nombre de traitements linguistiques (cf. chapitre 2 et nos contributions).

L'expression du besoin de l'utilisateur est une étape cruciale dans la recherche d'information [SJ95] et engendre des répercussions sur le déroulement de la recherche ainsi que sur la qualité des réponses qui sont données. Le but que l'utilisateur cherche à atteindre en spécifiant sa requête est d'obtenir des informations conformes à son besoin. Son besoin étant exprimé à l'aide d'une requête, l'utilisateur doit choisir les "bons" mots dans sa requête pour maximiser ses chances d'obtenir des documents pertinents car son besoin d'information est d'abord mental. Les documents répondant à son besoin sont soit *pertinents* soit *non pertinents*. Nous définirons plus précisément dans la section 1.5.1 la notion de pertinence. À l'issue de la RI, une liste de documents est retournée à l'utilisateur en fonction de leur pertinence supposée.

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

Nous présentons dans la section suivante les sources d'information utilisées lors de la RI pour répondre à la requête de l'utilisateur.

1.2.2 Le document et la *collection de documents*

Dans le cadre de nos travaux, les documents textuels sont les supports de l'information que recherche l'utilisateur. Les documents textuels peuvent exister sous une forme structurée (documents html, documents xml (INEX [2]).) ou non. Nos recherches s'intéressent plus spécifiquement aux documents textuels non structurés. Dans la suite de ce manuscrit, nous utilisons le terme *document* pour nommer ce type de documents.

Pour résumer, on peut dire que les *collections de documents* sont une source d'information "globale" dans laquelle l'utilisateur pouura satisfaire son besoin d'information, alors que les documents constituent l'unité d'information retournée à l'utilisateur à l'issue du processus de RI.

Pour faire correspondre un ensemble de documents à une requête, il est nécessaire de disposer d'une technique de mise en correspondance. Nous présentons dans la section suivante la notion de *SRI* qui permet entre autres, cette mise en correspondance.

1.2.3 Les systèmes de recherche d'information

Un certain nombre de mécanismes sont mis en place à travers des SRI pour permettre la mise en correspondance entre les documents et les requêtes. Les SRI sont des moteurs de recherche chargés de retrouver les documents pertinents pour une requête donnée. Un SRI est composé de deux parties très liées. La première partie est visible, et constitue une interface entre l'utilisateur et les *collections de documents*. C'est grâce à cette interface que l'utilisateur spécifie sa requête, et reçoit en résultat une liste de documents ordonnés par degré de pertinence supposé. La deuxième partie est "cachée" et repose sur un modèle théorique permettant de faire la mise en correspondance des requêtes et de la collection.

Il existe un certain nombre de modèles théoriques dans la littérature les plus connus étant le modèle booléen [Sal71b], le modèle vectoriel [SL68], [Sal71b], et le modèle probabiliste [RSJ76]. Dans le modèle booléen, les requêtes sont représentés sous forme de termes reliés par des opérateurs booléens (ET, OU, NON, ...). Le modèle vectoriel [SM86] considère les documents et les requêtes comme des vecteurs pondérés, chaque élément du vecteur représentant le poids d'un terme dans la requête ou le document. Le modèle probabiliste tente d'estimer la probabilité qu'un document donné soit pertinent pour une requête donnée.

Il existe de nos jours des modèles plus évolués que les modèles qui viennent d'être présentés. On peut citer par exemple les modèles de logique floue [OMK91] ou les modèles booléens étendus [SFW83] (vs modèles booléens), le modèle vectoriel généralisé [WZW85], les modèles *LSI* (*Latent Semantic Indexing*) [FDD+88] ou neuronaux

[2] http ://inex.is.informatik.uni-duisburg.de/

[WH91] (vs modèles vectoriels), les modèles bayésiens [Pea88], inférentiels [TC90], les réseaux de croyance [RNM96] (vs modèles probabilistes). Nous limitons notre présentation aux trois premiers modèles.

1.2.4 Le processus de RI

Les *SRI* tentent de fournir une réponse à un besoin spécifié par l'utilisateur, en mettant en correspondance la requête et les documents d'une collection. Le processus de RI décrit les différentes étapes à travers lesquelles une liste de documents est restituée à l'utilisateur. Ce processus est souvent appelé *processus en U* [BC92], et se décompose en deux étapes principales : *l'indexation* ([DDL$^+$90], [SJ95], [Sal71a]) et *la recherche*.

La première étape (ou *indexation*) peut être vue comme une phase de *préparation* pendant laquelle le *SRI* analyse chaque document de la collection afin d'en extraire les mots les plus discriminants apellés *index* [VR79]. L'ensemble des *index* d'un document constitue son *descripteur* et permet un accès rapide au document.

La deuxième étape (ou *recherche*) décrit la manière dont le *SRI* compare les *descripteurs* des documents avec ceux de la requête, et calcule un score de similarité entre le document et la requête. Le calcul de similarité est effectué à l'aide de modèles de recherche tels que le modèle *booléen*, le modèle *vectoriel*, ou le modèle *probabiliste* par exemple. Nous donnons plus de détails sur ces modèles dans la section 1.4. La figure 1.1 représente un schéma du processus de RI.

Dans la figure 1.1, 2 niveaux conceptuels sont représentés et concernent aussi bien les documents que les requêtes. Les niveaux conceptuels décrivent les grandes phases du processus de recherche.

Le niveau 1 correspond à la phase *d' indexation*. Cette étape est détaillée dans la section 1.3. Durant cette étape, la requête Q est soumise au SRI sous sa forme brute[3] et, après *l' indexation*, la requête et les documents sont représentés chacun par leurs descripteurs. La pondération permet de traduire l'importance que le *SRI* accorde aux différents index. Elle fait souvent suite à l' indexation.

Le niveau 2 représente la phase de recherche. Ce niveau est matérialisé par le calcul du degré de similarité entre la requête et les documents. Grâce à cette mesure de similarité, une liste de documents potentiellement pertinents est restituée à l'utilisateur. L'utilisateur juge alors la pertinence des documents qui lui sont restitués.

La requête et la collection de documents sont des éléments qui peuvent être considérés indépendants des SRI. En effet, quel que soit le SRI utilisé pour la recherche, la structure initiale de la requête (formulation de la requête) et des documents (contenu des documents) reste inchangée. La manière d'indexer les requêtes et les documents

[3] c'est à dire qu'aucun vocabulaire spécifique n'est utilisé et aucune transformation n'est effectuée sur la requête ni sur les documents

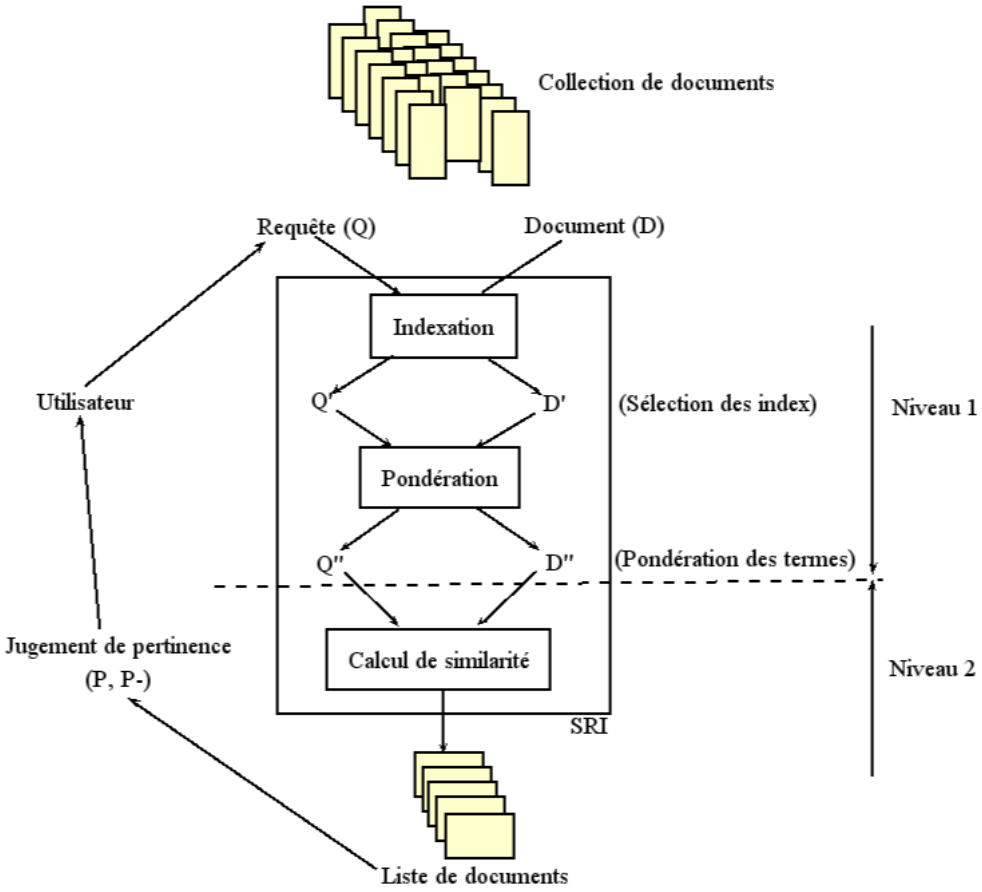

FIG. 1.1 – Schéma de la RI inspiré de [Fur92]

est fortement liée au SRI utilisé. La description des requêtes est faite lors des phases d'analyse de la requête et la description des documents pendant la phase d' indexation des documents. Nous détaillons dans la section suivante le principe de l' indexation.

1.3 L' indexation en RI

L' indexation consiste à extraire des documents les mots les plus discriminants encore appelés index. Cette première tâche est généralement effectuée en marge du processus de recherche car, la construction des index peut être assez longue en fonction du nombre de documents de la collection ainsi que de la taille des documents. D'après [SJ92], les index ont un caractère réducteur car tous les termes d'un document ne sont pas importants à prendre en compte pour la recherche. L' indexation peut se faire de 3 manières différentes : manuellement, de manière semi-automatique, ou de manière automatique.

L' indexation manuelle

Elle est faite par un spécialiste du domaine. Même si certaines variabilités peuvent exister entre 2 spécialistes dans le choix des index, ce genre d' indexation a l'avantage d'être précis dans les résultats [RFN99]. En effet, les spécialistes d'un domaine choisissent de meilleurs termes pour indexer les documents, étant donné leur connaissance du domaine. Il faut cependant noter que le temps d' indexation manuelle est très élevé et donc rend l' indexation manuelle coûteuse en temps.

L' indexation semi-automatique

L' indexation est réalisée en deux étapes. Dans la première, les index sont automatiquement extraits des textes et sont ensuite transmis aux spécialistes du domaine qui les valident en utilisant un *thésaurus* [4] ou base terminologique [MdG91].

L' indexation automatique

Elle ne nécessite pas d'intervention humaine lors du processus d' indexation [MK60]. L' indexation est alors composée d'un ensemble de traitements automatisés tels que par exemple l'extraction des index, l'utilisation d'un *anti-dictionnaire* pour supprimer les mots de liaison et les autres mots non discriminants ainsi que les différents processus de suppression des variantes des mots.

[APC01] compare dans son article les différents types d' indexation présentés précédemment. Ses conclusions sont que les avantages et les inconvénients de chacune des méthodes sont liés au domaine et à la collection utilisée. Nous nous intéressons dans cette thèse à l' indexation automatique. L'automatisation du processus d' indexation nécessite de choisir les "*bons index*" et donc de définir les conditions sous lesquelles un terme est choisi comme index. Nous présentons dans la section suivante comment sont extraits les index ainsi que les techniques de pondération utilisées par les *SRI*.

1.3.1 Extraction automatique des index

Le document est l'unité informationnelle utilisée lors de la RI. Les *SRI* associent aux documents un ensemble de descripteurs correspondant à une liste de mots clés ou *index*. Les termes choisis comme *index* traduisent le contenu du document, et sont extraits directement des documents. La question que nous traitons dans cette sous-section est : *comment extraire les index ?*.

L'algorithme utilisé par les *SRI* pour extraire les index consiste à segmenter le texte en un ensemble de termes. Cette segmentation est réalisée à travers une analyse lexicale du texte [5] où le choix du délimiteur joue un rôle important dans la qualité de la

[4] Un *thésaurus* est "*une liste structurée de concepts, destinés à représenter de manière univoque le contenu des documents et des questions dans un corpus donné*" [AG92].

[5] Nous présentons en détail l'analyse lexicale dans le chapitre 2 de cette thèse

segmentation. Par exemple, *Etats-Unis*, *I.R.I.T*, *3D*, *brosse à dent* sont tous des termes ayant des délimiteurs différents. En reprenant l'exemple du terme *Etats-Unis*, si le tiret ("-") est considéré comme délimiteur, alors le terme *Etats-Unis* n'est plus considéré comme un terme composé mais comme deux termes différents. Cette étape d'extraction n'est pas exempte d'erreurs ; les systèmes ont souvent recours à des ressources externes comme des *thésaurus* ou des *dictionnaires* (cf. chapitre 2) pour valider le choix des index. La validation des index permet de ne pas considérer tous les termes d'un document comme des index. En effet, lors de l'extraction des index, les termes les plus discriminants doivent être détectés et les *mots vides* (tels que les pronoms personnels, les articles, les mots de liaison, ou les prépositions) doivent être éliminés. L'élimination des mots vides se fait grâce à l'utilisation d'un *anti-dictionnaire*[6]. Une technique complémentaire pour choisir les index est la *racinisation* [Por80]. Elle consiste à éliminer les différentes variations morphologiques d'un mot en extrayant la racine du mot. Par exemple *formé*, *former*, *formation*, *formateur* sont reconnus comme provenant d'une même racine. D'autres techniques plus évoluées ne considèrent pas les termes de manière isolée mais tentent d'extraire des *groupes de mots* [Ril95] pour former les index. Une autre technique proche de la *racinisation* considère des séquences de n caractères comme index : c'est la technique des *n-gram* [Dam95]. Nous détaillerons dans le chapitre 2 les traitements linguistiques qui sont utilisés dans le processus de RI.

Les différents index qui sont extraits n'ont pas la même importance dans le document ou dans la requête. La pondération est une technique qui permet d'affecter un score de préférence aux index. Nous détaillons ce principe dans la section suivante.

1.3.2 Pondération des index

Le but de la pondération est double : réduire la taille de l'ensemble des *descripteurs* des documents et des requêtes (nombre d'index), et accorder une certaine importance aux index à travers des poids.

La pondération consiste à affecter une valeur aux index en fonction de leur caractère discriminant, ou de leur degré d'informativité dans les documents. La pondération se base sur des mesures statistiques locales (c'est à dire liées au document), ou globales (c'est à dire liées à la collection). Le moyen courant d'associer un poids à un index consiste à calculer sa fréquence d'apparition dans le document ou dans la collection. La pondération locale est notée *TF* (ou *Term Frequency*) et permet de déterminer la fréquence d'apparition d'un terme dans le document ou dans la requête. La pondération globale est notée *IDF* (ou *Inverse document Frequency*) et représente la fréquence d'apparition d'un terme dans toute la collection. Par exemple, un terme qui apparaît très fréquemment dans toute la collection, et très peu dans un document, est moins informatif qu'un terme qui apparaît fréquemment dans un document et très peu dans toute la collection. [RSJ76] a été parmi les premiers à utiliser la distribution statistique des termes dans les documents comme moyen de pondération. L'utilisation de la notion

[6] Un anti-dictionnaire contient une liste de mots vides à éliminer.

de fréquence pour discriminer les termes a été initialement étudiée par Zipf [Zip49]. La loi de Zipf s'énonce formellement de la manière suivante :

$$rang * fréquence = constante. \tag{1.1}$$

En d'autres termes, la formule 1.1 représente le fait que lorsque l'ensemble des termes d'un document est ordonné par fréquence décroissante, la fréquence d'un terme est inversement proportionnelle à son rang. Zipf constate que peu de termes sont utilisés fréquemment dans les documents alors que la plupart ou un grand nombre de termes sont utilisés peu fréquemment. Partant de cette loi de Zipf, la pondération peut être locale ou globale.

La pondération locale est effectuée dans le document. Elle est caractérisée par la mesure TF (terme frequency) qui permet de favoriser les termes qui apparaissent le plus fréquemment dans le document. Par exemple, [Lee95] propose dans son approche une normalisation du poids des termes afin de limiter l'effet des termes trop fréquents.

Un même terme présent dans deux documents peut avoir des valeurs de pondération différentes. Par exemple, un terme apparaissant dans tous les documents de la collection n'est pas discriminant par rapport à un terme qui n'apparaît que dans quelques documents. Il est donc nécessaire d'évaluer l'importance d'un terme non seulement par rapport à sa fréquence d'apparition dans un document mais également dans toute la collection. La pondération globale utilise la mesure *IDF* présentée dans [SJ72] pour calculer l'importance d'un terme dans la collection. La pondération globale est exprimée à travers le facteur nommé IDF (Inverse document Frequency). IDF mesure l'importance d'un terme dans toute la collection. Il est généralement exprimé par la formule 1.2

$$IDF = log\frac{N}{n}. \tag{1.2}$$

où N représente le nombre total de documents dans la collection et n le nombre de documents contenant le terme considéré.

La mesure TF*IDF donne une bonne approximation de l'importance du terme dans le document. Cependant, cette mesure ne tient pas compte de la longueur du document qui est un élément important. En effet, un terme dans un document long a plus de chances d'apparaître plusieurs fois qu'un autre terme dans un document court. Plus le document est long, plus les termes utilisés se répètent. La longueur des documents peut aussi induire l'utilisation d'un grand nombre de termes pour décrire un sujet. Pour pallier ce type d'inconvénients, certains travaux comme ceux de Robertson [RW94] et Singhal [SSMB96] normalisent la pondération en intégrant la taille des documents dans le calcul. Une autre mesure de pondération souvent utilisée en RI est la pondération BM25. C'est un modèle de pondération basé sur le modèle probabiliste [RWHB$^+$95] (cf. 1.4.2).

Plus le poids d'un terme est élevé, plus ce terme est important pour la requête ou le

document. Nous présentons dans la section suivante les principales mesures de similarité ainsi que la manière dont elles sont calculées en RI.

1.4 Calcul de similarité entre la requête et les documents

Les mesures de similarité permettent aux *SRI* de calculer un score de pertinence des documents. Ces scores de pertinence se basent sur la notion de pondération que nous avons présentée précédemment pour comparer les documents et les requêtes. Les mesures de similarité sont formalisées dans les systèmes par des modèles théoriques sous-jacents. Les modèles de base qui sont utilisés en RI sont le modèle booléen ([Sal71b], [SM83], [SFW83]), le modèle vectoriel, et le modèle probabiliste. Nous présentons les deux derniers modèles (vectoriel et probabiliste) qui sont les modèles utilisés dans le cadre de nos travaux. Nous précisons dans chaque cas comment les mesures de similarité sont calculées.

1.4.1 La similarité vectorielle

Le modèle vectoriel est le modèle le plus populaire en RI du fait de sa facilité de mise en œuvre et des performances qu'il permet d'obtenir. Le modèle vectoriel propose une pondération positive non binaire des termes de la requête et des documents. Cette pondération est ensuite utilisée pour calculer le degré de similarité entre les documents et la requête. Le modèle vectoriel ordonne les documents par degré de similarité décroissant en fonction du degré de correspondance entre les documents et la requête. Les requêtes et les documents sont représentés dans un espace vectoriel des termes d' indexation, sous forme de vecteurs de termes pondérés. Par exemple la requête Q est représentée par le vecteur $\vec{Q} = (w_{1,q}, w_{2,q}, ..., w_{n,q})$ où n représente le nombre maximal des termes index de la requête, et $w_{i,q}$ le poids du terme t_i dans la requête Q. Les documents sont aussi représentés par des vecteurs $\vec{D} = (w_{1,j}, w_{2,j}, ..., w_{n,j})$ avec $w_{i,j}$ représentant le poids du terme t_i dans le document D_j. La mesure de similarité entre la requête et le document est donnée par le cosinus de l'angle qui existe entre $\vec{Q}$ et $\vec{D}$:

$$sim(D_j, Q) = \frac{\vec{D_j}.\vec{Q}}{\left|\vec{D_j}\right| . \left|\vec{Q}\right|} = \frac{\sum_{i=1}^{n} w_{i,j} * w_{i,q}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2} * \sqrt{\sum_{i=1}^{n} w_{i,q}^2}} \tag{1.3}$$

Dans la formule (1.3), $\left|\vec{Q}\right|$ et $\left|\vec{D_j}\right|$ sont les normes des documents et des vecteurs. Comme les poids sont des valeurs positives, la mesure du cosinus de l'angle formé par le document et la requête varie entre 0 et 1. Cela permet de classer les documents en fonction de leur degré de similarité avec la requête.

Les avantages du modèle vectoriel sont liés d'une part au modèle de pondération qu'il utilise et qui permet d'améliorer les performances de la recherche. D'autre part, le modèle vectoriel permet de retrouver des documents qui répondent partiellement à la requête de l'utilisateur et permet aussi d'établir un classement des documents en fonction de leur degré de similarité avec la requête. D'autres variantes du modèle vectoriel

ont été proposées dans la littérature comme par exemple le modèle vectoriel généralisé [WZW85], les modèles *LSI* (*Latent Semantic Indexing*) [FDD+88] ou neuronaux [WH91]. Nous limitons notre présentation au modèle vectoriel de base.

Une autre catégorie de modèles basés sur l'utilisation de mesures de probabilité pour calculer la similarité entre les documents et la requête est présentée dans la section suivante. Il s'agit du modèle probabiliste de calcul de similarité.

1.4.2 La similarité probabiliste

Ce modèle utilise la théorie des probabilités pour calculer les différents scores de pertinence des documents.

Considérons une requête Q et un document D_j issu de la *collection de documents*. Le modèle probabiliste estime la probabilité que le document D_j soit pertinent pour la requête Q. Le calcul de ces probabilités nécessite de disposer de deux ensembles de données : *un ensemble de documents pertinents connus a priori* (N), et un ensemble de documents non pertinents pour la requête. Le calcul des probabilités consiste alors à affecter à chaque document un score de similarité qui correspond à la formule 1.4 ([Fur92]) :

$$\frac{P(D_j \textit{ pertinent pour } Q)}{P(D_j \textit{ non pertinent pour } Q)} \tag{1.4}$$

De manière plus formelle, soit N l'ensemble de documents que l'on sait être pertinents pour la requête Q et $\bar{N}$ l'ensemble des documents non pertinents pour Q. Soit $P(N|\vec{D_j})$ la probabilité que le document D_j soit pertinent pour la requête Q et $P(\bar{N}|\vec{D_j})$ la probabilité que le document D_j ne soit pas pertinent pour Q. La similarité entre le document D_j et la requête Q correspond alors à la formule 1.5 :

$$sim(D_j, Q) = \frac{P(N|\vec{D_j})}{P(\bar{N}|\vec{D_j})} \tag{1.5}$$

En utilisant le théorème de Bayes on obtient :

$$sim(D_j, Q) = \frac{P(\vec{D_j}|N) * P(N)}{P(\vec{D_j}|\bar{N}) * P(\bar{N})} \tag{1.6}$$

Dans la formule 1.6, $P(\vec{D_j}|N)$ représente la probabilité de sélectionner le document D_j parmi la liste des documents pertinents, et P(N) représente la probabilité qu'un document choisi dans la collection soit pertinent.

La pondération *BM25* est basée sur un modèle probabiliste pour calculer la similarité entre les documents et la requête. Pour cela, les poids affectés aux termes sont une combinaison des mesures $Okapi_{tf}$ et *IDF*, et se calculent de la manière suivante :

$$Okapi_{tf} = \frac{TF}{TF + k_1((1-b) + b\frac{dl}{dlavg})} \tag{1.7}$$

où TF est la fréquence du terme et dl (resp. dlavg) la longueur (resp. longueur moyenne) des documents. k_1 et b sont des paramètres qui peuvent être modifiés.

Le score du document D peut être alors calculé de la manière suivante en utilisant la formule BM25 :

$$BM25(Q, D_j) = \sum_{t \in Q \cap D_j} (Okapi_{tf} * IDF_t * Freq_{Q_t}) \tag{1.8}$$

avec $Freq_{Q_t}$ la fréquence du terme t dans la requête Q. BM25(Q,D) correspond alors à une mesure de similarité entre le document D_j et la requête Q.

D'autres extensions du modèle probabiliste existent. Nous ne les présentons pas dans ce manuscrit, mais pour plus de détails on pourra consulter les articles de référence ([Pea88] pour les réseaux bayésiens, [TC90] pour les réseaux inférentiels, [RNM96] pour les réseaux de croyance, ...).

1.5 Adéquation entre le besoin d'information et la RI : évaluation de la recherche

L'évaluation des SRI est une préoccupation dans la communauté de la RI. Cette évaluation peut se faire de deux manières différentes en fonction des éléments que l'on souhaite mesurer. Certains critères comme la quantité d'espace utilisée, le temps de réponse du système, la puissance de calcul peuvent être utilisés pour mesurer la performance des systèmes. Cependant, ce type d'évaluation ne tient pas compte du contexte de la recherche. Un SRI est utilisé dans le but de répondre à un besoin donné, spécifié par un utilisateur. L'efficacité d'un système réside donc dans sa capacité à retrouver les documents pertinents pour l'utilisateur. Nous présentons dans la sous-section suivante (section 1.5.1) la notion de pertinence qu'il est important de définir. Nous indiquons ensuite les mesures usuelles que la communauté de la RI utilise pour mesurer la pertinence des réponses des systèmes : le *rappel* et la *précision* (section 1.5.2.1) ainsi que la mesure de précision moyenne (section 1.5.2.3).

1.5.1 La notion de pertinence

La notion de pertinence est à la fois intimement liée au jugement individuel d'un utilisateur et estimée par les SRI. Nous pouvons donc distinguer deux types de pertinence : la pertinence (ou mesure de ressemblance) système, et la pertinence utilisateur.

La pertinence (ou mesure de ressemblance) système

Pour être en mesure de répondre de manière efficace à la requête de l'utilisateur, les SRI doivent s'appuyer sur un modèle de pertinence qui leur permet de calculer pour chaque document un score de pertinence. La pertinence apparaît donc ici non pas comme une notion subjective, mais comme une valeur numérique calculée par les SRI. Cette *pertinence système* a cependant des limites car elle est estimée à partir d'un score de

ressemblance entre la requête et les documents, et détermine une pertinence supposée des documents pour l'utilisateur.

La pertinence utilisateur

C'est une notion subjective [SM83] car elle dépend du niveau de satisfaction que l'utilisateur tire de la liste de documents qui lui est restituée par le système. En effet, deux utilisateurs différents ayant soumis la même requête au SRI ne jugent pas de la même manière les réponses du système. Dans le cas où le jugement de pertinence n'est pas absolu (c'est-à dire que l'utilisateur dit si le document est pertinent ou non pertinent) mais donné par un degré de pertinence des documents, le désaccord entre plusieurs utilisateurs est nettement plus prononcé. Cela est dû au fait que les besoins sont différents et que le même besoin peut être exprimé différemment en fonction de l'utilisateur. De plus, l'interprétation que l'utilisateur fait des documents qu'il reçoit dépend en partie de ses connaissances personnelles et de son expérience, ainsi que du contexte dans lequel s'effectue sa recherche [HEH+95]. La pertinence *utilisateur* permet à ce dernier d'exprimer sa satisfaction par rapport aux documents potentiellement pertinents, que le système lui restitue.

Tout l'enjeu du processus de RI est de minimiser la distance entre la pertinence système et la pertinence utilisateur. Plusieurs mesures standards en RI peuvent être utilisées pour évaluer les performances des *SRI*. Nous détaillons dans la section suivante les mesures usuelles d'évaluation de performance des systèmes.

1.5.2 Les mesures usuelles d'évaluation des SRI

Pour évaluer les performances des différents SRI, un certain nombre de mesures standards sont proposées dans la littérature. Ces mesures permettent d'avoir une base homogène d'évaluation. Dans les sections suivantes, nous focalisons notre attention sur 5 mesures principales que nous avons utilisées dans nos travaux : le rappel, la précision, la MAP (Mean average Precision), la F-mesure, et les mesures de haute précision (P@5, P@10, P@15).

1.5.2.1 Le Rappel et la Précision

Le rappel et la précision sont deux mesures de base pour évaluer les performances des systèmes. La figure 1.2 illustre le principe de fonctionnement de ces deux mesures. Dans cette figure, les documents pertinents de la collection sont connus à l'avance et permettent de mesurer pour chaque système la quantité de documents pertinents retrouvés.

Le rappel

Cette mesure calcule la capacité du SRI à retrouver les documents pertinents de la collection. Cette mesure peut être vue comme une mesure de couverture du système.

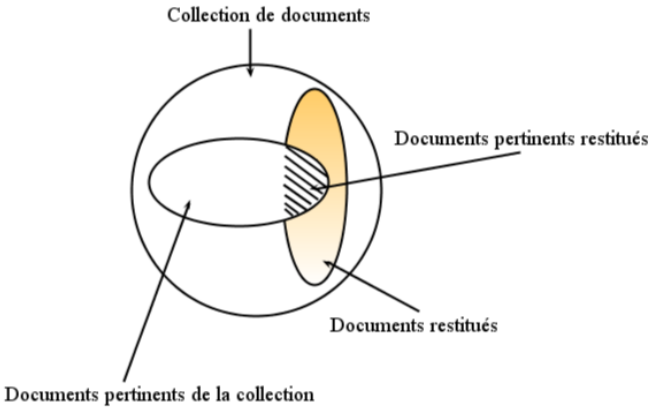

FIG. 1.2 – Rappel et précision pour une requête donnée [BYRN99]

Le rappel indique le pourcentage de documents pertinents qui ont été retrouvés par le SRI par rapport à l'ensemble des documents pertinents de la collection. Par exemple, supposons que le système restitue un ensemble de documents pertinents que l'on note P, et soit N l'ensemble des documents pertinents de la collection. Le rappel se calcule alors de la manière suivante :

$$Rappel = \frac{Card(P)}{Card(N)} \tag{1.9}$$

Dans la formule 1.9, Card(X) représente le nombre d'éléments que contient X.

La précision

Cette mesure calcule la capacité du SRI à retrouver uniquement les documents pertinents. La précision permet de mesurer la fraction des documents pertinents parmi ceux qui ont été retrouvés par le système. En supposant que P soit l'ensemble des documents pertinents restitués par le système en réponse à une requête, et que D soit l'ensemble des documents restitués par le système, la mesure de précision se traduit par la formule 1.10 :

$$Précision = \frac{Card(P)}{Card(D)}. \tag{1.10}$$

Un SRI idéal est un SRI qui restitue tous les documents pertinents (rappel = 1), et tous les documents qu'il retrouve sont pertinents (précision =1) pour la requête de l'utilisateur. Cependant, le rappel et la précision sont deux mesures qui varient généralement en sens inverse. Le calcul de la précision est relativement facile. En effet, si un jugement de pertinence[7] est disponible pour tous les documents que le système

[7]le jugement de pertinence correspond à la liste des documents pertinents pour une requête donnée.

restitue, il est alors facile de déterminer la valeur de la précision du système. Il faut cependant que le nombre de documents retrouvés ne soit pas très grand. Dans le cas où un très grand nombre de documents est retrouvé, une technique consiste à choisir dans la liste retournée les x premiers documents pour l'évaluation. La mesure de rappel est quant à elle un peu plus délicate car son calcul nécessite la connaissance du nombre exact de documents pertinents dans la collection, donc nécessite un traitement de toute la collection [Sar95]. Dans le cas de très grandes collections, le calcul du rappel réel est impossible. Dans ce cas, une solution consiste à calculer la mesure de rappel après qu'un certain nombre de documents aient été retrouvés.

Prenons un exemple pour décrire le fonctionnement de ces deux mesures que nous venons de présenter. Généralement, les documents restitués sont ordonnés par pertinence (calculée par le système) décroissante. Il est alors possible d'examiner les documents restitués par ordre de pertinence.

1.5.2.2 La courbe Rappel/Précision

Soit une requête Q, et soit N = (d_3, d_8, d_{12}, d_{11}, d_1, d_{23}, d_{210}, d_2, d_{13}, d_7) l'ensemble des documents que l'on sait pertinents pour la requête Q.

Soit S un SRI qui retourne les documents du tableau 1.1 en réponse à la requête Q.

[1] $d_{12}*$	[6] d_3*	[11] d_{88}	[16] d_{31}
[2] d_1*	[7] d_{33}	[12] d_{77}	[17] d_{72}
[3] d_{17}	[8] $d_{11}*$	[13] d_7*	[18] $d_{23}*$
[4] d_8*	[9] d_5	[14] $d_{210}*$	[19] d_4
[5] d_{15}	[10] d_2*	[15] d_{18}	[20] $d_{13}*$

TAB. 1.1 – Liste des documents restitués par un SRI pour la requête Q

Dans le tableau 1.1, les documents sont ordonnés par pertinence décroissante. Les chiffres entre crochets représentent le rang du document dans la liste restituée. Les documents suivis du symbole "*" correspondent aux documents pertinents restitués par le système. Le premier document de la liste (document d_{12}) est pertinent. On en déduit que d_{12} correspond à un taux de rappel de 10% (d'après la formule 1.9) puisque seulement un document pertinent sur 10 a été retrouvé à ce moment.

Lorsque le système S retrouve le document d_{12} au rang 1, il obtient une précision de 100% au taux de rappel de 10%.

Les listes de documents restituées par les SRI sont en général ordonnées par degré de pertinence système. Il est alors possible d'examiner les listes en partant du document ayant obtenu le meilleur score. Les taux de rappel et précision varient dans ce cas en fonction des documents analysés. En général, onze points de rappel sont considérés en RI : (0%, 10%, 20%, ..., 100%). À chaque nouveau document analysé, on calcule alors le taux de rappel réel correspondant (R), et le processus est stoppé lorsque R est inférieur ou égal au point de rappel choisi R_i, soit ($R_{i-1} < R <= R_i$). Le taux de précision P(R)

est calculé à chacun de ces points de rappel.

En reprenant l'exemple présenté dans le tableau 1.1, le prochain document pertinent que S restitue (après le document d_{12}) est le document d_1 et il se situe au rang 2. Le système obtient donc un taux de précision de 100% (2 documents pertinents sur 2 documents retournés) au taux de rappel de 20% (2 documents pertinents retrouvés sur l'ensemble des 10 documents pertinents pour la requête). Les valeurs de rappel et précision ainsi calculées sont représentées à l'aide d'une courbe rappel/précision aux 11 points de rappel standards. La figure 1.3 illustre cette courbe.

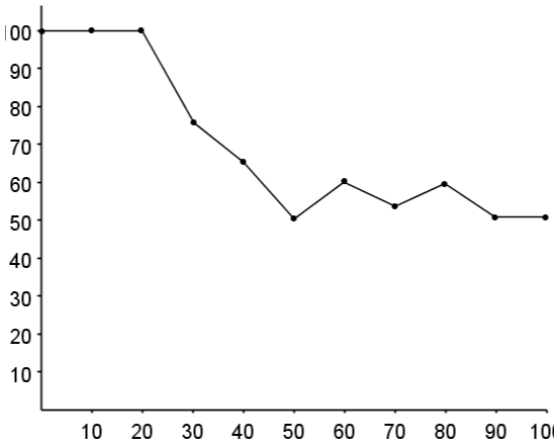

FIG. 1.3 – Précision aux 11 points standards de rappel

L'interpolation L'interpolation est une technique qui est souvent utilisée lorsque l'on souhaite calculer par exemple la précision à des points différents des 11 points de rappel standards tels que nous l'avons présenté précédemment, car ces points de rappel standards ne peuvent pas toujours être atteints de manière exacte.

Modifions l'exemple que nous avons présenté précédemment pour le calcul de la précision aux 11 points de rappel standard. Supposons que la liste des documents pertinents pour la requête Q corresponde à N $=(d_3, d_8, d_{11})$. Les documents restitués par le système sont les mêmes que ceux présentés dans le tableau 1.1. Dans cet exemple, la précision est calculée aux points de rappel 33,3% (1 document pertinent retrouvé sur l'ensemble des 3 documents pertinents pour la requête), 66,6% (2 documents pertinents retrouvés sur l'ensemble des 3 documents pertinents), et 100% (tous les 3 documents pertinents sont retrouvés).

La précision moyenne L'évaluation de la performance des SRI est généralement effectuée sur un ensemble de requêtes (|Q|). Dans ce cas, une valeur moyenne des valeurs de précision peut être calculée.

$$Prec_{moy}(R_i) = \sum_{k=1}^{|Q|} \frac{Prec_k(R_i)}{|Q|}. \tag{1.11}$$

Dans la formule 1.11, $Prec_{moy}(R_i)$ représente la mesure de précision moyenne au point de rappel R_i, |Q| représente le nombre de requêtes, et $Prec_k(R_i)$ correspond à la précision au point de rappel R_i pour la requête numéro k. Une interpolation peut être utilisée pour permettre le calcul du taux de précision moyen au point de rappel R_i. L'interpolation se fait en utilisant la formule suivante :

$$P(R_i) = max_{\ R_{i-1}<R<=R_i}\ P(R). \tag{1.12}$$

La règle de l'interpolation présentée dans l'équation 1.12 peut être résumée de la manière suivante : *la précision interpolée pour un niveau de rappel i est la précision maximale obtenue pour un rappel compris entre le niveau i-1 et i.* Dans notre exemple, la précision interpolée au point de rappel 30% sera égale à la précision réelle au point de rappel 33,3% soit 25%.

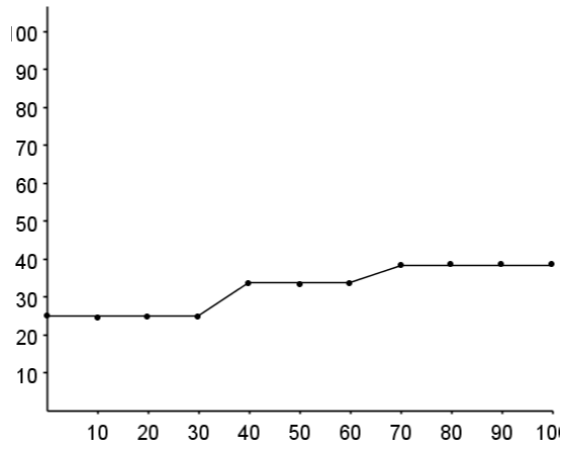

FIG. 1.4 – Précision interpolée aux 11 points standards de rappel

Dans la figure 1.4, on obtient la courbe rappel/précision par interpolation des points de rappel standards. Ainsi, les points de rappel à 0%, 10%, 20%, et 30% obtiennent une valeur de précision égale à 25%. De même, la précision à 66,6% de rappel réel (lorsque 2 documents pertinents sont retrouvés) est interpolée aux points de rappel 40%,50%, et 60%, soit 33,3%. Pour finir, la précision à 100% de rappel réel (les 3 documents pertinents sont retrouvés) est interpolée pour les points de rappel de 70% à 100%, soit 37,5%.

1.5.2.3 La moyenne des précisions moyennes (MAP)

Cette mesure calcule la moyenne des valeurs de précision moyenne non interpolées sur l'ensemble des documents pertinents. Le calcul de la précision moyenne (pour une requête Q donnée) se fait en déterminant à chaque document pertinent retrouvé la précision correspondante. Lorsqu'un document non pertinent est retrouvé, sa précision est égale à 0. La précision moyenne établit donc la moyenne de toutes ces mesures afin de déterminer la performance du système pour la requête Q. La formule 1.13 donne la méthode de calcul de la MAP :

$$MAP = \frac{1}{n} \sum_{Q_i} \frac{1}{|R_i|} \sum_{D_j \in R_i} \frac{j}{r_{ij}}. \tag{1.13}$$

Dans la formule 1.13, r_{ij} correspond au rang du *j-ième* document pertinent pour la requête Q_i. $|R_i|$ indique le nombre de documents pertinents pour la requête Q_i, et n correspond au nombre de requêtes de test. L'exemple présenté dans le tableau 1.2 illustre le calcul de la MAP.

Q1	Q2	commentaires
1	4	rang du 1er document pertinent
5	8	rang du 2ème document pertinent
10		rang du 3ème document pertinent

TAB. 1.2 – Listes restituées par un système en réponse aux requêtes Q1 et Q2.

La MAP vaut alors :

$$MAP = \frac{1}{2} * [\frac{1}{3}(\frac{1}{1} + \frac{2}{5} + \frac{3}{10}) + \frac{1}{2}(\frac{1}{4} + \frac{2}{8})] = 0,4083. \tag{1.14}$$

1.5.2.4 Mesures de haute précision : P@X

Les mesures de haute précision permettent de ne pas évaluer l'ensemble des documents contenus dans la liste restituée par un SRI. En effet, ces mesures permettent d'examiner les listes à différents niveaux de coupe. On considère alors les X premiers documents, et la précision est calculée en fonction de la valeur de X. L'idée est que un système qui retourne en tête de liste un grand nombre de documents pertinents obtient une P@X supérieure à un autre système pour lequel les documents pertinents sont dispersés dans la liste restituée. Les valeurs de X peuvent être fixées à 5, 10, 15, 30, ou 100 documents par exemple. Nous parlons ici de mesures de haute précision car on évalue la capacité du système à retrouver un maximum de documents pertinents pour un faible nombre de documents retournés. Cette situation est très courante dans le cas où les utilisateurs recherchent dans les premiers documents l'information pertinente. Si la valeur de X est plus grande que le nombre total de documents retrouvés, les documents manquants sont considérés non pertinents. La valeur de haute précision correspondante est alors diminuée. Par exemple, un système qui restitue 2 documents tous pertinents aura une valeur de P@5 égale à 2/5, même si seulement 2 documents

sont retrouvés. Dans nos travaux, nous nous intéressons à la haute précision lorsque 5, 10, ou 15 documents sont retournés.

1.5.2.5 La R-Précision

Elle mesure la précision (ou le rappel) après que R documents aient été restitués pour la requête. R représente le nombre total de documents pertinents pour la requête. Cette mesure compense les limites des mesures de haute précision quand la précision est calculée pour x documents et que le nombre total de documents |P| est inférieur à x. Si la valeur de R est plus grande que le nombre total de documents retrouvés, tous les documents non retrouvés sont alors considérés non pertinents.

1.5.2.6 La F-mesure

D'autres mesures combinant le rappel et la précision sont utilisées en RI pour évaluer les performances des systèmes. Nous présentons ici la F-mesure. [SBH97] définit la mesure harmonique qui combine les mesures de rappel et de précision selon la formule 1.15

$$F - mesure = \frac{2}{\frac{1}{Rappel} + \frac{1}{Précision}}. \tag{1.15}$$

$$F - mesure = \frac{2 * Rappel * Précision}{Rappel + Précision}. \tag{1.16}$$

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 et est obtenue quand tous les documents classés sont pertinents, et quand tous les documents pertinents ont été classés.

Les campagnes d'évaluation en RI permettent d'évaluer sur des collections différentes plusieurs SRI, afin de valider les différents modèles mis en œuvre, et comparer les systèmes. Nous présentons dans la section suivante la campagne d'évaluation TREC qui est l'une des campagnes les plus utilisées en RI. Nous donnons également un bref aperçu de son évolution au cours du temps.

1.6 Les campagnes d'évaluations : le cas de TREC

Les recherches dans le domaine de la RI se sont attachées à fournir des modèles théoriques pour formaliser et valider les différentes techniques de recherche proposées. Les objectifs de ces campagnes étaient de favoriser l'application des techniques de RI sur de grandes collections de données, de favoriser l'échange de technologie entre les industriels et la communauté scientifique, de comparer les performances des différentes techniques, et enfin de définir un protocole d'évaluation homogène pour toute la communauté de RI.

Le projet Cranfield [Cle67] est l'un des projets pilotes en RI pour l'évaluation des systèmes. Dans le projet Cranfield, les collections utilisées étaient constituées de résumés d'articles scientifiques et avaient une faible taille. Les principes d'évaluation qui ont été proposés dans ce projet sont encore utilisés de nos jours pour l'évaluation des SRI. Dans ce projet, l'évaluation des performances (efficacité) d'un système est basée sur un ensemble de requêtes et une collection de documents, ainsi que sur une liste de documents pertinents pour chacune des requêtes. L'évaluation revient donc à comparer les réponses réelles du système aux réponses idéales qui doivent être restituées en réponse à la requête. Ce même cadre d'évaluation est réutilisé dans des campagnes telles que TREC[8] qui met en compétition un ensemble de SRI. Les collections telles que celles de Cranfield sont des collections de petite taille et ne reflètent plus la variété et la quantité d'information qui circule de nos jours. Le tableau 1.3 donne un aperçu du contenu de quelques-unes de ces collections de faible taille.

Description	**Medline**	**Cranfield**	**CISI**	**CACM**
Nb. documents	1033	1398	1460	3204
Nb. termes	4315	2882	5755	3029
Nb. moyen de termes par document	46,6	49,8	38,2	18,4
Nb. requêtes	30	225	112	64
Nb. moyen de termes par requête	9,5	8,5	23,3	9,3
Nb. moyen de documents pertinents par requête	23,2	8,2	27,8	15,3

Tab. 1.3 – Exemple de collections de test de faible taille.

Les campagnes qui ont suivi le projet Cranfield ont permis l'évaluation des systèmes sur de plus larges collections couvrant différents domaines.

La campagne TREC est financée par la DARPA (Defense Advanced Research Projects Agency) et le NIST (National Institute of Standards and Technology). C'est une des campagnes de référence en RI. La campagne TREC offre d'importantes collections de test et un ensemble de tâches diverses qui évoluent d'une année à l'autre. Les expérimentations qui y sont menées sont complètes. La première campagne TREC (TREC-1) voit le jour en 1992 avec 25 participants issus du monde académique et industriel. Un certain nombre de tâches existent dans TREC et sont dédiées à certains aspects de la recherche tels que la génomique, le web, le filtrage, les blogs, les questions réponses, etc. La 17ème édition de TREC est TREC 2008 et elle comprend 5 tâches[9] dont la tâche *blog track* qui s'intéresse à l'étude des informations contenues dans la "blogosphère", la tâche *entrepise track* qui s'intéresse à l'information manipulée dans les entreprises, etc.

À chaque session, TREC met à disposition des participants à la campagne un ensemble de documents et de requêtes. Pour chacune des requêtes, l'ensemble des documents pertinents est déterminé par des juges humains. TREC met aussi à disposition des participants un programme nommé *trec-eval* qui permet de calculer, pour un ensemble de requêtes, les performances des systèmes selon plusieurs critères et mesures.

[8] http ://trec.nist.gov
[9] http ://trec.nist.gov

Les mesures que nous avons présentées dans la section précédente sont notamment des mesures de TREC. Plusieurs tâches sont traitées dans les campagnes TREC. La principale tâche est la tâche adhoc. Dans cette tâche, l'évaluation se fait en comparant les documents pertinents restitués par les divers systèmes participants pour une requête, et la liste des documents pertinents jugés par les juges de TREC pour la même requête testée, en utilisant le programme trec-eval. Le programme trec-eval calcule pour les 1000 premiers documents, les performances des listes restituées par les systèmes. Ce programme permet de calculer plus d'une centaine de mesures de performance, dont celles présentées dans la section 1.5.2.

Les collections de test et les méthodes d'évaluation proposées dans la campagne TREC font l'objet de nombreuses critiques. On peut, par exemple, souligner le fait qu'elles proposent uniquement des mesures quantitatives pour évaluer les résultats de la recherche. Une des critiques qui est faite à TREC est que la pertinence des documents est binaire (pertinent ou non pertinent) or, il est reconnu que la pertinence est plutôt une notion subjective qui dépend de l'utilisateur. Les travaux présentés dans [P.96] ont montré que les jugements de pertinence pour un même besoin, diffèrent en fonction du juge qui examine les documents ainsi que l'instant du jugement. dépendant complètement de l'utilisateur. Malgré cela, TREC reste sans conteste la référence en matière d'évaluation des SRI. Nous détaillons dans le chapitre 4 les collections de TREC ainsi que les différentes mesures que nous avons utilisées pour évaluer nos travaux.

D'autres initiatives telles que TREC ont vu le jour en Europe et au Japon. On y trouve par exemple :

- en Europe les campagnes annuelles CLEF (Cross Language Evaluation Forum) [Pet00] lancées en 2000 pour l'évaluation de systèmes de recherche d'information multilingue et INEX lancé en 2002 [10] pour l'évaluation de systèmes de recherche d'information sur des documents XML.
- Amaryllis, une version française de TREC de 1996 à 1999. Amaryllis a intégré la campagne CLEF en 2002. Il s'occupe de la tâche évaluation monolingue de collections de textes Français.
- le projet annuel NTCIR (NII-NACSIS Test Collection for IR Systems).

Nous utilisons dans nos travaux les collections de la campagne TREC, ainsi que les mesures de TREC que nous avons présentées. En effet, pour les besoins de notre étude, nous souhaitons pouvoir comparer d'une part les résultats des méthodes que nous proposons avec ceux obtenus par d'autres systèmes (en l'occurrence les systèmes participant à TREC), et d'autre part la structure et la taille des requêtes de TREC nous permet d'appliquer des traitements linguistiques aux requêtes. De plus, ces collections étaient disponibles au moment de nos expérimentations.

[10] http ://www.is.informatik.uniduisburg.de/projects/inex03/

1.7 Conclusion

Dans ce chapitre, nous avons présenté les principes de la RI, à travers les différentes étapes du processus de RI. La requête de l'utilisateur est traitée par le SRI afin de déterminer les documents les plus pertinents. Plusieurs modèles théoriques existent (modèle vectoriel et modèle probabiliste par exemple) et sont utilisés par les SRI pour effectuer la mise en correspondance entre une représentation de la requête et des documents. Cette représentation des documents et des requêtes est réalisée lors de la phase d' indexation, et un calcul de similarité permet aux SRI de restituer à l'utilisateur les documents qu'ils supposent pertinents.

Cependant, un certain nombre de problèmes résident dans le processus de RI tel que nous l'avons présenté. En effet, les SRI traitent toutes les requêtes qu'ils manipulent de la même manière, c'est à dire que les requêtes sont considérées comme un ensemble de mots clés isolés. Dans l'approche que nous proposons, les requêtes sont distinguées à travers une typologie basée sur un ensemble de caractéristiques linguistiques. Il est donc nécessaire de trouver un type de représentation autre que les mots-clés (communément appelé "sacs de mots"), afin de prendre en compte toutes les spécificités du langage naturel.

De plus, nous utilisons dans nos travaux les données issues de la campagne d'évaluation TREC. Les données de TREC sont très nombreuses et ne sont pas utilisées dans leur ensemble. Nous mettons en place dans nos travaux des techniques de fusion de données afin de prendre en compte aussi bien les spécificités des systèmes que les résultats qu'ils restituent.

Dans le chapitre suivant, nous utilisons des techniques de Traitement Automatique du Langage (TAL) pour caractériser les requêtes. Le domaine du TAL essaie d'exploiter des connaissances linguistiques pour apporter plus de compréhension dans le traitement des documents et des requêtes de l'utilisateur. Nous présentons les principaux paliers du TAL (morphologique, sémantique et syntaxique) ainsi que leur apport dans la RI.

Chapitre 2

Les traitements linguistiques en RI

2.1 Introduction

La compréhension du besoin d'information de l'utilisateur, exprimé à travers une requête, est une tâche très difficile en RI car la requête n'est qu'une expression partielle d'un besoin mental de l'utilisateur. Belkin [BOB82] parle dans ses travaux d'un état anormal de connaissances (Anomalous State of Knowledge) pour l'utilisateur, lorsque celui-ci est confronté à un manque de connaissances sur un sujet. Cet état est l'élément déclencheur de la recherche de l'utilisateur. Plus la requête de l'utilisateur est précise, plus la recherche est efficace. Mais, d'après Bates [Bat86], "la probabilité que deux personnes utilisent le même terme pour désigner la même chose équivaut à moins de 20%". Furnas quant à lui [FDD+88] stipule que "... la probabilité que deux personnes choisissent le même terme est comprise entre 7 et 18%". L'expression de la requête est pourtant garante de l'efficacité de la recherche, et a des impacts sur le déroulement de la recherche.

Une autre difficulté à laquelle font face les SRI est qu'ils ne comprennent pas le langage humain, et n'ont donc pas une connaissance parfaite du contenu (en termes de sémantique) de la collection de documents qu'ils utilisent et des requêtes qu'ils traitent. Comme le dit [Lef00] : "comprendre un texte, pour une machine, consiste à mettre en correspondance les informations sur les objets, les évènements, les faits décrits par ce texte, avec un modèle pré-établi, dont une représentation existe en machine. En fonction du degré de finesse du modèle, de la richesse des informations qu'il prend en compte, sa compréhension sera plus ou moins élaborée". Actuellement, les moteurs de recherche n'ont une compréhension que très partielle des informations qu'ils traitent.

Dans le chapitre précédent, nous avons présenté les mécanismes de RI ainsi que la manière dont les requêtes et les documents sont traités par les SRI, sans indiquer quels traitements de TAL étaient mis en jeu. Nous avons en particulier noté que les SRI considèrent généralement la requête et les documents comme un " sac de mots ", l'appariement se faisant sur la base du nombre d'éléments en commun entre la requête et le document. Les variantes des mots sont généralement prises en compte par des traitements assez systématiques et qui peuvent être considérés comme très pauvres

d'un point de vue de l'analyse du texte (comme la recherche de radicaux de Porter [Por80] par exemple). De plus, les mots sont considérés de façon isolée ce qui limite la " compréhension " du texte (requête ou document).

Ainsi, les techniques de Traitement Automatique des Langues (TAL) peuvent être considérées comme très utiles en RI, en particulier lors de l' indexation des documents ou lors de la reformulation de requête. Le TAL est une branche pluridisciplinaire qui met en oeuvre des modèles et des outils informatiques pour traiter le langage humain dans toute sa complexité, en appliquant des traitements linguistiques sur les textes. Les travaux présentés dans [JM00] donnent une description assez complète du traitement automatique des langues.

Comme nous l'avons vu, certains traitements issus du TAL sont déjà utilisés en RI ; cependant, il s'agit généralement de traitements de bas niveau au point de vue linguistique. L'utilisation de traitements complexes, en particulier lors de l' indexation n'a pas en effet montré sa supériorité, que ce soit sur de petits corpus ou dans les campagnes d'évaluation à grande échelle telles que TREC [SJ97].

Dans ce chapitre, nous nous focalisons sur les traitements linguistiques tels que utilisés en RI, dans le processus d' indexation (section 2.2) et dans celui de reformulation de requêtes (section 2.3). Nous continuons ce chapitre en présentant des travaux qui utilisent les traitements linguistiques pour caractériser les informations (section 2.4). Cette dernière section justifie l'utilisation de caractéristiques linguistiques dans la représentation de requêtes dans le cadre de systèmes de RI adaptatifs (qui adapteraient les traitements effectués) à la requête, objectif à plus long terme auquel contribue cette thèse.

2.2 Les techniques de TAL et l'indexation en RI

Les travaux présentés par [Lal96], [Bla90], et [OP97] indiquent que la faiblesse des approches courantes de RI est liée au manque de pertinence des index choisis pour représenter le contenu des documents.

Une typologie des types d' indexation proposée par Lefèvre [Lef00] et reprise dans [Pic05] présente les différents domaines de TAL utilisés lors de l' indexation (tableau 2.1) :

2.2.1 Segmentation et apport de la morphologie

La segmentation est la première étape qui intervient dans l' indexation de textes [GT94]. Il s'agit d"identifier les unités élémentaires qui composent le texte et qui potentiellement donneront lieu à des unités d' indexation. Cette identification est basée sur le choix des séparateurs pris en compte. Les choix réalisés ont un impact direct sur la qualité de l' indexation. Par exemple, Etats-Unis, I.R.I.T, 3D, brosse à dent ont tous des délimiteurs différents. En reprenant l'exemple de " Etats-Unis " , si le tiret ("-") est considéré comme délimiteur, alors le terme Etats-Unis n'est plus considéré comme

	Désignation	**Niveau Caractéristique**	**Traitements**
0	Indexation libre par fichier inversé brut	Découpage	Découpage et inversion
1	Indexation libre par fichier inversé de mots significatifs	Lexique	Élimination des mots vides, inversion
2	Indexation contrôlée par fichier inversé de mots appartenant à une liste	Lexique	Sélection des mots appartenant à une liste
3	Indexation lire par fichier inversé de radicaux ou mots associés par un radical commun	Morphologie	Élimination des mots vides, inversion, dérivation inverse (liens entre les mots ayant le même radical)
4	Indexation libre par fichier inversé de lemmes	Morphologie / Syntaxe	Élimination des mots vides, lemmatisation, inversion
5	Indexation libre par fichier inversé de syntagmes ou mots composés	Morphologie / Syntaxe	Lemmatisation, extraction de groupes nominaux présents dans le texte
6	Indexation libre par syntagmes nominaux étendus	Morphologie / Syntaxe+	Extraction des GN, dérivations par nominalisations, associations, ...
7	Indexation contrôlée par liste terminologique	Morphologie / Syntaxe++	Extraction des GN, dérivations, filtrage par liste
8	Indexation contrôlée par champs sémantiques ou liste d'autorité	Sémantique	Extraction des GN, dérivations, classement dans des champs sémantiques
9	Indexation contrôlée par thésaurus	Sémantique	Extraction des GN, dérivations, filtrage par thésaurus (liens sémantiques)
10	Indexation à rôles	Sémantique+	Idem 5, 6, 7 ou 9 + attribution de rôles aux index
11	Indexation contrôlée par domaines ou vedettes-matières	Sémantique + / Classement	Idem 8 ou 9, puis remontée au domaine ou vedette-matière (équivaut à un classement)
12	Indexation structurée	Sémantique++ / Classement	Idem 9, 10 et 11 associés : structuration des index par niveaux et relations sémantiques

TAB. 2.1 – Typologie des modes d'indexation (P. Lefèvre, 2000)

une unité mais comme deux termes séparés. La littérature scientifique dans le domaine

de la RI ne prend plus la peine de détailler les séparateurs considérés dans telle ou telle application, malgré l'impact possible des choix. L'impact est essentiellement en termes de précision puisque le choix du bon niveau de segmentation évitera de confondre "unis" dans "Etat-Unis" et dans "les époux unis" par exemple.

La prise en compte des variantes morphologiques est également un point de variabilité entre les différents moteurs d' indexation. Lors de l' indexation, les variantes morphologiques seront regroupées via un même index. La morphologie s'intéresse à la formation des mots et plus spécifiquement aux phénomènes de flexion, de dérivation et de composition. On distingue généralement différents types de morphologie. La morphologie flexionnelle [Mor06] étudie en particulier (mais pas seulement) les variantes des mots par rapport au genre et au nombre. Il est alors possible d'obtenir la forme regroupant les variantes. La morphologie dérivationnelle [DFS02] étudie la construction des mots (ceux-ci pouvant changer de catégorie grammaticale). Elle s'intéresse donc aux liens entre lemmes par exemple entre le nom "fin " et le verbe " finir ". Les travaux présentés dans [DFS02] recensent un ensemble de ressources et d'analyseurs morphologiques qui peuvent être utilisés en morphologie dérivationnelle.

En RI, l'extraction d'une forme unique à partir de variantes repose sur trois types de principes. La lemmatisation, basée essentiellement sur la morphologie flexionnelle va extraire une forme canonique, le lemme[1] : la forme infinitive pour un verbe ou la forme masculin singulier pour un adjectif. Les lemmatiseurs tels que FLEMM [Nam00] en est un exemple. La racinisation (stemming en anglais) et la troncation vont aussi regrouper des variantes, mais sur la base de leur racine commune (niveau morphologie flexionnelle et dérivationnelle). La forme commune est généralement appelé (pseudo) radical ou racine. Le principe utilisé consiste à utiliser la troncature ou une simple suppression de caractères pour réduire les affixes des termes. Les algorithmes de racinisation les plus connus ont été développés par Lovins [Lov68] et Porter [Por80]. Ces algorithmes suppriment dans un premier temps les terminaisons des termes, et recomposent dans une deuxième étape les racines obtenues, en rajoutant des terminaisons prédéfinies. L'algorithme de Porter applique ces deux traitements de manière simultanée, tandis que Lovins les applique de manière successive. Par exemple, les termes computer, computers, computing, computational sont regroupés autour de la pseudo-racine **comput** avec l'algorithme de Porter. Il y a donc une gestion de la dérivation et de la flexion.

Prendre en compte les variations morphologiques, par exemple en regroupant les formes singulier/pluriel, permet théoriquement d'améliorer la valeur du rappel lors de la recherche. En effet, les termes que les utilisateurs spécifient dans leurs requêtes peuvent être présents dans des documents pertinents, mais sous plusieurs formes différentes.

Les travaux d'évaluation de différentes méthodes dans le domaine ne s'accordent pas sur leur efficacité. Les travaux présentés dans [Har91] comparent 3 algorithmes de *racinisation* pour l'anglais : *S-stemmer*, *Lovins* et *Porter*. Les résultats obtenus avec ces 3 algorithmes ont été comparés à ceux d'un algorithme qui n'utilise pas de technique de *racinisation*. Les conclusions de ces travaux stipulent qu'aucun des 3 algorithmes n'ap-

[1] Terme dépourvu de toute forme de flexion.

porte d'amélioration notable par rapport aux performances du système n'utilisant pas de *racinisation*, le pourcentage de requêtes bénéficiant d'une *racinisation* étant le même que celui des requêtes ne bénéficiant pas du *stemming*. Dans les travaux de [Kro93], les auteurs arrivent à des conclusions différentes de celles de [Har91] en utilisant la même collection de test. Ils montrent que l'utilisation de la *racinisation* apporte des améliorations par rapport aux performances initiales, ces améliorations étant plus importantes lorsque les documents sont assez courts. [Hul96] conclut, quant à lui, que le *racinisation* apporte en général des améliorations sauf pour des requêtes longues (exemple des requêtes de TREC), à des taux de rappel faibles. D'autres expérimentations montrent que plus la langue dans laquelle sont exprimés les documents et les requêtes est morphologiquement complexe (comme par exemple le français et l'italien), plus l'analyse morphologique des termes permet d'améliorer les performances [AvdWKvB00], [CDHK01]. Les expériences dans [GGS97] pour le français montrent que l'application de la *racinisation* permet d'augmenter de 18% la précision. Dans les travaux de [JZ00], les auteurs démontrent que la normalisation morphologique des termes augmente de 30% la précision. Ces expériences montrent que certains facteurs influencent la qualité de la *racinisation*. En effet, il arrive que certains termes qui ne devraient pas être regroupés sous la même *pseudo-racine* le soient. On parle alors de *sur-racinisation* pour signifier que la *pseudo-racine* est trop large. Par exemple, la pseudo-racine *nat* regroupe à la fois les termes *nature* et *natation*, bien que ces deux termes n'aient pas de lien sémantique. Il peut aussi arriver que d'autres termes devant être regroupés ne le soient pas. C'est la *sous-racinisation*, c'est à dire que la *pseudo-racine* n'est pas assez large. Par exemple, la pseudo-racine *adaptat* ne permet pas de relier les termes *adapter* et *adaptateur*.

2.2.2 Apport de l'analyse syntaxique

L'analyse morpho-syntaxique extrait la catégorie grammaticale et morpho-syntaxique (genre, nombre) de chaque mot extrait. Ceci est une phase nécessaire à l'extraction de lemmes évoquée dans la section précédente. Par exemple, le mot porte peut désigner à la fois un nom (porte) ou un verbe (porter). L'analyse syntaxique permet de lever ce genre d'ambiguïtés qui peuvent apparaître dans les documents et les requêtes. Dans cet exemple, une analyse morpho-syntaxique permet de lever l'ambiguïté sans avoir recours à une analyse syntaxique complète. La morpho-syntaxe analyse les mots en présence et permet de déterminer la catégorie à laquelle le mot appartient. Par exemple, la présence d'un article avant " porte " permet de savoir qu'il désigne un nom et pas un verbe. En RI, les éléments de certaines catégories grammaticales sont systématiquement considérés comme mots vides (pronoms personnels par exemple), alors même qu'ils sont généralement porteur de sens (phénomène d'anaphore par exemple).

L'analyse syntaxique permet également d'identifier des groupes nominaux et leur structure, leur fonction par rapport à un verbe. Ainsi, en RI, l'analyse syntaxique permet d'extraire des syntagmes qui peuvent devenir les termes d' indexation. L'extraction de syntagmes peut permettre ainsi de gérer les groupes de mots ou les expressions. L' indexation par syntagmes est moins ambiguë que l' indexation par les mots composants considérés indépendamment.

L' indexation par des termes complexes (groupes de termes, paire adjacentes) est étudiée par plusieurs auteurs [MBSC97], [ABC+96], [SM83]. Cependant, plusieurs questions se posent quant à l'utilisation des termes complexes dans l' indexation. On peut citer le problème de manque de couverture entraînant une baisse du rappel, l'extraction des termes complexes et leur pondération.

Les termes complexes peuvent être extraits des documents par un calcul de la fréquence de co-occurrence des termes qui les composent [ALN03], ou par des approches qui combinent l'utilisation de techniques statistiques et symboliques [Cla03]. Les techniques symboliques permettent d'identifier les termes dont la structure des syntagmes est connue. Les méthodes statistiques peuvent générer des combinaisons de termes syntaxiquement incorrectes, ce qui a pour effet de réduire l'efficacité de la recherche, en terme de précision. D'autres approches [AGBS00], [JG01] utilisent la notion de patrons (exemple : NOM - NOM) basée sur une analyse syntaxique de surface pour déterminer les termes complexes. Il existe aussi d'autres approches [Dai96], [LLYM04] dites mixtes, qui combinent les aspects syntaxiques et la co-occurrence des termes pour permettre une meilleure détection des termes complexes. La principale difficulté lors de l'utilisation des termes complexes est liée à leur variabilité. D'après [Dai02], les variantes des termes complexes peuvent être syntaxiques ou typographiques. Les travaux présentés dans [GGHR00] s'intéressent à l'impact des termes complexes sur l' indexation lorsque ces derniers sont utilisés en complément des termes simples. Les résultats obtenus ne montrent pas une amélioration notable en termes de rappel précision.

La prise en compte des connaissances syntaxiques lors de l' indexation en RI se heurte à un autre type de difficulté : la pondération. Nous avons vu dans le chapitre 1 que le choix des index se base sur une pondération des termes, en fonction de leur fréquence d'apparition dans le document. Cependant, les termes complexes apparaissent bien moins fréquemment que les termes seuls dans les documents. Le problème de la pondération de ces termes complexes se pose alors car, malgré leur faible fréquence d'apparition, ils n'en demeurent pas moins importants. Les mesures couramment utilisées en RI sont basées sur la pondération des termes (TF*IDF). Comme le souligne l'étude réalisée dans [SJ95], la pondération des structures complexes a un impact sur les performances de la recherche. Dans [THH00], les auteurs ont montré que l'utilisation du facteur IDF était inadaptée pour les structures complexes. Certains auteurs ont alors proposé des mesures de pondération alternatives, basées sur les connaissances syntaxiques des termes [Had02], [PB97]. Ces mesures permettent de pondérer les index en fonction de la catégorie grammaticale des syntagmes. Les travaux présentés dans [Fag87] utilisent le poids des termes composant le terme complexe pour déterminer son poids final. Les résultats obtenus sont mitigés et ne permettent pas de juger de manière précise la pertinence de ce genre de pondération. D'autres chercheurs proposent une pondération dite syntaxique, en accordant une importance relative aux différents types de syntagmes [PB97]. Ce genre de pondération semble apporter une amélioration dans les résultats. La pondération des termes est une étape préliminaire dans le processus d' indexation. [Fox92] utilise deux représentations vectorielles pour différencier les index formés par des termes, et les index formés par des syntagmes. Les résultats obtenus par

ces deux stratégies sont alors fusionnés. Les travaux présentés dans [SLWPC99] privilégient, quant à eux, l'utilisation d'un seul index, regroupant à la fois les syntagmes et les termes simples.

2.2.3 Apport des connaissances sémantiques

Il est souvent fait cas dans la littérature de deux types d' indexation faisant appel à des ressources sémantiques [Baz05] : l' indexation sémantique et l' indexation conceptuelle. L' indexation sémantique est basée sur le sens des termes [Mih04], et utilise des techniques de désambiguisation de mots pour indexer les documents et les requêtes. Dans [MM00], les auteurs trouvent une amélioration de 16% pour le rappel et de 4% pour la précision lorsqu'ils combinent l' indexation basée sur les listes de synonymes des termes de WordNet [2] et l' indexation basée sur les mots clés. L' indexation conceptuelle quant à elle utilise des concepts issus d'ontologies pour indexer les documents [AGM04], [GMV99]. Cette approche regroupe les termes ayant des caractéristiques communes dans les documents, et considère les regroupements comme des unités de sens ou concepts. De ce fait, l' indexation conceptuelle est fortement liée à un domaine spécialisé. Les résultats obtenus par [WA98] montrent une amélioration des mesures de rappel et de précision par rapport à un SRI classique, lorsqu'une ressource de type ontologie[3] est utilisée. Dans le cas des ontologies, les travaux présentés dans [AGCS06b], [AGCS06a] s'intéressent à l'identification et à l'organisation des termes d'un domaine de connaissance à travers l'utilisation de ressources termino-ontologiques. Les travaux présentés par Chrisment et ses collègues [HCM06] s'intéressent à la mise à jour d'une ontologie à partir de l'analyse d'un corpus et de la gestion de types abstraits (concepts de haut niveau d'abstraction), afin d'améliorer l' indexation des documents dans le domaine de l'astronomie.

Les informations sémantiques utilisées pour l' indexation peuvent provenir de ressources dites internes (construites à partir des documents) [Gau06] ou externes (préexistantes). Comme ressources externes, nous pouvons citer Wordnet, EDR[4], les ontologies de domaine et les thésaurus spécialisés tels que UMLS, MeSH. Ces ressources externes sont soit génériques (par exemple WordNet) soit relatives à un domaine particulier. Un thésaurus [AG92], [Hud94] est "*une liste structurée de concepts, destinés à représenter de manière univoque le contenu des documents et des questions dans un corpus donné*" [VS86]. Les thésaurus contiennent une liste de termes vedettes utilisables comme termes d' indexation et l'ensemble des termes reliés, par une relation de synonymie, de généricité/spécificité ou une relation " est lié à ". Un thésurus peut être construit soit manuellement soit de manière automatique [Gre92], [CL92]. WordNet quant à lui est une ressource qui traite le vocabulaire Anglais. Un travail équivalent existe pour les langues Européennes ; il s'agit du projet EuroWordNet[5] qui couvre 7

[2] wordnet.princeton.edu

[3] Ontologie est le terme utilisé pour désigner "une compréhension partagée d'un domaine donné" [Gua97], [HMCE07].

[4] www.ijnet.or.jp/edr

[5] www.illc.uva.nl/EuroWordNet

langues Européennes et utilise les mêmes principes que WordNet. Le projet WordNet a été mis en place par le Cognitive Science Laboratory (Princeton University) sous la direction de George A. Miller et Christiane Fellbaum depuis 1985. Un élément important de WordNet est la notion de *synset* qui correspond à un ensemble de mots synonymes (synonym set en anglais). Des liens sémantiques sont utilisés pour relier les *synset*[6] entre eux. Un terme qui a plusieurs sens se retrouve alors dans plusieurs *synsets*. [Voo94] utilise WordNet en RI pour désambiguiser le sens des termes, en calculant la valeur de co-occurrence entre les termes et les *synset* qui y sont liés.

Outre l' indexation, les techniques de TAL peuvent être appliquées au niveau de la requête à travers l'expansion de la requête. Nous nous intéressons dans la section 2.3 à l'application des techniques de TAL pour l'expansion des requêtes.

2.3 Les techniques de TAL et la reformulation des requêtes

L'expansion des requêtes est une technique qui est utilisée pour enrichir la requête par l'ajout de nouveaux termes pertinents, reliés à ceux de la requête initiale ; elle permet de préciser la requête de l'utilisateur. L'utilisation des connaissances morphologiques pour l'expansion de la requête est obtenue de manière implicite par l'application de la *racinisation*. En effet, le remplacement des termes de la requête par des (pseudo-racines) équivaut à plusieurs requêtes similaires lorsque par exemple les termes utilisés sont des synonymes. Cette expansion implicite est liée au fait que les *pseudo-racines* regroupent différents termes sous la même forme [VR79]. Une autre approche pour l'expansion de la requête consiste à utiliser des connaissances sémantiques pour l'enrichissement de la requête. Deux stratégies peuvent être alors utilisées. Soit l'ajout des nouveaux termes est réalisé par rapport à la requête entière [QF95], soit les différents termes de la requête sont étendus individuellement [GWR99] en les enrichissant avec des termes qui co-occurrent avec eux dans la collection de documents.

L'utilisation des synonymes est également un moyen utilisé en RI pour l'expansion de la requête [Voo94]. Ces synonymes offrent la possibilité aux SRI de mettre en correspondance des termes sémantiquement proches bien que graphiquement différents. Dans une relation de synonymie, la premier terme correspond à la *forme canonique* et le second terme à la *forme synonyme* [TM06]. Cette forme canonique en général correspond à la forme la plus connue. Le phénomène de synonymie, s'il n'est pas géré, peut introduire du silence lors de la recherche d'information car, par exemple, un document parlant de *voitures* est tout aussi pertinent qu'un document parlant *d'automobiles*, lorsque la requête initiale porte sur les voitures. La principale difficulté d'une telle approche est le choix des synonymes à utiliser pour l'expansion [MM00].

Plusieurs expérimentations sur l'expansion des requêtes ont été effectuées en utilisant des ressources internes ou externes. [PKJ99] utilise des requêtes structurées, composées de conjonction de concepts, chaque concept étant représenté par une disjonction de synonymes, pour effectuer la recherche. Bien que performant, ce type de requêtes est

[6] un synset représente un noeud (ou concept) dans wordnet

rarement utilisé par les utilisateurs à cause de la complexité du langage des requêtes. [Voo94] a mené une expérimentation sur l'expansion manuelle des requêtes en utilisant des synonymes provenant de *WordNet* [7]. [CS04] utilise les liens sémantiques pour étendre les requêtes. Plusieurs auteurs ont étudié l'impact des synonymes provenant d'un *thésaurus* sur l'expansion de requêtes. Cependant, l'expansion automatique des requêtes avec des termes synonymes est problématique car, les termes de la requête sont ambigus et peuvent par conséquent entraîner le choix de mauvais synonymes dans l'expansion. Dans [XC00], les auteurs réalisent l'expansion des requêtes à partir des concepts extraits des documents restitués par le système. Cette approche permet d'obtenir des résultats intéressants, à condition que les documents utilisés pour l'extraction des concepts soient pertinents.

L'expansion des requêtes n'est pas une tâche facile. Elle permet certes d'enrichir les requêtes par l'ajout de nouveaux termes, mais tout l'intérêt d'une telle méthode réside dans la sélection des termes additionnels, ainsi que dans le choix des ressources externes qui sont utilisées.

2.4 Les caractéristiques linguistiques (CL) des requêtes

Dans la section précédente, nous avons présenté l'utilisation des techniques issues du TAL pour l' indexation de documents et la reformulation de requêtes. Ce sont les phases de la RI qui sont généralement concernées par ces traitements. Cependant, récemment, certains travaux se sont intéressés à une analyse des requêtes.

[MWH02] s'est interessé à étudier les corrélations qui peuvent exister entre des caractéristiques linguistiques des requêtes et la précision moyenne obtenue par les systèmes. Ils ont travaillé sur des requêtes de la campagne CLEF. Les caractéristiques linguistiques qu'ils ont retenues se centrent sur les aspects syntaxique et de forme des mots ; elles ont été calculées à la main. Ils ont montré que le nombre de noms propres et la précision étaient corrélés.

L'approche présentée dans [MT05] s'intéresse également à ce type de corrélation. Cependant, l'étude présentée ici est plus générale dans la mesure où plus de caractéristiques ont été étudiées, que ces caractéristiques ont été extraites automatiquement et que à la fois le rappel et la précision ont été considérés. Ils ont étudié 200 requêtes de la campagne TREC et les ont caractérisées par 13 éléments qui sont énumérés dans le tableau 2.2 et qui seront détaillés dans le chapitre 4, section 4.3.

Le tableau 2.3 présente les corrélations significatives qui existent entre les performances des systèmes et les caractéristiques linguistiques. Les nombres correspondent à la corrélation (Pearson) et à la significativité (p-valeur). Le tableau 2.3 prouve que SYNTDIST et la précision sont négativement corrélés comme SYNSETS et le rappel.

Ce résultat montre clairement que, selon les caractéristiques de requêtes, certains systèmes seront plus performants : par exemple, plus le nombre de sens d'un mot est

[7] Wordnet est une base de données lexicale pour l'anglais disponible à l'adresse http ://wordnet.princeton.edu

Caractéristiques morphologiques	
Nombre de mots	NBWORDS
Taille des mots	LENGTH
Nombre moyen de morphèmes par mot	MORPH
Nombre moyen de termes suffixés	SUFFIX
Nombre moyen de noms propres	PN
Nombre moyen d'acronymes	ACCRO
Nombre moyen de valeurs nulériques (dates, quantités, ...)	NUM
Nombre moyen de mots non reconnus	UNKNOWN
Caractéristiques syntaxiques	
Nombre moyen de conjonctions	CONJ
Nombre moyen de prépositions	PREP
Nombre moyen de pronoms personnels	PP
Profondeur syntaxique	SYNTDEPTH
Distance syntaxique moyenne entre les mots	SYNDIST
Caractéristiques sémantiques	
Nombre moyen de polysémie	SYNSET

TAB. 2.2 – Caractéristiques linguistiques des requêtes [MT05]

Campagne TREC	Variables significatives pour le rappel	Valeurs significatives pour la précision
TREC3	PREP	SUFFIX
	SYNTDEPTH	NBWORDS
	SYNSET(-0,302 ; 0,012)	CONJ
TREC5		**SYNTDIST(-0,396 ; 0,000)**
		SYNTDEPTH
TREC6	**SYNSET(-0,284 ; 0,045)**	
	PN	
TREC7	SYNSET	PN
		LENGTH
		SYNTDIST(-0,234 ; 0,047)

TAB. 2.3 – Corrélations statistiquement significatives entre caractéristiques linguistiques et performance des systèmes [MT05]

élevé, plus le rappel sera bas ; plus la requête est complexe, moins la précision sera élevée. Ce résultat préliminaire a motivé l'hypothèse suivante : il est possible de grouper des questions selon leurs caractéristiques et décider quel système doit être employé pour chaque groupe de requêtes. Cela correspond à un des éléments clés de notre thèse.

Ces caractéristiques linguistiques n'ont pas été choisies au hasard, mais chacune d'entre elle vise à étudier certains aspects des requêtes.

2.5 Conclusion

Le couplage TAL/ RI n'est pas nouveau même si les récents travaux dans le domaine des ontologies à donné un nouvel élan aux recherches. Certains traitements de bas niveau sont inclus dans tout système de RI (segmentation, prise en compte des variantes morphologiques) mais les traitements plus sophistiqués issus du TAL ont du mal à être implanté dans les systèmes commerciaux dans la mesure où ils n'ont pas montré leur supériorité en termes de performances classiquement utilisées en RI (rappel/précision) à grande échelle. L'autre frein concerne des mises en œuvre et des temps de calcul supérieurs aux traitements statistiques ou de bas niveau utilisés actuellement.

Dans ce chapitre nous avons indiqué les différents types de traitements issus du TAL (morphologique, syntaxique, sémantique) qui interviennent durant les phases d' indexation et d'expansion de requêtes, nous avons également présenté les résultats diffusés dans la littérature concernant l'évaluation de ces traitements en RI. Nous avons par exemple vu dans ce chapitre que la mise en œuvre de l'analyse morphologique lors de l' indexation a un impact sur le rappel, tandis que l'utilisation des techniques d'analyse syntaxique et sémantique permet en théorie d'améliorer la précision. Cependant, les approches d' indexation ou d'expansion de requêtes, basées sur l'utilisation de connaissances linguistiques sont tributaires du choix des termes qui sont utilisés pour l' indexation.

Nous proposons dans cette thèse une nouvelle utilisation des techniques de TAL. Il s'agit de réaliser une analyse linguistique des requêtes, afin de proposer les meilleures stratégies de recherche à utiliser. Nous avons choisi d'analyser l'impact de la combinaison de connaissances linguistiques sur les requêtes, avec des *techniques de recherches* qui sont, elles, basées sur des approches statistiques. Nous appliquons des techniques de TAL sur les requêtes afin de détecter leurs caractéristiques communes [KM07], [KMDB07a]. Nous analysons ensuite le comportement de différents SRI sur ces catégories de requêtes [KML06]. Dans notre approche, les résultats obtenus par plusieurs SRI sont combinés à l'aide de techniques de fusion[8] [KMDB07b], [HMK07], à l'issue de l'analyse linguistique des requêtes. Nous donnons plus de détails sur nos propositions dans le chapitre 4. Le chapitre 3 a pour but de présenter les différents concepts de fusion qui se rapportent à nos travaux, ainsi que leur application en RI.

[8] cf. chapitre 3

Chapitre 3

La fusion en RI

3.1 Introduction

Comme nous l'avons vu au chapitre 1, un *SRI* implémente une *technique de recherche* - TR qui est un algorithme ou encore un modèle utilisé pour calculer la similarité entre les requêtes et les documents. On peut voir le *SRI* comme un système complexe qui utilise un algorithme précis pour la recherche. La figure 3.1 présente une interaction simple en RI dans laquelle une liste de documents, ordonnés par pertinence décroissante, est restituée par une technique de recherche pour répondre à une requête.

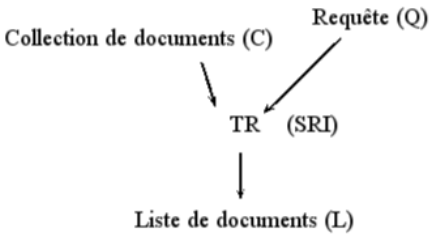

FIG. 3.1 – Schéma d'une interaction simple en RI

Dans le scénario proposé par la figure 3.1, la TR est utilisée sur une seule collection de documents pour répondre à la requête de l'utilisateur. Les interactions en RI peuvent être plus complexes que celles présentées dans la figure 3.1. Elles peuvent être regroupées en deux grandes catégories.

Dans la figure 3.2, nous présentons les 2 principaux scénarios qui peuvent avoir lieu lors du processus de recherche. Dans la figure 3.2, L_{ij} représente la liste de documents restituée par le SRI_j lorsque la collection C_i est utilisée ; L représente la liste finale qui est restituée à l'utilisateur. Dans le scénario *(a)*, une requête est soumise à 3 SRI différents qui utilisent la même collection de documents. Chacun des *SRI* restitue alors une liste de documents en réponse à la requête. Ce scénario est suivi par exemple lors des campagnes d'évaluation TREC où différents SRI rentrent en compétition et sont évalués

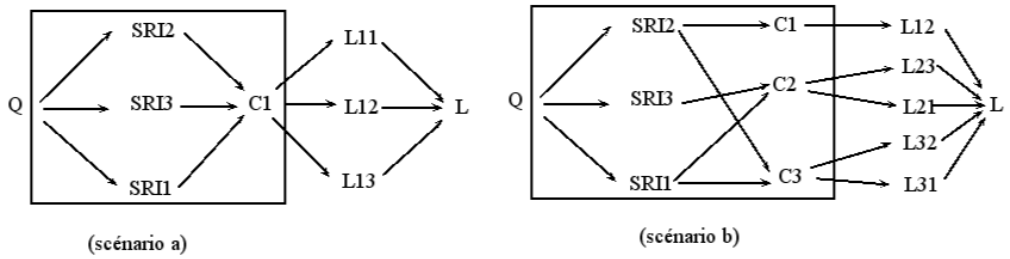

FIG. 3.2 – Interactions "complexes" en RI

sur la même base. En effet, la même collection de documents et les mêmes requêtes sont utilisés par les *SRI* participants. Dans le scénario *(b)*, 3 *SRI* interrogent chacun une ou plusieurs collections différentes pour répondre à la même requête. Ce scénario est suivi par exemple dans le cas des méta moteurs de recherche [1] qui utilisent plusieurs *SRI* différents pour répondre à une requête. La requête soumise au méta moteur est redirigée vers différents systèmes, et les différents résultats de ces systèmes (sous-listes) sont fusionnés pour former une liste finale (L) qui sera restituée.

La fusion est une technique qui est utilisée en RI pour améliorer l'efficacité de la recherche, en combinant différents systèmes ou différentes ressources d'information. Les chances d'obtenir des documents pertinents pour la requête de l'utilisateur sont alors maximisées. Nous présentons dans ce chapitre les deux principaux types de fusion utilisés en RI à savoir la *fusion de collections*, et la *fusion de données*.

La *fusion de collections* se définit comme étant une technique qui permet de combiner les sous-listes obtenues par les SRI [VGJL94] lorsqu'ils interrogent des collections de documents *différentes*. Comme nous l'avons vu dans la figure 3.2(b) par exemple, lorsque le système interroge plusieurs collections différentes pour répondre à la même requête, il obtient plusieurs sous-listes de documents. La fusion de collections consiste à regrouper les listes de documents en une seule liste qui sera restituée à l'utilisateur. Cette manipulation est transparente pour l'utilisateur pour qui le nombre de collections utilisées, ainsi que les différentes sous-listes intermédiaires sont des informations masquées ; l'utilisateur a l'impression que la recherche a été effectuée dans une seule collection [AM01]. La fusion de collections est présentée dans la section 3.2 de ce chapitre.

La *fusion de données* s'intéresse aux scénarios du type 3.2(a) où plusieurs TR sont utilisées sur la *même* collection de documents pour répondre à une requête. Les travaux présentés dans [FS94] montrent que combiner plusieurs SRI avec leurs spécificités permet d'améliorer les résultats qu'obtiennent chacun des systèmes pris individuellement. La fusion de données est présentée dans la section 3.3 de ce chapitre.

[1] Exemple :*MetaCrawler* http ://www.metacrawler.com/, *ProFusion* http ://www.profusion.com/ et *SavvySearch* http ://www.search.com/

La classification est une technique répandue en RI. Par exemple, dans les travaux présentés par [BEB99], les auteurs se sont attachés à analyser l'impact du nombre de classes sur les résultats, et montrent qu'une amélioration de la précision de 8% peut être obtenue lorsque le meilleur groupement de documents est choisi. Les techniques de fusion peuvent être combinées avec la classification. Nous terminons ce chapitre en donnant un aperçu des travaux appliquant des techniques de classification à la fusion dans la section 3.4.

3.2 La fusion de collections

Comme nous l'avons précisé en introduction de ce chapitre, la fusion de collections en RI s'intéresse au regroupement de plusieurs sous-listes de documents dans une seule liste qui sera restituée à l'utilisateur. La difficulté de la fusion de collections réside dans la détermination du nombre de documents dans chaque sous-liste afin de maximiser le nombre de documents pertinents qui seront restitués à l'utilisateur, ainsi que dans le classement des documents dans la liste finale. Dans [TVGJL95], les auteurs proposent 3 méthodes de fusion de collection, et montrent que leur méthode permet d'obtenir une augmentation de la précision de 10% comparée à la précision obtenue lorsque l'ensemble des collections est traité comme une collection unique.

La fusion de collection peut être formalisée par la définition suivante :

– Soit Q une requête,
– L_i la sous-liste restituée lorsque la collection C_i est utilisée,
– N le nombre total de documents à retrouver.

L'objectif de la fusion de collection est de déterminer les valeurs v_1, v_2, ..., v_C avec C correspondant au nombre de collections utilisées, telles que :

$$\begin{cases} \sum_{i=1}^{C} v_i = N \\ \sum_{i=1}^{C} |F_Q^{L_i}(v_i)| \ est\ maximum \end{cases} . \tag{3.1}$$

Dans l'équation 3.1, le nombre de documents pertinents restitués doit être maximisé. En pratique, la fonction $F_Q^{L_i}(v_i)$ qui correspond à la distribution des documents pertinents dans la liste restituée (c'est à dire les rangs auxquels les documents pertinents sont situés), n'est pas connue et doit être approximée.

Deux approches simples peuvent être envisagées pour déterminer le nombre de documents à choisir dans chaque sous liste.

Méthode 1 *La première méthode consiste à considérer que les sous-listes contiennent la même distribution de documents pertinents. Dans cette hypothèse, choisir le même nombre de documents dans chaque sous-liste permet de maximiser le nombre de documents pertinents qui sont restitués à l'utilisateur. Le nombre de documents à choisir dans chaque sous-liste est déterminé par l'équation 3.2 :*

$$v_i = \frac{N}{|L_i|}. \tag{3.2}$$

Dans l'équation 3.2, la somme du nombre de documents retenu pour chaque sous liste (v_i) est égale au nombre de documents à restituer (N). $/L_i/$ représente le nombre de documents à sélectionner dans la sous-liste L_i. Cette méthode n'est pas toujours satisfaisante car les différentes collections peuvent posséder des spécificités différentes, donc avoir une distribution différente des documents pertinents.

Méthode 2 *La deuxième méthode propose de sélectionner les N documents qui ont obtenu les plus grands scores de similarité dans les différentes sous-listes. Cette méthode est toutefois limitée, car elle est basée sur les scores des documents des sous-listes. Ces scores peuvent être différents en fonction des systèmes utilisés pour la recherche (différence d'échelle, unité de grandeur, etc.).*

Plusieurs approches de la *fusion de collections* ont été proposées dans la littérature. Parmi elles, les travaux présentés dans [VGJL95] s'intéressent d'une part à la détermination du nombre de documents à sélectionner dans les sous-listes, et d'autre part à la fusion de ces sous-listes. Les collections utilisées dans [VGJL95] proviennent de TREC et le système SMART [Buc85] a été utilisé.

Dans ces travaux, le nombre de documents à sélectionner (niveau de coupe) dans chaque sous-liste est déterminé à partir de la distribution des documents pertinents (RDD), obtenue pour une requête et une collection donnée. Le nombre de documents est choisi en fonction des niveaux de coupe, en calculant un score pour chaque document. Par exemple, on suppose que la valeur de N est égale à 10, et que 3 collections (A, B, et C) sont utilisées pour la recherche. On suppose aussi connu les niveaux de coupe pour les 3 sous-listes provenant des collections A, B, et C, soit N_1=3, N_2=2, et N_3=5. Soit D_{ij} un document situé au rang j dans la sous-liste i. Grâce aux différents niveaux de coupe et à la valeur de N, on associe au document D_{11} un score égal à 3/10 (le numérateur correspond au niveau de coupe, et le dénominateur à la valeur de N). On obtient respectivement un score de 1/5 pour D_{21}, et un score de 1/2 pour D_{31}. En se basant sur ce premier calcul, le document D_{31} est sélectionné en première position. Il reste alors 9 documents à sélectionner dans les 3 sous-listes. Le calcul précédent est répété et les scores des documents sont recalculés. Ainsi, le prochain document sélectionné au rang 2 est D_{32} (avec un score de 5/9). Ce processus est répété jusqu'à ce que les 10 documents soient sélectionnés.

La deuxième méthode proposée dans [VGJL95] utilise une technique de classification pour regrouper les requêtes, afin de réaliser la fusion des sous-listes. La classification des requêtes se déroule en deux étapes. Dans la première phase, les requêtes sont regroupées en fonction du nombre de documents retrouvés en commun en réponse à deux requêtes. L'hypothèse est que si pour deux requêtes un grand nombre de documents identiques est restitué, alors les deux requêtes sont similaires. Un représentant de chaque classe de requêtes ou *centroïde*, est calculé après les regroupements. Ce *centroïde* est obtenu en faisant la moyenne des vecteurs requêtes de chaque classe. La méthode de classification utilise un ensemble de requêtes d'entraînement, et une liste de documents pertinents pour chaque requête d'entraînement. Dans la deuxième phase, chaque requête de test est comparée aux centroïdes des classes de requêtes afin de déterminer la classe à laquelle

elle sera affectée. Un poids est attribué à chaque classe de requête et chaque collection. Ce poids correspond au nombre moyen de documents pertinents (parmi les k premiers documents, k étant un paramètre fixé au départ de la méthode), restitués en réponse aux requêtes de la classe, et provenant de la collection analysée. Ainsi, pour une classe de requêtes donnée, un ensemble de poids est calculé pour chaque collection utilisée.

Suite aux travaux de Voorhees [TVGJL95], Yager et ses collègues [YR98] proposent en 1998 plusieurs approches pour fusionner les listes de documents et utilisent un paramètre permettant de préciser la manière dont les documents sont sélectionnés dans la liste finale. Ce facteur est combiné à la sélection des documents dans les plus grandes collections (nombre de documents) et permet d'avoir de meilleurs résultats que ceux de [VGJL95].

Baumgarten et ses collègues [Bau97] ont aussi traité le problème de la fusion de collection, mais lorsque les collections de documents sont classées. Ils proposent une approche théorique basée sur un calcul de probabilité pour réaliser la fusion.

Dans [CLC95], les auteurs appliquent une technique de combinaison linéaire des scores afin de fusionner les sous-listes intermédiaires. Dans leur approche, l'ensemble des collections est considéré comme un document et ils utilisent le système INQUERY pour sélectionner les collections à utiliser pour la requête en cours. Les collections sont sélectionnées en fonction du nombre de documents pertinents (connus *a priori*) qu'elles contiennent. Les résultats qu'ils obtiennent ne montrent pas une amélioration sensible des performances.

La *fusion de collections* trouve dans le Web une application directe, où les sources d'information sont distribuées et sont exploitées par des méta-moteurs de recherche. Dans le cadre de nos travaux, une seule collection de documents est accessible par différents systèmes. Nous nous situons alors dans le cadre de la *fusion de données* que nous présentons dans la section suivante.

3.3 La fusion de données

Plusieurs études ont montré l'apport de la fusion de données sur les performances des *SRI* [BCCC93], [BOB82]. Contrairement à la fusion de collections que nous avons présentée précédemment [VGJL94], la *fusion de données* combine un certain nombre de facteurs tels que *l'expression des requêtes* ou les différentes TR utilisés par les SRI. Dans le cadre de la campagne *TREC*, des techniques de fusion ont été appliquées à partir de la tâche *adhoc*[2] de TREC. Appliquer la fusion de données permet de prendre en compte les spécificités des SRI lors de la recherche. À l'issue de la recherche, les documents retournés par les *SRI* peuvent être différents à cause des techniques utilisées. D'après Vogt [VC98], la fusion des données peut être soumise à trois effets différents :

[2]Dans cette tâche, la collection de documents est statique tandis que les requêtes qui sont soumises aux systèmes sont variables

1. **L'effet d'écrémage ou *Skimming Effect***
   Ce phénomène consiste à sélectionner dans chaque liste de documents restituée par les *SRI* les n premiers documents. L'écrémage se base sur le principe que les documents pertinents qui sont restitués dans les listes sont différents. Cette technique permet non seulement d'augmenter le rappel mais aussi la précision lors de la fusion. En effet, le rappel augmente à cause de la diversité des documents pertinents à travers les différentes listes, et fusionner ces listes permet en théorie d'obtenir plus de documents pertinents. La précision, quant à elle, augmente car les systèmes classent par degré de pertinence décroissante les documents et les documents en tête de liste sont tous potentiellement pertinents.
2. **L'effet de Choeur ou *Chorus Effect***
   On parle d'effet de choeur lorsqu'un ou plusieurs documents sont retrouvés par plusieurs SRI. Le choeur représente donc une sorte d'accord entre les *SRI* sur la pertinence d'un ou plusieurs documents. En prenant en compte la fréquence d'apparition des documents dans les différentes listes, ceux qui sont retrouvés par plusieurs systèmes se voient attribuer un grand score.
3. **L'effet du Cheval noir (Dark Horse Effect)**
   Cet effet tend à préférer les résultats obtenus par un SRI particulier (par exemple le meilleur SRI). On peut remarquer que contrairement à l'effet de *Choeur* qui s'intéresse à l'ensemble des *SRI*, l'effet du *cheval noir* se focalise sur un système en particulier lors de la fusion.

Ces différents effets sont intéressants à considérer quel que soit le type de fusion de données qui est appliqué.

Nous présentons dans cette section les différentes techniques de *fusion de données* appliquées aux requêtes et aux TR.

3.3.1 Combinaison des requêtes

Nous avons vu dans le chapitre 2 qu'il existe différents phénomènes linguistiques qui influent sur l'analyse du besoin d'information de l'utilisateur : un même besoin peut être exprimé de différentes manières. L'étude de la combinaison des requêtes vient de ce constat. L'idée est qu'une requête n'exprime qu'une partie du besoin d'information et que la combinaison de plusieurs expressions d'un même besoin permet d'obtenir plus de documents pertinents lors de la recherche. Les travaux présentés dans [BCCC93], utilisent les données de la campagne TREC pour réaliser la fusion des requêtes. Dans ces travaux, pour le même besoin d'information, 10 experts fournissent des requêtes booléennes qui sont combinées grâce au *SRI* INQUERY[3] [TC91] pour rechercher les documents pertinents. Les résultats qu'ils ont obtenus montrent une amélioration des performances liée à la combinaison des requêtes. [TC91] montre dans ses travaux que

[3] SRI développé à l'Université du Massachusetts par les équipes de Turtle et Croft qui utilise des réseaux d'inférence probabilistes

combiner des requêtes exprimées *en langage naturel* et *en langage booléen* permet d'obtenir de meilleurs résultats que ceux obtenus par l'utilisation de ces deux types de requêtes pris séparément. Fox et Shaw [FS94] obtiennent quant à eux des résultats qui indiquent que les performances sont meilleures lorsque les requêtes en langage naturel sont combinées avec des requêtes *booléennes étendues*. La comparaison entre ces deux travaux est toutefois biaisée car Beklin et ses collègues combinent différentes expressions d'une requête, alors que Fox et Shaw combinent les résultats obtenus par chacune des expressions de la requête. Saracevic et ses collègues [Sar95] utilisent dans leurs expériences plusieurs expressions de requêtes pour la fusion, et constatent qu'un document a d'autant plus de chances d'être pertinent qu'il apparaît dans un grand nombre de listes, en réponse aux diverses expressions de la requête. En d'autres termes, plus un document est retrouvé pour différentes expressions de la même requête, plus ce document a une grande probabilité d'être pertinent. Dans les campagnes TREC, les requêtes (*topics*) comportent 3 parties différentes : *un titre, une partie descriptive, et une partie narrative*. Ces différentes parties sont de taille variable et représentent le même besoin d'information. [ACBJ02] et ses collègues se sont intéressés à ce genre de problème à savoir l'impact de la taille des requêtes dans la fusion. Ils ont combiné les 3 parties des requêtes TREC (*titre+description*, et *titre+narratif*) et ont appliqué une normalisation de la taille des requêtes. Leur approche consiste à utiliser le score de la requête ayant la plus petite taille comme base de calcul, et un facteur est utilisé pour réduire le score des requêtes plus longues. L'équation de normalisation est donnée dans la formule 3.3 :

$$score(D_i)_{total} = score(D_i)_{small} + \sum_{j=1}^{n-1} \frac{taille(small)}{taille(Q)} * score(D_i)_j. \tag{3.3}$$

où :

- D_i est un document
- *score(D_i)* le score de similarité du document D_i
- *small* la requête ayant la plus petite taille
- *taille(Q)* le nombre de termes de la requête Q
- n le nombre total de requêtes

Leurs résultats montrent que cette normalisation permet d'obtenir des résultats de 24 à 32% meilleurs que les TR traditionnelles comme les méthodes vectorielles, qui ne sont pas adaptées lorsque la taille des requêtes est variable.

Dans le but de réduire la variabilité dans l'expression du besoin d'information, des travaux de recherche se sont attachés à étudier l'impact de la combinaison de plusieurs expressions de requêtes sur les performances des systèmes. Cependant, les résultats obtenus par les *SRI* dépendent aussi des techniques (ou stratégies de recherche) qu'ils utilisent pour identifier les documents pertinents. Dans [HMK07], les auteurs proposent une nouvelle application de la réinjection de pertinence pour réaliser la fusion. Cette méthode de fusion est basée sur les performances des SRI lorsque les 5 premiers documents ont été restitués. Les résultats montrent que des améliorations significatives sur la précision moyenne peuvent être obtenues. Dans [MT07], les auteurs fusionnent

plusieurs expressions de requêtes TREC (différentes sections des requêtes) et obtiennent une amélioration de la précision moyenne. Nous présentons dans la section suivante des méthodes de fusion combinant différentes TR.

3.3.2 Combinaison des techniques de recherche dans les *SRI*

Les stratégies ou *techniques de recherche* utilisées par les SRI sont évaluées sur la base du nombre de documents pertinents qu'ils restituent en réponse à la requête. Dans le cadre de TREC, les systèmes sont évalués par rapport aux listes de résultats qu'ils soumettent[4] qui contiennent en général pour chaque document associé à une requête un score de pertinence. Les techniques de fusion combinent les scores attribués aux documents par les différents *SRI*, afin d'augmenter l'efficacité de la recherche. Dans le cas où le score de similarité entre le document et la requête n'est pas disponible, il existe d'autres techniques qui se basent sur le rang des documents dans la fusion pour améliorer la recherche. Nous présentons dans les sections suivantes ces deux catégories de combinaison basées sur la combinaison des scores ou sur les rangs des documents.

3.3.2.1 Techniques de combinaison des scores

Fox et Shaw [FS94] ont proposé des formules basées sur une combinaison linéaire des scores des documents qui sont très fréquemment utilisées ou modifiées dans le domaine de la RI. Ils ont montré que la combinaison de plusieurs techniques de recherche augmente l'efficacité globale de la recherche. D'après leurs conclusions, la formule CombSUM appliquée la collection TREC-2 apporte des améliorations de la R-Précision de l'ordre de 13 %. Les formules de fusion proposées par les auteurs sont résumées dans le tableau 3.1 :

Nom Fonction	Définition
CombMAX	MAX (scores de similarité)
CombMin	MIN (scores de similarité)
CombSUM	SOMME (scores de similarité)
CombANZ	SOMME (scores de similarité)/Nombre scores de similarité non nuls
CombMNZ	SOMME (scores de similarité) * Nombre scores de similarité non nuls

Tab. 3.1 – Formules de combinaison de Fox et Shaw

Parmi les formules présentées dans le tableau 3.1, la formule *CombSUM* calcule la somme des scores de tous les documents retournés par les *SRI*. La formule CombMNZ prend en compte le nombre de systèmes qui ont retrouvé le même document et multiplie la valeur de CombSUM par ce nombre. Ces techniques de fusion ont été utilisées avec

[4]Ces listes de résultats sont appelées *runs*. Un run comporte un ensemble de requêtes avec pour chacune d'entre elles la liste des documents que le système a jugé pertinents, classés par pertinence décroissante. Un run correspond au résultat d'une exécution de trec_eval.

succès par [Lee97] qui montre que CombMNZ permet d'obtenir de meilleures performances que CombSUM. Dans [VC98], les auteurs proposent d'utiliser un modèle linéaire pour combiner les différents scores obtenus. Les résultats de leurs travaux montrent que cette méthode est seulement efficace lorsque le taux de chevauchement des documents pertinents est très élevé et le taux de chevauchement des documents non pertinents est faible.

Les travaux présentés dans [Lee97] indiquent que le chevauchement entre des ensembles de documents différents (nombre de documents communs entre des ensembles de documents) est également un facteur très important à considérer dans la fusion. Lee [Lee97] a proposé les formules suivantes pour calculer le taux de chevauchement existant entre des ensembles de documents :

$$R_{Overlap} = \frac{R \cap L_i}{\bigcup_{i=1}^{n}(R \cap L_i)} \tag{3.4}$$

$$NR_{Overlap} = \frac{NR \cap L_i}{\bigcup_{i=1}^{n}(NR \cap L_i)} \tag{3.5}$$

où $R_{Overlap}$ représente le taux de chevauchement des documents pertinents et $NR_{overlap}$ le taux de chevauchement des documents non pertinents, R et NR représentent respectivement la liste des documents pertinents et non pertinents. Les expérimentations de Lee ont montré que lors de la fusion, plusieurs exécutions du programme trec_eval de TREC ont tendance à restituer les mêmes documents pertinents mais des documents non pertinents différents. Pour Lee, CombMNZ est meilleur que CombSUM, et produit de bons résultats dans le cas où $R_{overlap}$ est élevé (entre 0,75 et 0,82) ,et $NR_{overlap}$ bas (entre 0,30 et 0,40). Beitzel et ses collègues [BJC+04] ont un peu contredit cette hypothèse en montrant que l'amélioration n'est pas tant liée au taux de chevauchement qu'au nombre de documents pertinents qui n'apparaissent que dans un résultat de recherche.

Nous présentons dans la section suivante les techniques de fusion de données qui se basent sur le rang des documents dans les listes restituées par les SRI.

3.3.2.2 Techniques de fusion basées sur les rangs des documents

D'après [SNC01], les travaux dans la littérature de la fusion de données ont souvent utilisé les mesures de similarité à la place des rangs des documents. L'explication qui est donnée dans [SNC01] est que les rangs des documents sont obtenus une fois que leur score de similarité est calculé. Soboroff et ses collègues [SNC01] se sont intéressés à la comparaison entre l'utilisation des scores de similarité et les rangs des documents, lors de la RI. Ils ont combiné les sous-listes des différents SRI en remplaçant les scores de similarité par une mesure qui prend en compte les rangs des documents. La formule 3.6 est utilisée pour calculer cette mesure :

$$Rank\_Sim(rang) = 1 - \frac{rang - 1}{nombre\_de\_documents\_retrouvés} \tag{3.6}$$

Les résultats obtenus montrent que l'utilisation des rangs est plus performante que l'utilisation des scores de similarité. Dans les travaux de [VGJL94], une technique simple

de fusion, basée sur les rangs des documents, a été proposée. Elle consiste à choisir les documents classés en première position dans les différentes listes retournées par les *SRI*, puis les documents classés en deuxième position, et ainsi de suite (cf. section 3.2), après avoir supprimé les doublons. Lee [Lee97] utilise le rang des documents comme alternative aux scores de similarité, et il obtient de bons résultats en termes de précision moyenne. Farah et ses collègues [FV07] proposent une technique d'agrégation des rangs prenant en compte les documents ayant le même rang dans les listes de documents. Les expérimentations qu'ils ont effectuées montrent que leur méthode permet d'obtenir de meilleurs résultats que CombSUM et CombMNZ. Une des conclusions est que la fusion doit être effectuée sur les listes restituées par les meilleurs systèmes. Lillis et ses collègues [LTP+06] utilisent une approche de fusion probabiliste basée sur les performances passées des systèmes, pour un ensemble de requêtes de test. Les résultats qu'ils obtiennent sont supérieurs à ceux obtenus avec CombSUM. Nous détaillerons cette méthode dans le chapitre 6.

D'autres techniques d'utilisation des rangs en RI existent. On peut citer par exemple la théorie du vote. La théorie du vote [Kel88], [Mou88], [Rik82] étudie les techniques à mettre en place pour choisir un représentant dans le cas d'un vote. Cette théorie peut être appliquée à la RI en considérant le processus de recherche d'information comme étant une instance du problème de vote, les documents étant perçus comme des candidats et les *SRI* comme des électeurs. L'exemple présenté dans le tableau 3.2 donne les classements effectués par 10 SRI pour 3 documents D_1, D_2, et D_3.

	1	2	3
5 :	D_3	D_1	D_2
3 :	D_2	D_3	D_1
2 :	D_3	D_2	D_1

TAB. 3.2 – Profil de votes pour 3 documents effectués par 10 SRI

Dans le tableau 3.2, la première ligne représente le rang auquel est classé le document. La première colonne donne le nombre de SRI ayant classé les documents de la même manière. L' intersection d'une ligne et d'une colonne donne le profil de vote d'un document. Par exemple, le document D_3 est classé 5 fois en première position. Dans l'exemple présenté dans le tableau 3.2, lorsque la règle de la majorité est appliquée le document D_3 est le vainqueur de l'élection, car il a été classé 7 fois premier.

Deux principaux courants existent dans la théorie du vote : le premier compare les documents (ou candidats) deux à deux, et détermine le vainqueur en appliquant un algorithme de vote *majoritaire* : c'est le cas de l'algorithme de vote *Condorcet* proposé par Montague et ses collègues [MA01]. En appliquant l'algorithme *Condorcet* à l'exemple du tableau 3.2, le document D_3 est le vainqueur *Condorcet* car il a été classé en majorité devant tous les autres documents (9 fois devant le document D_1 et 7 fois devant le document D_2). Le deuxième courant dans la théorie du vote se base sur le classement des candidats pour leur affecter un score : c'est le cas de l'algorithme *Borda count* [MA01].

Contrairement aux algorithmes basés sur le principe de la *majorité simple* qui attribue un score de *un* chaque fois qu'un document se retrouve classé en première position (*zéro* sinon), l'algorithme *Borda* affecte un score de n (correspondant au nombre de documents) au document classé en première position, *n-1* au document en deuxième position, ... Le document qui obtient le plus grand nombre de points remporte l'élection. Dans l'exemple précédent, le vainqueur est le document D_3 qui obtient un total de 27 points, contre 18 points pour le document D_2 et 15 points pour le document D_1.

Les algorithmes de vote *Condorcet* et *Borda count* sont utilisés en RI pour fusionner les documents en fonction de leur rang dans les sous-listes. Les résultats montrent que l'algorithme *Condorcet* obtient de meilleures performances que l'agorithme CombMNZ proposé par Fox et Shaw, aussi bien avec ou sans scores de similarité [MA01]. [AM01] montre aussi que *Condorcet* obtient de meilleurs résultats que l'algorithme *Borda*.

Les techniques de fusion peuvent également être couplées à d'autres méthodes statistiques telles que la classification. Nous nous intéressons dans cette thèse à l'application de la classification à la fusion que nous utilisons pour regrouper les requêtes en fonction de leurs caractéristiques linguistiques. Nous donnons dans la section 3.4 quelques applications du couplage classification/ RI et plus spécifiquement la classification des requêtes.

3.4 Application de la classification à la fusion en RI

L'utilisation des techniques de classification n'est pas nouvelle en RI. L'utilisation la plus commune consiste à classer les documents. L'intérêt de la classification des documents est de permettre une séparation entre différents éléments en fonction d'une certaine ressemblance des documents. Par exemple, les documents pertinents sont regroupés dans une même classe et les documents non pertinents dans une autre classe [Bel00], [KSYCKS01]. D'autres approches de classification essaient d'organiser les documents selon un certain nombre de critères et des travaux comme ceux présentés dans [Cro80], [vRC75] et [Voo85] montrent l'intérêt de combiner les techniques de classification avec les techniques de recherche basées sur le principe de la similarité entre documents et requêtes. D'après Rijsbergen [VR79], "*Closely associated documents tend to be relevant to the same request*", ou en d'autres termes, *les documents pertinents pour une même requête ont tendance à se regrouper*.

En RI, les techniques de classification peuvent être appliquées soit sur les documents, ou sur les requêtes ([BYRN99]). Il existe un très grand nombre de travaux sur la classification des documents et très peu sur la classification des requêtes. La classification basée sur les mots des documents a montré des résultats intéressants. Cependant, la faible taille des requêtes (en termes de nombre de mots) pose un problème (synonymie, homonymie, etc) pour la classification des requêtes (sur le web par exemple), car il est difficile de déduire le sens exact des mots. Les travaux de [WNZ02] proposent d'utiliser

les *logs*[5] des utilisateurs pour effectuer la classification des requêtes. Selon les auteurs, lorsque pour un ensemble de requêtes les *clics* des utilisateurs s'effectuent sur les mêmes documents, alors les requêtes peuvent être dans une certaine mesure considérées comme similaires.

Les travaux présentés dans [HO04b] utilisent des techniques de classification pour regrouper les requêtes suivant leurs caractéristiques statistiques. Pour chaque classe de requêtes déterminée, les auteurs proposent de détecter le meilleur système à utiliser pour répondre aux requêtes appartenant à un même groupe. Le contexte d'évaluation de la méthode proposée est TREC. Dans ces travaux, les auteurs essaient d'améliorer l'efficacité de la recherche en établissant des classes de requêtes basées sur des caractéristiques *statistiques* des requêtes. Ils proposent d'utiliser 3 critères statistiques pour la classification des requêtes :

- La longueur de la requête, déterminée à partir du nombre de mots-clés qui composent la requête [HO03b].
- La distribution de la quantité d'informations des termes de la requête. Dans leur approche, les auteurs considèrent que ce critère équivaut au rapport entre l'*idf* (cf. chapitre 1, section 1.3.2) minimum et l'*idf* maximum des termes de la requête.
- Le score de clarté (ou d'ambiguïté) de la requête [CTZC02], [VPOAR03]. Ce score est calculé par la formule

$$\omega = -\frac{\log \frac{n_Q}{N}}{\log N} \tag{3.7}$$

Où N correspond au nombre de documents dans la collection, n_Q correspond au nombre de documents qui contiennent au moins un terme de la requête. Plus n_Q est petit, plus ω est grand, ce qui sigifie que la requête est très spécifique.

Leurs résultats montrent que leur méthode permet d'obtenir de meilleurs résultats (en termes de rappel/précision) que le meilleur système. Dans nos travaux, nous appliquons la fusion sur des classes de requêtes construites à partir des caractéristiques linguistiques (CL) des requêtes. Nous avons utilisé une technique de classification supervisée (*CAH*[6]) et non supervisée (*k-means*). L'annexe B décrit ces deux techniques de classification.

3.5 Conclusion

Nous avons vu que les techniques de fusion sont utilisées pour essayer d'améliorer les performances de la recherche [HMK07], [Cro00]. Deux grands courants se sont imposés dans la fusion : la fusion de collections, et la fusion de données. La première approche combine les résultats de la recherche provenant de plusieurs collections de documents dans le but de retourner à l'utilisateur une seule liste de documents, en réponse à sa requête. Cependant, ce genre de combinaison masque un certain nombre de variabilités pouvant apparaître au niveau des techniques utilisées par les systèmes, pour identifier

[5] Les logs constituent l'historique des pages visitées par les utilisateurs

[6] CAH pour Classification Ascendante Hiérarchique. Voir [LMP06] pour plus de détails

les documents les plus pertinents pour la requête. Plusieurs approches ont donc tenté d'appliquer les techniques de combinaison aux requêtes et aux techniques utilisées par les *SRI* (parseurs, stemming, pondération, etc). Les résultats obtenus sont satisfaisants mais il existe peu d'explication sur les résultats obtenus. [BJC+03] propose un environnement unifié pour appliquer la fusion de données. Dans leurs travaux les auteurs essaient d'expliquer l'impact des techniques de fusion sur la recherche en utilisant les mêmes techniques de recherche (par exemple le même parseur, les mêmes mots-vides, etc) avec différentes techniques de fusion. Des approches de classification des documents et des requêtes ont été appliquées en RI pour réorganiser les documents qui seront restitués à l'utilisateur [BYH07]. Les techniques de classification en RI sont basées sur le principe de regroupement stipulé par [VR79]. La classification est utilisée pour la fusion car elle a montré des intérêts à être utilisée en RI, notamment à travers l'enrichissement de la requête ou des documents restitués, et la réorganisation des documents par ressemblance. Nous pouvons dire en conclusion que, malgré les différentes techniques de fusion qui ont été proposées dans la littérature, et le peu d'explication sur les conditions sous lesquelles la fusion s'avère efficace, les variations entre les performances des systèmes restent un problème à résoudre car aucun système ne peut être le "meilleur" dans toutes les circonstances.

Dans nos travaux, nous utilisons les techniques de classification non pas pour regrouper les documents par ressemblance comme dans la littérature, mais pour classifier les requêtes en fonction d'un certain nombre de critères linguistiques. La linguistique est utilisée en général en RI soit lors de la phase d' indexation, soit lors de la formulation ou de l'expansion de la requête. Nous essayons d'une part d'analyser les caractéristiques linguistiques qui influent sur les performances des systèmes (grâce à des techniques de TAL), puis nous utilisons des techniques de fusion pour choisir pour chaque classe de requête, la(les) meilleure(s) stratégie(s) à appliquer. L'originalité de ce travail réside non seulement dans la manière dont les requêtes sont catégorisées, mais aussi par les méthodes de fusion que nous proposons pour répondre au mieux aux différentes classes de requêtes. La classification des requêtes est faite à travers une phase d'apprentissage pendant laquelle les classes sont déterminées, et lors d'une phase de test nous affectons de manière automatique une requête à une classe. Nous combinons finalement différents SRI pour prendre en compte la variabilité de leur comportement, notamment la variabilité de leurs résultats, et augmenter l'efficacité de la recherche. Nous proposons donc de modéliser dans un processus adaptatif ces différentes étapes combinant des aspects linguistiques de la requête, une classification des requêtes, un processus d'entraînement/test, et des techniques de fusion de données. Le chapitre suivant présente la problématique que nous avons abordée lors de la thèse et décrit notre contribution de manière détaillée.

Deuxième partie

Contributions

Problématique

Nous avons présenté dans la première partie de cette thèse un état de l'art des travaux se rapportant à la RI, ainsi qu'à l'application de la linguistique et de la fusion en RI.

Les SRI essaient de répondre au mieux à une requête qui leur est soumise, en mettant en œuvre des mécanismes de recherche s'appuyant sur une collection de documents contenant potentiellement l'information recherchée. Un des reproches qui est fait à ces SRI traditionnels est qu'ils mettent en œuvre en majorité des techniques statistiques pour analyser les requêtes et les collections de documents. Ainsi par exemple, la requête est considérée comme une succession de mots-clés isolés traduisant le besoin d'information de l'utilisateur. Cette représentation du besoin est limitée car, en réalité, les mots de la requête ne sont pas totalement indépendants les uns des autres. La nécessité d'utiliser des techniques de TAL vient de ce constat, et peut permettre de pallier les limites des techniques traditionnelles à prendre en compte les spécificités de la langue dans laquelle sont exprimés la requête et les documents.

Nous avons présenté dans le deuxième chapitre trois aspects linguistiques (morphologie, syntaxe, et sémantique) ainsi que leur utilisation courante en RI, c'est-à dire au niveau de l'expansion des requêtes et de l' indexation des documents. Plusieurs travaux s'intéressent à l'application de traitements linguistiques en RI afin d'améliorer la performance de la recherche. Par exemple, les travaux présentés dans [Kar99] appliquent des traitements linguistiques pour déterminer les caractéristiques linguistiques des documents dans le but d'analyser l'impact de telles caractéristiques sur la sélection des documents pertinents. D'autres travaux se sont attachés à prédire la difficulté des requêtes en fonctions de leurs caractéristiques linguistiques. Amati et ses collègues [ACR04] prédisent la difficulté de la requête, et améliorent la performance de la recherche en appliquant des techniques d'expansion sur les requêtes faciles. He et Ounis [HO04a] utilisent plusieurs méthodes pour prédire la difficulté des requêtes, notamment une basée sur l'écart type de l'*idf* des termes de la requête. Cette méthode suppose que les termes d'une requête difficile possèdent des valeurs d'*idf* identiques. Dans [MWH02], les auteurs essaient de déterminer des corrélations entre les caractéristiques linguistiques des requêtes et les performances des systèmes. Les résultats qu'ils obtiennent montrent une corrélation de 0,4 entre la quantité de noms propres contenus dans la requête et la précision moyenne obtenue par les .

Dans nos travaux, nous appliquons des traitements linguistiques aux requêtes mais dans un but différent des travaux présentés précédemment. En effet, notre objectif n'est ni de prédire la difficulté des requêtes, ni de classifier les documents en fonction d'un certain nombre de traits linguistiques. Notre hypothèse de départ est liée à la variabilité des résultats obtenus par différents SRI [BH04]. Elle suppose que les SRI obtiennent des performances différentes en fonction des spécificités linguistiques des requêtes. Comme dans [HO04b], [HO03a], nous supposons que si nous sommes capables d'extraire de classifier les requêtes en fonction de certaines de leurs caractéristiques, alors nous pouvons déterminer pour chacune des classes la meilleure stratégie à utiliser. Notre

méthode diffère de leur méthode dans le choix et le nombre des critères de classification des requêtes. En effet, nous ne considérons pas les requêtes comme un ensemble de mots isolés pour lesquelles des caractéristiques statistiques sont extraites, mais nous caractérisons les requêtes par un ensemble de caractéristiques linguistiques (CL). Nous nous inspirons pour cela des travaux présentés dans [MT05] en utilisant les CL qui ont été proposées. Le chapitre 4 donne une description plus complète de l'ensemble des CL que nous avons utilisées dans nos travaux.

Notre choix s'est porté sur les requêtes pour permettre une indépendance avec les collections de documents utilisées sur lesquelles l'application de traitements linguistiques est plus longue.

Nous nous situons dans le contexte de la campagne d'évaluation TREC qui met à disposition un ensemble de requêtes dont le format est adapté à l'application de traitements linguistiques. Comme le dit [SLWPC99], les techniques de TAL nécessitent des requêtes longues. Le contexte de TREC nous permet d'appliquer des traitements de TAL aux requêtes, afin de déterminer leurs CL.

Dans le troisième chapitre de cette thèse, nous avons introduit les techniques de fusion ainsi que leur utilisation en RI. L'utilité de telles techniques réside dans le fait qu'elles permettent de prendre en compte les variabilités de différents systèmes. L'objectif des travaux que nous présentons dans ce chapitre est de proposer des méthodes de fusion pour répondre à ce problème de variabilité. Dans cette perspective, nous utilisons différents SRI comme ressources, et nous leur appliquons des techniques de fusion de données pour répondre à la requête de l'utilisateur [KML06]. Nous proposons dans nos travaux deux types de fusion. Dans le premier cas, nous mettons en œuvre des techniques simples de fusion qui ne prennent pas en compte le profil linguistique des requêtes. Un des avantages des méthodes de fusion que nous proposons est leur facilité de mise en œuvre. Nous utilisons pour ce faire deux opérateurs issus de la théorie des ensembles : l' union et l' intersection. Le deuxième type de fusion que nous proposons [KMDB07a] s'adapte en fonction du contexte de la requête (caractéristiques linguistiques) en mettant en place un couplage classification de requêtes et fusion de données. Ce couplage constitue une nouvelle approche en RI, et utilise des techniques de fusion inspirées de la technique de fusion probabiliste proposée par [LTP+06].

Notre contribution est présentée à travers 4 chapitres.

Dans le chapitre 4, nous procédons à une analyse des différentes collections que nous utilisons dans nos expérimentations. Ces collections concernent aussi bien les documents que nous exploitons (TREC), les caractéristiques linguistiques des requêtes ainsi que leur mode d'extraction. Nous présentons ensuite la méthodologie que nous avons suivie pour classifier les requêtes. Nous terminons en présentant les résultats des expérimentations que nous avons effectuées.

À l'issue de l'analyse des collections dont nous disposons dans nos travaux, nous abordons, dans le chapitre 5, deux techniques de fusion de données : la fusion systématique, et la fusion basée sur des critères de chevauchement des documents. Les

expérimentations que nous présentons valident nos propositions.

Dans le chapitre 6, nous donnons le détail de la fusion adaptative que nous proposons. Cette fusion est basée sur les classes de requêtes définies à partir de leur profil linguistique. Nous présentons ensuite 2 algorithmes de fusion probabiliste que nous nommons MaxProb et MaxProbSeg, ainsi que les expérimentations que nous avons menées.

Nous terminons la présentation de nos contributions en donnant, dans le chapitre 7, le couplage analyse canonique/RI que nous avons mis en place. Il s'agit d'un nouvel axe de recherche que nous abordons à travers une collaboration avec des collègues mathématiciens.

Chapitre 4

Analyse des collections de données

4.1 Introduction

Notre contribution se base sur un ensemble de données qui se répartissent en 3 grandes catégories : les requêtes, les SRI, et les documents. Comme nous l'avions présenté dans le chapitre 1, le triptyque *<requêtes (Quoi)/ SRI (Comment)/ documents (Où)>* constitue une vue simplifiée de la RI. Les données que nous avons utilisées pour nos expérimentations proviennent toutes de la campagne d'évaluation TREC. Dans la campagne TREC, nous disposons d'un ensemble de requêtes et de documents, de la liste des documents pertinents pour une requête donnée (jugement de pertinence), ainsi que des résultats obtenus par les différents SRI ayant participé aux tâches proposées par TREC.

Les données dont nous disposons sont nombreuses et variées. Dans la première partie de ce chapitre nous passons en revue les collections que nous avons utilisées dans nos expérimentations. Dans la section 4.2, nous donnons quelques précisions sur la campagne d'évaluation TREC qui contient les collections d'évaluation que nous avons utilisées. Dans la section 4.3, nous présentons les différentes caractéristiques linguistiques de requêtes que nous utilisons ainsi que les outils utilisés pour extraire ces informations. Nous terminons ce chapitre en présentant les résultats d'une première analyse que nous avons effectuée sur ces collections de données. En effet, dans la section 4.4, nous montrons la validité de la classification des requêtes par leurs caractéristiques linguistiques en détectant pour chacune des classes le meilleur système à utiliser. Notre hypothèse est que les systèmes obtiennent des performances différentes en fonction des spécificités linguistiques des requêtes. Les résultats sont alors comparés à ceux obtenus par les meilleurs systèmes.

4.2 Les collections de TREC

Plusieurs tâches sont définies dans TREC. On peut citer par exemple les tâches de question-réponse dans lesquelles les systèmes doivent retourner à l'utilisateur non plus

des documents susceptibles de contenir la réponse mais la réponse à une question précise, la tâche RI translingue qui s'intéresse à la recherche de documents dans une langue différente de celle de la requête, la tâche Terabyte qui s'intéresse aux très grands corpus de documents, etc. Les collections de TREC sont composées d'un ensemble important de documents, d'un ensemble de *topics* correspondant aux besoins d'information de l'utilisateur. Dans les collections de TREC, un ensemble de jugements de pertinence est associé aux requêtes TREC. Les jugements de pertinence correspondent pour chaque *topic* à l'ensemble des documents qui sont jugés pertinents par des juges humains. Les jugements de pertinence permettent de comparer les résultats *réels* des systèmes aux résultats *théoriques* établis par les juges. TREC a pour objectif :

- d'encourager les travaux de recherche en informatique documentaire permettant l'accès à des bases volumineuses en fournissant : une base de test importante, des procédures d'évaluation uniformes, un forum pour les organismes intéressés par une comparaison de leurs résultats,
- d'accroître la communication entre l'industrie, l'université et les instances gouvernementales en créant un forum ouvert aux échanges d'idées en matière de recherche.

Nous nous sommes intéressés dans nos travaux aux collections *adhoc* et *détecction de la nouveauté* de TREC car nous y avions accès du fait de la participation de notre équipe à ces campagnes TREC. Nous présentons dans les sections 4.2.1 et 4.2.2 les caractéristiques des collections pour les tâches *détection de la nouveauté* et la tâche *adhoc*, auxquelles nous avons consacré nos expérimentations.

4.2.1 La tâche *détection de la nouveauté*

La tâche *détection de la nouveauté* a été introduite en 2002 lors de la campagne TREC-11. Étant donné une requête et une liste ordonnée de documents pertinents, l'objectif de cette tâche est de retrouver les passages (phrases) pertinents et nouveaux répondant à la requête. Nous avons utilisé dans nos travaux les documents issus de la première sous-tâche qui consiste à détecter les passages pertinents dans les documents. La collection de départ est constituée de 50 requêtes provenant des campagnes TREC6, TREC7, et TREC8. Ces 50 requêtes ont été sélectionnées parmi celles pour lesquelles les jugements de pertinence comprenaient entre 10 et 70 documents pertinents. En 2002, TREC a choisi de sélectionner 50 requêtes issues des besoins d'information identifiés par les numéros 300 à 450. Le NIST (National Institute of Standards and Technology) a sélectionné les documents effectivement pertinents pour chacun de ces besoins d'information (jugements de pertinence), avec un maximum de 25 documents par besoin, et les a fournis aux participants. Un exemple de requête est décrit dans le tableau 4.1.

Une requête est composée d'un certain nombre d'élements :

- un numéro qui identifie la requête,
- un titre (T) qui donne une description du sujet de la requête en quelques mots,
- une description (D) de la requête exprimée en général à travers une phrase,

Topic : 185
Title : Reform of the U.S. Welfare System
Descriptive : document will report on reform of welfare system in the U.S. at the federal, state or local levels of government.
Narrative : A relevant document will report on the actual or proposed overhaul of the welfare system at all levels of government. Some examples of reforms to the welfare system are work, education and training programs aimed primarily at welfare recipients.

TAB. 4.1 – Exemple de requête : la requête 185

– une partie narrative (N) qui explique plus en détail le type de documents pertinents et non pertinents qui est recherché ou non. La partie narrative est exprimée en général dans un paragraphe.

Dans une seconde étape, des juges indiquent quelles phrases de ces documents sont effectivement pertinentes. Le même principe a été utilisé en 2003, avec 50 besoins. Les caractéristiques de ces deux collections sont fournies dans le tableau 4.2.

	NIST-2002	NIST-2003
Nombre de besoins d'information	49	50
Nombre moyen de documents pertinents par besoin	22,3	25
Nombre moyen de phrases issues des documents par besoin	1321	796,4
Nombre moyen de phrases pertinentes par besoin	27,9	311,14
% moyen de phrases pertinentes	2,1	39,1

TAB. 4.2 – Caractéristiques de la collection de test de TREC 2002 et 2003

Les critères d'évaluation des systèmes sont ceux définis par TREC et sont directement issus des critères communément utilisés pour évaluer les SRI : les taux de rappel et de précision. Ces deux taux évoluant en sens inverse, une mesure globale, la mesure F combinant rappel et précision permet une comparaison rapide des résultats obtenus par différents systèmes. Cette mesure fait jouer un rôle symétrique au rappel et à la précision, sans privilégier l'un ou l'autre de ces critères. Ces mesures appliquées à la détection de la pertinence de passages ont été présentées dans le chapitre 1. La F-mesure présentée dans la formule 1.16 du chapitre 1 a été utilisée.

Lorsque l'évaluation prend en compte un ensemble de requêtes, la moyenne des résultats permet de mesurer les performances globales d'un système. D'après ces caractéristiques, un système qui retrouverait toutes les phrases (rappel égal à 1) obtiendrait une précision d'environ 0,02 pour TREC 2002, et une précision de 0,39 pour TREC 2003.

Treize groupes ont participé à la tâche détection de passages pertinents de TREC 2002, correspondant à un total de 43 systèmes ou ensembles de résultats. En pratique, un

groupe utilise généralement un seul outil pour lequel il teste différents paramètres (nous les appellerons *versions de systèmes* dans le reste du manuscrit). Quatorze groupes ont participé à cette même tâche de TREC 2003, correspondant à un total de 42 systèmes.

4.2.2 La tâche *adhoc*

De TREC-1 à TREC-6, les recherches sont centrées sur deux tâches principales : la tâche de *routage* et la tâche *adhoc*. Dans la première tâche, un flot de documents doit être filtré au fur et à mesure pour répondre à un ensemble de profils fixé. À l'inverse de la tâche de *routage*, la tâche *adhoc* suppose que les collections de documents sont fixes. Nous nous intéressons dans cette section à la tâche *adhoc* de TREC.

Nous utilisons dans nos expérimentations les collections de documents issues des campagnes *adhoc* de TREC3, TREC5, TREC6, et TREC7. Les documents de ces collections sont très variés et proviennent de différentes sources. Ils sont issus en majorité d'articles de journaux tels que le Financial Times, Los Angeles Times, Wall Street Journal, ...ou de collections de documents légaux, de brevets ou de documents spécialisés dans un domaine comme l'informatique par exemple. Selon [BYRN99], la taille moyenne des documents (pour la majorité d'entre eux) est d'environ 300 mots. La taille de ces collections est variable d'une année à l'autre (plus de 3Gb dans TREC3 par exemple). Chaque année de TREC comporte un jeu de 50 requêtes qui est soumis aux participants. Un exemple de requête est donné dans le tableau 4.1. Le résultat de l'évaluation des systèmes est obtenu à l'aide du programme trec_eval. Un exemple de résultat est donné dans la figure 4.1

Les requêtes de la tâche *détection de la nouveauté* correspondent à un sous-ensemble des requêtes *adhoc* de TREC (T, D, et N). L'évaluation des performances des systèmes est faite grâce au programme *trec_ eval*. Nous avons utilisé dans nos expériences les mesures de R-précision (cf. chapitre 1, section 1.5.2.5), la MAP (cf. chapitre 1, section 1.5.2.3), et les mesures de haute précision (P@5, P@10, et P@15) (cf. chapitre 1, section 1.5.2.4). Il s'agit des mesures les plus communément utilisées lorsque l'on souhaite réaliser des comparaisons de systèmes.

4.3 Analyse des requêtes

La variabilité des requêtes est un aspect qui est étudié dans le domaine de la *RI*. Dans [BW00], les auteurs s'y sont intéressés lors de leurs expériences dans la tâche " *Requête*" de TREC-8. Ces travaux montrent que l'utilisation de requêtes courtes permet d'avoir de meilleurs résultats lorsque les mesures de haute précision sont utilisées. Certains auteurs se sont intéressés à l'utilisation des aspects linguistiques des requêtes pour caractériser leur difficulté. On peut citer par exemple les travaux présentés dans [MT05]. Dans ces travaux, les auteurs ont montré que l'étude de certains aspects linguistiques de la requête permet de prédire la difficulté de la requête. Ils ont extrait automatiquement des requêtes un ensemble de caractéristiques linguistiques et ont observé que deux caractéristiques

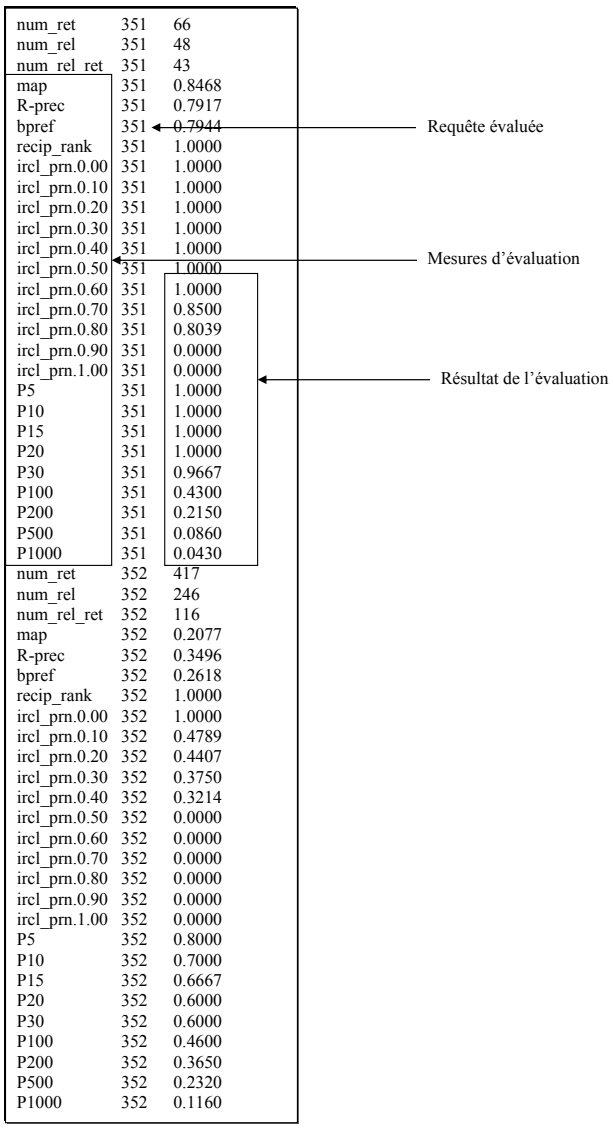

FIG. 4.1 – Résultat de trec_eval pour le système uwmt7a2 de TREC7

en particulier étaient corrélées négativement aux mesures de *rappel* et de *précision* : "*la complexité syntaxique de la requête*" pour le rappel, et le "*degré de polysémie de la requête*" pour la précision.

L'idée d'utiliser des techniques linguistiques en RI n'est pas nouvelle (cf. chapitre 2). Ces techniques sont souvent utilisées lors de la phase d' indexation des documents.

Tandis que certains auteurs [dLBEBM98] utilisent des techniques de TAL pour reformuler la requête et classifier les documents retrouvés, d'autres auteurs s'intéressent à l'analyse des facteurs qui rendent les requêtes difficiles pour les SRI [CYDP06]. Notre proposition utilise des éléments issus des travaux de [MT05] qui s'inscrivent dans le contexte de la caractérisation des requêtes difficiles. L'originalité de notre approche est de réutiliser les 13 caractéristiques linguistiques (CL) définies dans [MT05] pour effectuer une classification des requêtes. Ces critères sont extraits automatiquement. Notre idée est de ne plus percevoir les requêtes comme un ensemble de termes, mais de leur associer un certain nombre de CL permettant de les regrouper par ressemblance. Ce regroupement des requêtes permet alors d'analyser les SRI non plus par rapport à leur performance sur les requêtes prises individuellement, mais en fonction de leur performance sur des catégories linguistiques de requêtes. Pour cela, l'hypothèse de départ est que *chaque requête possède des spécificités linguistiques qui sont prises en compte à des degrés différents* par les SRI.

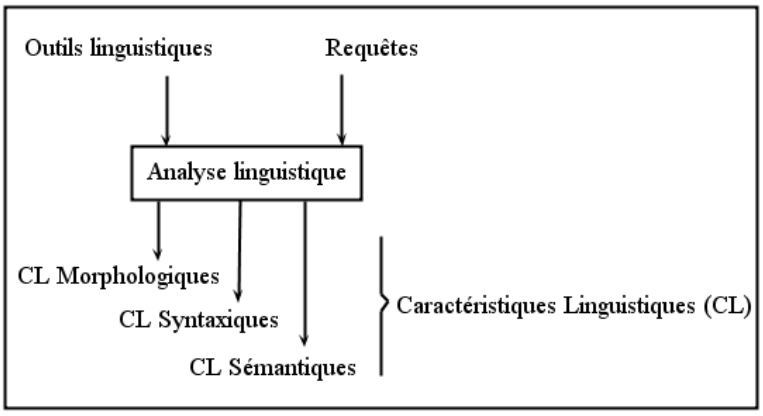

FIG. 4.2 – Analyse des variabilités des requêtes

La figure 4.2 donne un aperçu des points que nous abordons dans cette section. Dans cette figure, à l'aide d'outils de traitement linguistique (cf. sous-section 4.3.4), les requêtes sont analysées afin d'extraire leurs CL [MT05], [MT06]. Il s'agit des CL morphologiques (cf. section 4.3.1), des CL syntaxiques (cf. section 4.3.2), et des CL d'ordre sémantiques (cf. section 4.3.3). Nous présentons ensuite dans la section 4.3.4 les différents outils qui ont été utilisés pour l'extraction des CL. Dans cette section, nous illustrons le calcul des CL de la requête 185 4.3. Nous nous focalisons sur les parties *Title* et *Descriptive* de la requête.

4.3.1 Les caractéristiques morphologiques des requêtes

Nous avons vu dans le chapitre 2 que la morphologie s'intéresse à la structure des termes. Nous présentons dans cette section les différentes caractéristiques morpholo-

Title : Reform of the U.S. Welfare System **Descriptive** : document will report on reform of welfare system in the U.S. at the federal state or local levels of government.

TAB. 4.3 – Requête servant d'illustration dans le calcul des CL

giques de requêtes que nous utilisons dans nos travaux.

Nom CL	Description
NBWORDS	Nombre de mots
AVGMORPH	Taux moyen de morphèmes
PN	Taux moyen de noms propres
SUFFIX	Taux moyen de mots suffixés
ACCRO	Taux moyen d'acronymes
NUM	Taux moyen de valeurs numériques (dates, quantités, ...)
UNKNOWNS	Nombre moyen de mots inconnus

TAB. 4.4 – CL morphologiques

Le tableau 4.4 décrit 7 CL morphologiques que nous avons retenus pour caractériser les requêtes. Certains termes comme les noms propres (PN), les acronymes (ACCRO) et les valeurs numériques (NUM) sont facilement détectables dans le texte de la requête. Par exemple, si on suppose que les noms propres sont en majuscules, l'identification de tels termes est facile et ne nécessite pas de traitement linguistique compliqué. Il en est de même pour les valeurs numériques. Certains systèmes utilisent une base de données lexicale afin d'associer à chaque mot un *morphème*.

NBWORDS

Il correspond au nombre de mots de la requête. Ce nombre de mots est obtenu après une segmentation du texte par TreeTagger. Le nombre de mots de la requête est une caractéristique qui a été retenue ; ce nombre est désigné par la CL NBWORDS. L'hypothèse derrière cette CL est que les systèmes ne traitent pas de la même manière les requêtes courtes et les requêtes longues en RI, comme l'ont montrés les travaux dans [BJC+04]. Par exemple, pour la requête 185, la segmentation produit le résultat suivant :

Reform/of/the/U/S/Welfare/System/ document/will/report/on/reform/of/welfare/ system/in/the/U/S/at/the/federal/state/or/local/levels/of/government/

Le nombre de mots correspondant est égal à 28 pour la requête 185 (titre et description confondus). On constate que lors de cette segmentation U.S. a été découpé en 2 à cause du point (.) qui est considéré comme séparateur.

AVGMORPH

Il correspond au nombre moyen de *morphèmes* par mots. Cette CL peut rendre compte de la variété des aspects traités dans la requête. Elle est obtenue en utilisant la

base CELEX et TreeTagger (cf. section 4.3.4). Par exemple le mot *intolérable* est composé de 3 morphèmes (in+tolér+able). Le nombre moyen de morphèmes de la requête est obtenu en faisant la moyenne du nombre de morphèmes par mot de la requête. On peut cependant noter que l'extraction des morphèmes est fortement liée à la couverture de la base de données. La conséquence est que le nombre de morphèmes de certains mots rares, mal écrits, ou nouveaux est estimé à un.

Pour la requête 185 nous récapitulons dans le tableau ci-dessous (4.5) le nombre de morphèmes par mots de la requête.

Mot	NB. Morph	Mot	NB. Morph
Reform	2	system	1
of	1	in	1
the	1	the	1
U	1	at	1
S	1	federal	1
welfare	2	state	1
system	1	or	1
document	1	local	1
will	1	level	1
report	1	government	2
on	1	U	1
reform	2	S	1
of	1	of	1
welfare	2		

TAB. 4.5 – Nombre de morphèmes par mots de la requête 185

En faisant la moyenne du nombre de morphèmes par rapport au nombre de mots, on obtient $AVGMORPH = \frac{33}{28} = 1,1785$

PN

Il correspond au nombre de noms propres de la requête. Il est obtenu en étudiant la catégorie grammaticale des mots. Les noms propres sont analysés en tenant compte de leur casse (majuscule/minuscule).

Pour la requête 185, les noms propres sont au nombre de 2. Il s'agit de US (United States of America). Ces noms propres ont été détectés en utilisant TreeTagger (ref. section 4.3.4), en tenant compte des majuscules. Le nombre de PN vaut alors $PN = \frac{2}{28} = 0,07142$

SUFFIX

Il correspond au nombre de mots suffixés de la requête. Les suffixes les plus courants ont été extraits de la base CELEX et comparés avec les lemmes des mots de la requête. Un terme est considéré comme suffixé s'il contient au moins 3 lettres devant le suffixe. Par exemple, le terme *nation* n'est pas considéré comme étant un terme suffixé. Dans les travaux de [MT05], 56 suffixes ont été utilisés ; les suffixes retenus sont ceux qui

apparaissent au moins 50 fois dans la base celex. En nous référant qu tableau 4.6, on

Suffix	Freq.	Suffix	Freq.	Suffix	Freq.	Suffix	Freq.
able	411	ency	70	ion	837	ory	95
age	121	ent	117	ish	137	ous	229
al	448	er	1499	ism	229	out	63
ally	235	ery	98	ist	411	room	55
an	279	ess	56	ity	505	ry	56
ance	136	ful	159	ive	309	s	274
ant	146	house	69	ize	233	ship	85
ary	107	ial	82	less	208	th	94
ate	196	ian	121	line	51	up	95
ation	419	ible	69	ly	3012	ure	59
board	68	ic	457	man	181	ward	57
ed	528	ical	163	ment	257	way	71
en	79	ify	55	ness	1320	work	53
ence	195	ing	60	or	188	y	655

TAB. 4.6 – Liste des 50 suffixes utilisés

déduit que la requête 185 contient 2 mots suffixés : *federal* et *government*. Le nombre moyen de mots suffixés vaut alors $SUFFIX = \frac{2}{28} = 0,07142$.

ACCRO et NUM

Le nombre moyen d'acronymes et de termes numériques des requêtes sont obtenus en comparant les différents mots de la requête avec un ensemble de motifs prédéfinis. Pour la requête 185, il y a 2 accronymes : U.S. pour United States of America. La valeur de la CL ACCRO est alors égale à 0,07142. Aucune valeur numérique n'est présente dans la requête 185. NUM=0.

UNKNOWNS

Cette CL regroupe tous les mots qui n'ont pas été reconnus lors de l'étiquetage des mots de la requête (mots absents de TreeTagger). La CL UNKNOWNS permet d'analyser le comportement des systèmes lorsqu'un terme inconnu de la base de données qu'ils utilisent est présent dans le texte. Dans ce cas, les systèmes ne faisant pas appel à des bases de données lexicales traitent tous les mots de la même manière. Dans le cas contraire, les systèmes utilisant une base de données lexicale ne sauront pas traiter les mots inconnus.. Tous les mots de la requête 185 sont détectés. On obtient alors UNKNOWNS=0.

4.3.2 Les caractéristiques syntaxiques des requêtes

Nous avons retenu les 5 caractéristiques syntaxiques proposées par [MT05] pour nos travaux. Nous les présentons dans le tableau 4.7.

Nom CL	Description
PP	Taux moyen de pronoms personnels
CONJ	Taux moyen de conjonctions
PREP	Taux moyen de prépositions
SYNTDEPTH	Profondeur syntaxique moyenne
SYNDIST	Distance syntaxique

TAB. 4.7 – CL syntaxiques

PP, CONJ, et PREP

Ces caractéristiques sont détectées à travers l'étiquetage grammatical des mots de la requête. En RI, les pronoms personnels introduisent une certaine difficulté syntaxique lors de l' indexation des documents. Par exemple, la phrase *Le maire sera dans les locaux de l'IRIT demain. Il sera reçu par le directeur.* présente un exemple de problème introduit par les pronoms personnels. Dans l'exemple précédent, la deuxième phrase .. *Il sera reçu par le directeur* contient un pronom personnel *il*. Ce dernier fait référence à *Le maire* dans la phrase précédente. Il va donc s'en dire qu'il est important de repérer ce type d'information afin d'appliquer les traitements adaptés lors de l' indexation par exemple. Dans nos travaux, nous utilisons la CL PP pour faire référence aux pronoms personnels contenus dans le texte de la requête. Par exemple, pour la requête 185, il n'y a aucun pronom personnel. La valeur associée à PP est égale à 0. Les propositions détectées lors de l'étiquetage sont au nombre de 4. Il s'agit de *of*, *on*, *in* et *at*. La CL PREP vaut alors $PREP = \frac{2}{14} = 0,2142$. Pour terminer, il existe un verbe conjugué dans la requête 185 (*will*). La valeur associée à CONJ est égale à 0,0357.

SYNTDEPTH

La profondeur syntaxique correspond à la profondeur maximale (hauteur) de l'arbre syntaxique que l'on obtient après une analyse syntaxique de la requête avec SYNTEX. La profondeur syntaxique (SYNTDEPTH) est une mesure de complexité syntaxique en termes de hiérarchie. L'exemple 4.3 illustre comment cette CL est obtenue à partir d'une requête.

Dans la figure 4.3, la CL SYNTDEPTH a une valeur de 3 pour la requête numéro 185 de TREC-3 : "Reform of the US welfare system". Cette valeur correspond au niveau maximal d'imbrication des noeuds de l'arbre syntaxique de la requête. Nous avons illustré, pour des raisons de simplification, la partie *title* de la requête. Cette partie est aussi utilisée pour calculer la CL SYNDIST qui est présentée dans le paragraphe suivant.

SYNDIST

Contrairement à la CL SYNTDEPTH qui est calculée à partir d'une analyse verticale de l'arbre syntaxique de la requête, la CL SYNDIST est calculée à partir d'une analyse horizontale de l'arbre syntaxique. Cette analyse horizontale est basée sur un calcul de distance sur les liens syntaxiques identifiés dans la requête. Ce calcul est effectué à l'aide d'une analyse de dépendance entre les termes (réalisée par SYNTEX). Ainsi, une relation syntaxique entre deux mots adjacents (par exemple Reform/of) aura une

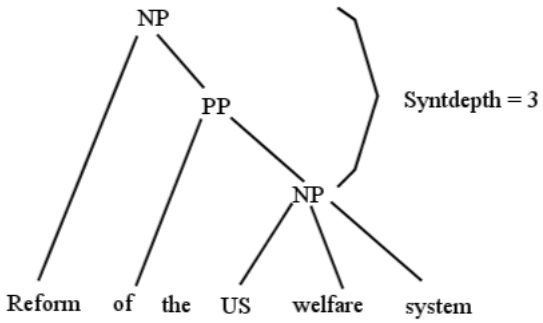

FIG. 4.3 – Valeur de SYNTDEPTH de la requête 185 de Trec-3 : Reform of the US welfare system

distance égale à 1. Si l'on considère par exemple le groupe nominal "the U.S welfare system" provenant de la requête 185, il existe un lien syntaxique entre le déterminant "the" et le nom "system", ce qui correspond à une distance syntaxique de 3. La valeur de SYNDIST est donc obtenue en divisant la somme des distances syntaxiques par le nombre de liens syntaxiques. Par exemple dans la figure 4.4, SYNDIST a une valeur

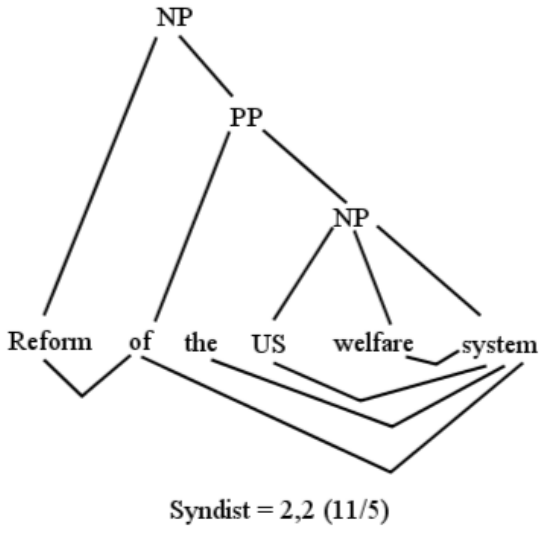

FIG. 4.4 – Valeur de SYNDIST de la requête 185 de Trec-3 : "Reform of the U.S welfare system"

égale à 11/5 soit 2,2.

Les deux CL qui viennent d'être présentées caractérisent deux types de complexité syntaxique des requêtes.

4.3.3 Les caractéristiques sémantiques des requêtes

La CL nommée *SYNSET* correspond au nombre moyen de *synset* dans lesquels les termes de la requête apparaissent. Cette information permet de mesurer le taux de polysémie du terme étudié. Les synonymes sont extraits directement du champ SYNSET de *WordNet*. Cette CL donne une indication sur l'ambiguïté sémantique de la requête. On cherche à mesurer le nombre de sens différents des termes. Plus les termes de la requête auront de synonymes, et plus la requête a des chances d'être ambiguë. Par exemple, en prenant le terme *reform* du titre de la requête 185, on constate qu'il apparaît dans 9 *synset* de *Wordnet*. La figure 4.5 nous montre le résultat obtenu avec *Wordnet* pour le terme *reform*.

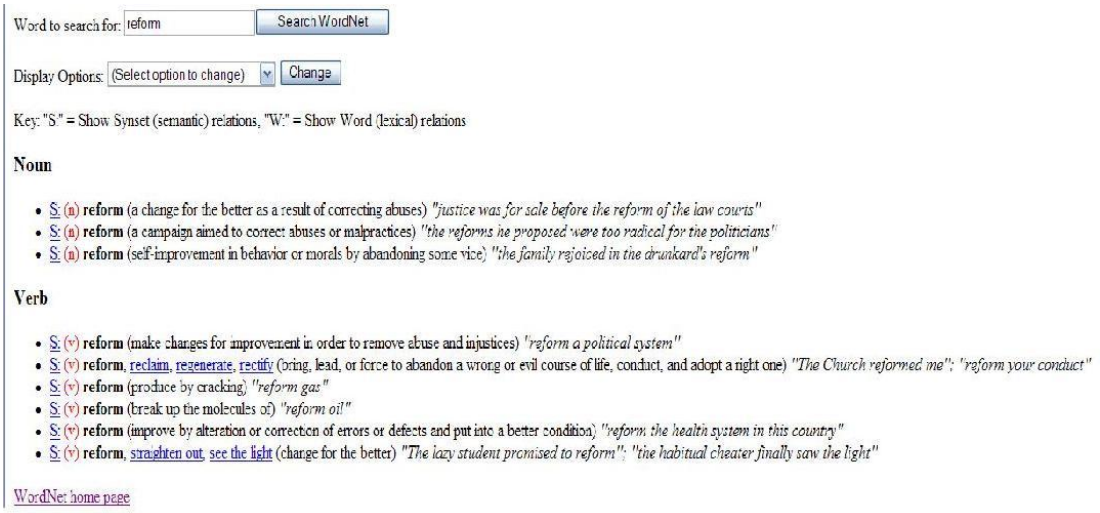

FIG. 4.5 – Nombre de synset auxquels appartient le terme *reform*

En calculant le nombre moyen de *synset* dans lesquels apparaissent les termes de la requête 185 (partie *title*), on obtient une valeur de SYNSET=4,5.

Nous venons de présenter les différentes CL que nous utilisons dans nos travaux, ainsi que la catégorie linguistique à laquelle elles appartiennent. Outre le fait qu'elles soient automatiquement extraites des requêtes, les CL que nous utilisons sont numériques et peuvent être traitées par des techniques statistiques comme la CAH ou les k-means (cf. chapitre 3). Le but est d'arriver à déterminer des classes de requêtes en fonction de leurs CL . Nous présentons dans la section suivante les différents outils qui sont utilisés pour extraire ces CL .

4.3.4 Les outils utilisés pour l'extraction des CL

Un certain nombre d'outils ont été utilisés lors de l'extraction des CL :

TreeTagger[1]

Cet outil permet l'étiquetage automatique des catégories grammaticales des mots avec l'application de la *lemmatisation* pour identifier des mots grammaticalement corrects. Il a été développé à *l'Institute for Computational Linguistics* à l'université de Stuttgart en Allemagne. TreeTagger peut être utilisé pour étiqueter l'allemand, l'anglais, le français, l'italien, l'espagnol, le portugais, le chinois ... Le tableau 4.8 donne un exemple de sortie obtenue avec TreeTagger.

Mot	Catégorie gramaticale	Lemme
The	DT	the
TreeTagger	NP	TreeTagger
is	VBZ	be
easy	JJ	easy
to	TO	to
use	VB	use

TAB. 4.8 – Exemple de sortie pour le texte "The TreeTagger is easy to use."

Dans le tableau 4.8, la catégorie grammaticale à laquelle appartiennent les mots est précisée, ainsi que la racine (ou lemme) du mot qui est utilisée lors de l' indexation.

Syntex

Cet analyseur de texte utilise un ensemble de règles grammaticales pour identifier les différentes relations syntaxiques qui existent entre les mots de la requête (par exemple les associations telles que verbe/objet, sujet/verbe, ...). Pour plus de détails sur cet outil, consulter les travaux présentés dans [FB01].

WordNet1.6 [2]

Il s'agit d'une base de donnée lexicale qui regroupe les noms, les verbes, les adjectifs et les adverbes dans des ensembles de synonymes appelés *synsets* qui expriment chacun un concept différent. Les *synsets* sont reliés par des relations lexicales.

CELEX (CEntre for LEXical Information)[3]

Il s'agit d'une base de données lexicale qui associe à chaque mot sa décomposition sous forme de *morphème*[4].

4.4 Expérimentations : détection du meilleur système en fonction des CL des requêtes

Dans un premier temps, nous avons utilisé des techniques d'analyse de données (ACP, CAH et k-means) pour classifier les requêtes en fonction de leurs CL. Dans

[1]Cet outil est disponible à l'adresse : www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/

[2]Disponible à l'adresse : http ://wordnet.princeton.edu/

[3]Cet outil est disponible à l'adresse : www.mpi.nl/world/celex

[4]Les plus petites unités de base en linguistique ayant une forme et un sens, et que l'on ne peut diviser en unités plus petites

les sections 4.4.1 et 4.4.2, nous présentons les différentes classes de requêtes que nous avons obtenues en analysant l'ensemble des requêtes de TREC, ainsi que les méthodes que nous avons utilisées à cet effet. Nous avons alors déterminé, pour chaque classe de requête, le(s) meilleur(s) système(s) à utiliser.

Dans un deuxième temps, nous avons constitué deux ensembles de requêtes : les requêtes d'entraînement et les requêtes de test. Dans la section 4.4.4, les requêtes d'entraînement nous permettent de réaliser la classification des requêtes et le choix du meilleur système à utiliser pour chaque classe. Les requêtes de test nous permettent de vérifier l'hypothèse selon laquelle la classification des requêtes permet de détecter la meilleure stratégie de recherche à utiliser. Pour cela, nous présentons la manière dont nous avons affecté les requêtes de test aux classes de requêtes dans la section 4.4.3. Nous évaluons ensuite les performances que nous obtenons pour chaque requête de test affectée à une classe donnée. Pour cela, le système que nous avons détecté comme étant le meilleur système à utiliser pour répondre aux requêtes d'une classe donnée est utilisé pour répondre aux requêtes de test de la classe. Nous comparons ensuite les résultats que nous obtenons avec ceux obtenus par les meilleurs systèmes ayant participé à TREC. Nous présentons dans la sections 4.4.4 les différents résultats que nous avons obtenus et nous y associons quelques commentaires.

4.4.1 La classification des requêtes

Dans cette section, nous présentons la méthode que nous avons employée pour la classification des requêtes. La classification est effectuée sur la base des CL des requêtes qui ont été automatiquement calculées. Les CL utilisées n'ont pas la même échelle de valeurs (par exemple, le nombre moyen de synonymes et le nombre d'acronymes). Pour annuler cet effet de taille et homogénéiser les variables, les données ont été préalablement centrées et réduites ([LMP06]).

Les données utilisées pour la classification sont stockées dans une matrice de dimension 180x13. Dans cette matrice, les requêtes représentent les individus (en lignes, soit 200) et les CL les variables (en colonne, soit 13). Chaque individu peut être interprété comme un vecteur initialement situé dans un espace à 13 dimensions, chacune des dimensions correspondant à une CL . De la même manière, chaque CL peut être interprétée comme un vecteur à 200 dimensions où chaque dimension correspond à une requête.

Après l'extraction des CL pour chaque requête avec les outils présentés dans la sous-section 4.3.4, les requêtes sont matérialisées par un ensemble de valeurs numériques.

$$\vec{Q} = (CL_1,\ CL_2,\ \ldots,\ CL_n). \tag{4.1}$$

Dans l'équation 4.1, chaque requête Q est représentée sous la forme d'un vecteur à n composantes, les composantes étant représentées par des CL . Pour l'ensemble des requêtes on obtient alors une matrice de dimension nxp ou n représente le nombre de requêtes, et p le nombre de CL .

$$A = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1p} \\ A_{21} & A_{22} & \dots & A_{2p} \\ A_{31} & A_{32} & \dots & A_{3p} \\ A_{i1} & A_{i2} & A_{ij} & A_{ip} \\ . & . & \dots & . \\ A_{n1} & A_{n2} & \dots & A_{np} \end{pmatrix}.$$

Dans la matrice A, l'élément A_{ij} représente la *j-ème* CL de la requête Q_i.

Dans la section section 4.4.4, les requêtes sont partitionnées en deux grandes catégories : les requêtes d'apprentissage, et les requêtes de test. Le partitionnement est réalisé grâce à un tirage aléatoire des requêtes, la proportion des requêtes d'apprentissage étant supérieure à celle des requêtes de test. Ce tirage est répété 10 fois, afin d'atténuer l'effet du tirage aléatoire des requêtes sur l'analyse que nous effectuons.

À travers une analyse statistique des données, nous essayons de voir les liens qui existent entre les requêtes et leurs CL . Cette analyse statistique est réalisée en 2 étapes. Dans la première étape, nous utilisons la technique de l'ACP (Analyse en Composantes Principales) pour visualiser les relations qui existent entre les CL et entre les requêtes. Les résultats de l'application de cette méthode à nos travaux sont présentés dans la section 4.4.1.1.

Dans la deuxième étape, nous effectuons un regroupement des requêtes par le biais de la technique de classification ascendante hiérarchique (CAH), qui est une méthode non supervisée (pas de connaissance *a priori* sur le nombre de classes) de classification. Cette méthode a été choisie dans le cadre d'une collaboration avec des collègues mathématiciens (voir dans le chapitre 7 pour plus de détails sur la collaboration). D'une manière générale, la mise en œuvre d'une méthode de classification non supervisée a pour but de mettre en évidence des groupes d'individus sans aucun *a priori* sur le nombre de groupes qu'il sera pertinent de retenir à l'issue de l'analyse. Le choix de cette méthode statistique est aussi lié à la taille des données que nous avons utilisées dans notre étude d'une part, et d'autre part à la performance de cette technique. La CAH nous permet d'une part de regrouper les requêtes en classes, et d'autre part de déterminer, grâce au dendrogramme, le nombre de classes souhaité. Les résultats que nous avons obtenus avec la CAH sont présentés dans l'annexe B. Dans notre approche, la CAH est couplée avec la technique de classification supervisée des k-means. Un des avantages de la méthode des k-means est qu'il n'est pas nécessaire de calculer au préalable les distances deux à deux entre tous les individus, opération très gourmande en temps et en espace mémoire dans certains algorithmes de classification. Dans le but de stabiliser les résultats de la classification, nous initialisons l'algorithme des k-means avec les barycentres des classes déterminées par la CAH . Le barycentre est considéré comme le représentant de la classe de requêtes. Ce barycentre peut toutefois ne pas coïncider avec une requête de la classe.Il représente alors une requête fictive.

4.4.1.1 L'ACP

L' ACP est une méthode statistique classique de réduction de dimension [MKB89]. Les données analysées proviennent d'une matrice qui correspond à un tableau comportant en lignes l'ensemble des requêtes, et en colonnes les différentes CL associées à ces requêtes. L' ACP permet de représenter les données initiales (dans notre cas, la matrice initiale a une dimension de 200*13) dans une dimension plus petite (en général en 2D ou en 3D. Nous nous limitons à la représentation 2D dans nos travaux).

Cette méthode permet de construire des composantes principales qui sont des combinaisons linéaires non corrélées des variables initiales. Dans notre étude, les variables sont représentées par les CL , et les individus par les requêtes. Après avoir appliqué l' ACP sur la matrice constituée des CL pour l'ensemble des requêtes, nous avons obtenu le premier résultat suivant :

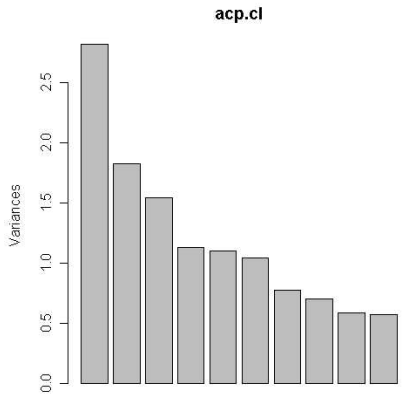

FIG. 4.6 – Contributions des valeurs propres aux axes (éboulis des valeurs propres)

La figure 4.6 indique les différentes contributions des valeurs propres aux axes de l'ACP. La première étape lors d'une ACP consiste à choisir la dimension des sous-espaces de projection. L'éboulis des valeurs propres traduisant la part d'inertie expliquée par chaque composante principale justifie une représentation en 3 dimensions. Nous limitons notre analyse aux 3 premiers axes car ils contiennent le maximum d'information (plus de 50% de l'information).

Le tableau 4.9 présente les corrélations que l'on obtient à l'issue de l'acp. Plus on avance dans les axes, plus les corrélations sont en moyenne plus faibles que pour les premiers axes. Dans le cas contraire, une corrélation plus importante indique que l'axe considéré est caractéristique de la variable. Par exemple, la première ligne du tableau 4.9 montre que la corrélation de la variable NBWORDS est plus grande lorsque l'on

	Axe1	Axe2	Axe3
NBWORDS	-0,145	+0,331	**+0,580**
NP	-0,061	+0,471	**-0,506**
ACRO	-0,107	+0,458	-0,410
PREP	-0,252	**+0,618**	+0,01
CC	+0,298	-0,192	+0,415
PP	-0,227	-0,239	+0,371
VBCONJ	-0,238	**-0,614**	-0,072
AVGSIZE	**+0,796**	-0,072	-0,215
AVGMORPH	**+0,809**	+0,021	+0,145
X,CONSTR	**+0,883**	-0,041	+0,055
AVGSYNSETS	**-0,616**	-0,078	+0,298
SYNTDEPTH	+0,164	**+0,526**	+0,449
SYNTDISTANCE	+0,215	+0,377	+0,326

TAB. 4.9 – Tableau de corrélation selon les 3 premiers axes principaux

considère l'axe3. Cette variable est donc mieux décrite sur cet axe que sur les axes 1 ou 2. On sélectionne dans le tableau 4.9 les plus grandes corrélations en valeur absolue. Ainsi, les variables CONSTR, AVGMORPH, AVGSIZE, et AVGSYNSETS sont caractérisées par le premier axe. L'axe 2 décrit les variables PREP, SYNTDEPTH et VBCONJ. Le troisième axe décrit les variables NBWORDS et NP.

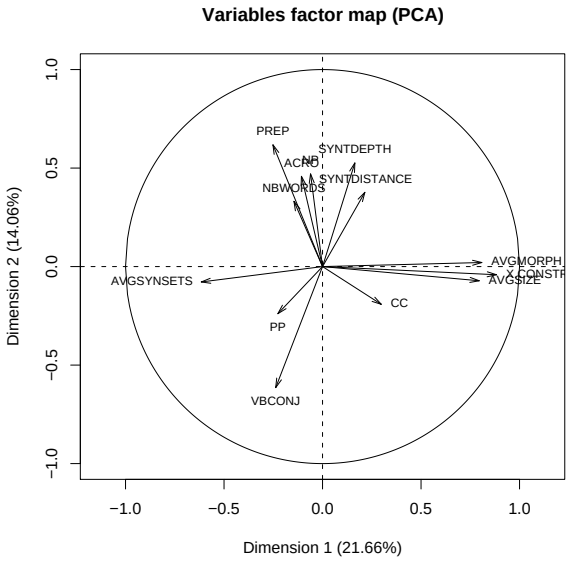

FIG. 4.8 – ACP suivant les 2 premiers axes - Représentation des variables

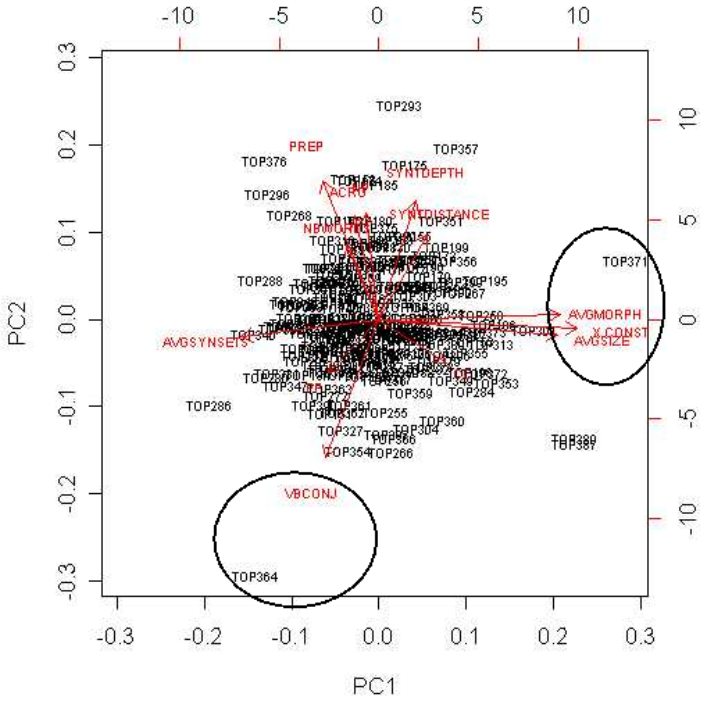

FIG. 4.7 – ACP suivant les 2 premiers axes principaux - représentation des variables et des individus

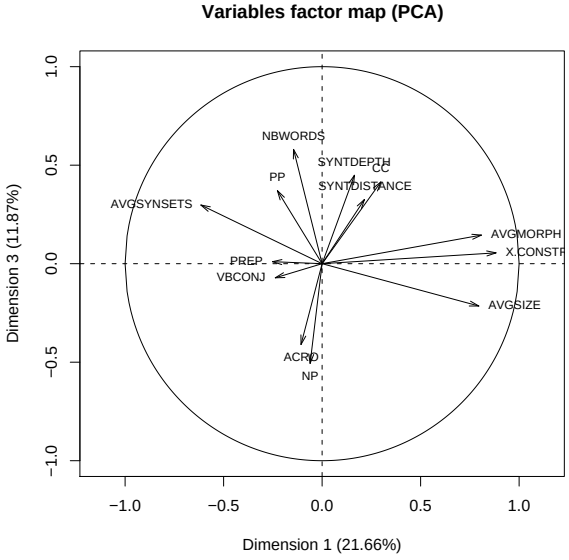

FIG. 4.9 – ACP suivant l'axe 2 et l'axe 3 - Représentation des variables

Les figures 4.7, et 4.10 représentent respectivement sur les axes 1 et 2 et les axes 1 et 3 les informations liées aux CL et aux requêtes. Sur ces figures, les individus (requêtes) et les variables (CL) sont représentés. Nous nous intéressons aux éléments que nous avons entourés sur les figures.

Dans la figure 4.7, on constate qu'il y a sur l'axe 1 une opposition entre 2 groupes de variables : <AVGSYNSETS> et <AVGMORPH, AVGSIZE, CONST>. Une interprétation que l'on peut faire de cette opposition est que si les requêtes possèdent en moyenne un grand nombre de synonymes, alors le nombre correspondant de morphèmes est faible et vice versa. Dans la figure, les requêtes qui sont le plus proches des extrémités sont fortement corrélées avec les CL situées dans leur direction. Par exemple la requête 371 est plus proche du groupe de CL <AVGMORPH, AVGSIZE, CONST> que de la CL <AVGSYNSETS>. Cette requête possède un faible nombre de synonymes (cf. l'opposition entre les 2 groupes de CL). Le même type d'analyse peut être effectué pour la requête 364 qui est proche de la CL *VBCONJ*. Cette requête possède un grand nombre de verbes conjugués.

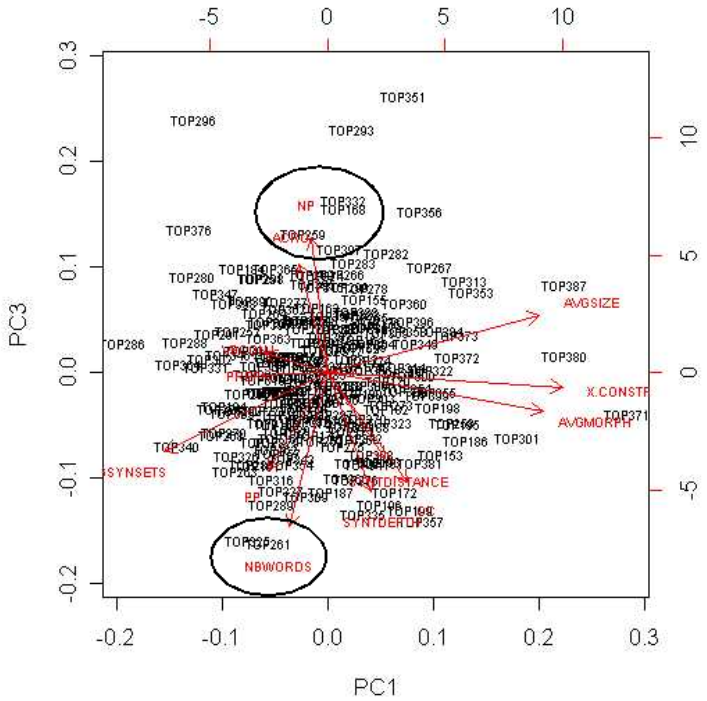

FIG. 4.10 – ACP suivant les axes 1 et 3 - Représentation des variables et des individus

Dans la figure 4.8, la longueur des vecteurs permet de déterminer la qualité de leur projection sur les axes. Plus le vecteur est grand, plus la projection qui est faite est bonne. Dans la figure 4.8, on constate par exemple que les vecteurs correspondant aux CL PP et CC ne sont pas bien représentés sur les 2 premiers axes principaux. En se limitant aux 2 premiers axes, on remarque que NBWORDS et NP semblent être dans la même direction.

Cependant, les figures 4.10, et 4.9 nous montrent que le vecteur NBWORDS caractérise le troisième axe principal et est opposé à NP. De manière similaire, l'analyse peut être poursuivie sur les deuxième et troisième axes principaux, mais l'information que l'on obtient est moins importante que celle obtenue à partir des deux premiers axes. Pour les individus situés au centre du graphique 4.7, il n'est pas possible d'analyser leurs relations avec les CL sans changer l'échelle de l'analyse.

La représentation que l'on obtient avec l' ACP nous indique des corrélations entre les individus et les variables que nous souhaitons exploiter plus en détail. Nous mettons en œuvre la technique de la CAH (section B) pour obtenir un regroupement des requêtes en fonctions de leurs CL .

4.4.1.2 La CAH

La méthode de la CAH est une méthode de classification que nous avons présentée en annexe B. Cette méthode apporte une vision complémentaire de celle de l'ACP, en proposant une classification des requêtes sans connaissance à priori du nombre de classes. Nous avons utilisé le critère de ward [LMP06] comme critère d'agglomération, comme le préconisent les statisticiens.

Le dendrogramme présenté dans la figure 4.11 donne le résultat de la classification que nous avons effectuée sur l'ensemble des requêtes.

L'interprétation d'un dendrogramme se fait en fonction d'un niveau de coupe de l'arbre fixé a posteriori. Ce niveau est habituellement fixé à une hauteur qui sépare "nettement" 2 noeuds successifs. Ici plusieurs choix pertinents du nombre de classes sont possibles : 6, 4 ou 3 classes (voir figure 4.11). Ces choix diffèrent exclusivement par un découpage plus ou moins poussé du 3ème groupe (à droite du dendrogramme sur la figure 4.11) en 1, 2 ou 4 classes. Nous avons choisi pour nos expérimentations de regrouper les requêtes en 3 classes. Le choix du nombre optimal de classes à retenir à l'issue de la classification a sans doute un impact sur les résultats qui seront obtenus. Toutefois, une étude complémentaire devrait permettre de déterminer le nombre de classes à utiliser, et de mesurer l'impact du nombre de classes sur les résultats. Dans la figure 4.11, nous avons de la gauche vers la droite la classe 1, la classe 2, et la classe 3.

Afin de tenter une description des classes en fonction des CL des requêtes, nous analysons les caractéristiques des centroïdes des 3 classes que nous avons déterminées. En effet, nous supposons que le centroïde de chaque classe caractérise la signature linguistique de la classe qu'il représente. Étant donné que le centroïde peut ne pas correspondre à une requête, on aurait pu choisir la requête la plus proche du centroïde

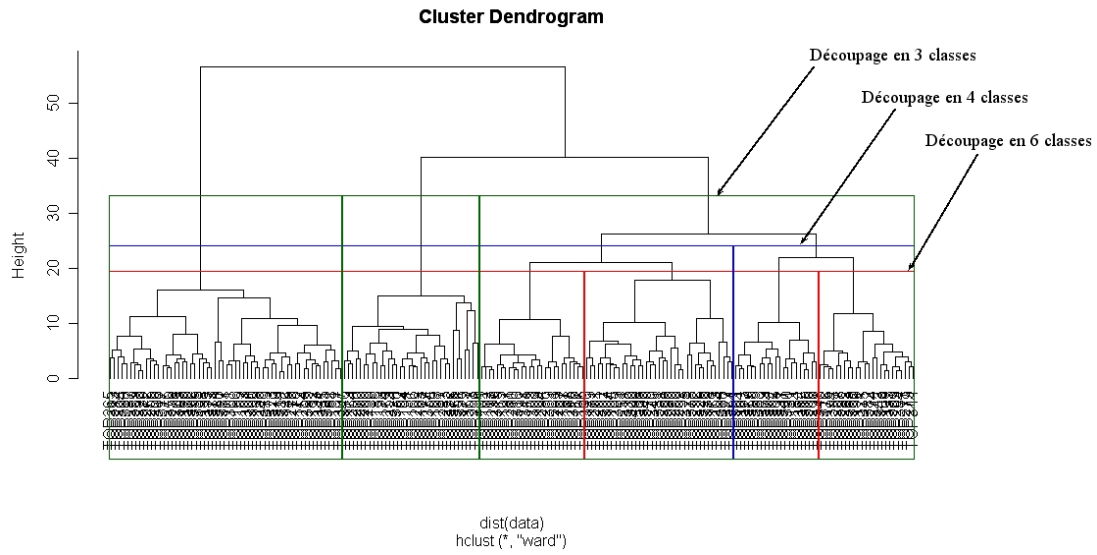

FIG. 4.11 – Dendrogramme représentant les classes de requêtes

comme étant la représentante de la classe considérée. Ainsi, les requêtes 287, 290 et 281 pourraient caractériser respectivement la signature linguistique des classes 1, 2 et 3 pour TREC5. Cependant, afin d'avoir une caractérisation la plus proche de la réalité, nous utilisons les différents centroïdes des classes pour déterminer la signature linguistique de la classe.

Les centroïdes sont composés de 13 CL correspondant à la moyenne des CL sur l'ensemble des requêtes d'une même classe. Nous indiquons dans le tableau4.10 les distances minimum, moyennes et maximum par rapport au centroïde de chaque classe.

distances	classe1	classe2	classe3
Minimum	2,3898	1,9960	1,5549
Moyenne	2,6625	3,5829	2,9593
maximum	2,5537	6,1855	4,7190

TAB. 4.10 – Distances intra-classe minimum, moyenne et maximum dans chaque classe de requête par rapport au centroïde de la classe

Les distances intra-classe sont calculées en fonction de la distance Euclidienne (racine carrée de la somme des carrés des différences entre les CL).

Dans le tableau 4.11, nous donnons les distances inter-classes qui existent entre les différentes classes de requêtes.

On remarque que la plus faible distance inter-classe est obtenue entre la classe1 et la classe3. La Classe1 et la classe2 sont les plus éloignées lorsque l'on considère les

	Distance inter-classe
Classe1-Classe2	3,5942
Classe1-Classe3	2,5226
Classe2-Classe3	3,3473

TAB. 4.11 – Distances inter-classe déterminées à partir du centroïde de chaque classe

centroïdes des classes.

En observant les caractéristiques de ces 3 représentants des classes, on dresse le tableau 4.12. Dans ce tableau, les deuxième et troisième colonnes sont déterminées en comparant les CL des 3 centroïdes. Dans la colonne " + ", la CL pour laquelle le centroïde considéré obtient la plus grande valeur par rapport aux 2 autres centroïdes est mentionnée. Pour la troisième colonne, on s'intéresse aux CL pour lesquelles la valeur est la plus faible parmi les 3 centroïdes.

Classe1	+	-
Centroïde1	SUFFIX	NP
	PP	ACCRO
	VBCONJ	PREP
	AVGMORPH	SYNTDIST
Classe2	+	-
Centroïde2	NP	NBWORDS
	ACCRO	VBCONJ
	PREP	SYNSET
	PP	SYNTDIST
Classe3	+	-
Centroïde3	NBWORDS	CC
	SYNSET	SYNTDEPTH
	SUFFIX	AVGMORPH

TAB. 4.12 – CL des 3 classes de requêtes de TREC5

Ainsi, dans ce tableau, la classe1 est définie comme étant celle qui comporte le plus grand nombre de mots suffixés (SUFFIX ; la classe3 contient le plus faible nombre de mots suffixés) et de pronoms personnels (PP ; la classe2 contient le plus faible taux de pronoms personnels). Dans cette classe (classe1), le centroïde traduit le fait qu'en moyenne, les requêtes de la classe1 contiennent le plus faible taux d'acronymes, de prépositions et de noms propres.

La classe2 est caractérisée par un fort taux de noms propres (NP), d'acronymes, de prépositions et de pronoms personnels. Cette classe contient en outre le plus faible taux de verbes conjugués, de synonymes et de prépositions.

Quant à la classe3, elle se distingue par un fort taux de synonymes, ainsi qu'un grand nombre de mots dans les requêtes de cette classe.

Pour mettre en valeur l'avantage de la classification des requêtes à partir de leurs CL, nous avons mené une série d'expérimentations. Dans un premier temps, nous avons découpé les requêtes en deux échantillons : les requêtes d'entraînement et les requêtes de test.

Afin d'évaluer la pertinence de la classification des requêtes, nous avons mis en place un protocole d'évaluation qui consiste à détecter pour chaque classe de requêtes d'entraînement, le(s) meilleur(s) système(s) à utiliser. Lors de la phase de test, les requêtes de test sont affectées dans une des classes déterminées, et nous comparons les résultats obtenus lorsque le système que nous avons détecté comme étant le plus indiqué à utiliser pour ladite classe de requêtes est utilisé. Afin de stabiliser les 3 classes obtenues, nous avons utilisé un algorithme de type k-means en partant des barycentres des classes obtenues par la CAH [LMP06].

Dans la section 4.4.2, nous donnons les résultats obtenus à l'issue de la classification, sans prendre en compte la phase d'apprentissage et de test des requêtes. Dans cette expérimentation, la classification est effectuée sur l'ensemble des requêtes. Nous présentons ensuite dans la section 4.4.3 le principe d'affectation des requêtes aux classes de requêtes. Dans la section 4.4.4, nous présentons les résultats que nous obtenons lorsque nous utilisons un apprentissage sur les classes de requêtes.

4.4.2 Analyse de la classification de l'ensemble des requêtes

Nous donnons dans cette section les résultats préliminaires que nous avons obtenus, c'est à dire sans intégrer une phase d'apprentissage et de test dans nos expérimentations. Nous utilisons la classification sur l'ensemble des requêtes que nous utilisons. Nous présentons ces résultats en deux étapes. Une première étape ou évaluation locale fait ressortir les analyses que nous avons effectuées en nous focalisant sur une classe de requêtes à la fois (section 4.4.2.1). La deuxième évaluation dite globale présente les performances moyennes que nous obtenons sur l'ensemble des requêtes traitées (section 4.4.2.2).

4.4.2.1 Évaluation locale

Dans le tableau 4.13, nous analysons les résultats obtenus pour les 3 classes de requêtes en utilisant les données de TREC5. Pour chaque mesure, nous comparons le résultat du meilleur système sur l'ensemble des requêtes de la classe choisie avec les résultats obtenus par le même meilleur système, mais cette fois sur l'ensemble des requêtes de la collection (50 requêtes par session de TREC). Par exemple, en considérant les requêtes de la classe 1, le système ETHme1 obtient la meilleure MAP (0,3407) de la classe (on l'appelle alors Best_{Cl1}). Ce même système obtient 0,3070 comme valeur de MAP sur l'ensemble des 50 requêtes (ETHme1_all) (idem pour les autres systèmes uwgxc1_all, genrl3_all, ...). Il peut cependant arriver que la meilleure performance pour une classe donnée soit obtenue par plusieurs systèmes. Dans ce cas, nous mentionnons les résultats de l'ensemble des systèmes. Par exemple, lorsque l'on considère la mesure P@5 pour

la classe 2 de requêtes, 4 systèmes obtiennent la meilleure performance. Ils s'agit des systèmes uwgcx0, uwgcx1, genrl3, et genrl4.

Classe1		Classe2		Classe3	
	MAP		**MAP**		**MAP**
Best$_{Cl1}$	**0,3407**	Best$_{Cl2}$	**0,3396**	Best$_{Cl3}$	**0,3255**
Ethme1_all	0,3070	Uwgcx0_all	0,2944	LNaDesc2_all	0,2585
	R-Prec		**R-Prec**		**R-Prec**
Best$_{Cl1}$	**0,3633**	Best$_{Cl2}$	**0,3829**	Best$_{Cl3}$	**0,3528**
Ethme1_all	0,3305	Uwgcx0_all	0,3444	LNaDesc2_all	0,2929
	P@5		**P@5**		**P@5**
Best$_{Cl1}$	**0,7481**	Best$_{Cl2}$	0,5200	Best$_{Cl3}$	**0,6769**
Ethme1_all	0,6160	Uwgcx0_all	0,5560	LNaDesc2_all	0,4960
		Uwgcx1_all	0,5520		
		Genrl3_all	0,5800		
		Genrl4_all	0,5240		
	P@10		**P@10**		**P@10**
Best$_{Cl1}$	**0,6778**	Best$_{Cl2}$	0,4400	Best$_{Cl3}$	**0,5923**
Ethme1_all	0,5460	Genrl3_all	0,4700	LNaDesc2_all	0,4780
	P@15		**P@15**		**P@15**
Best$_{Cl1}$	**0,6420**	Best$_{Cl2}$	0,3867	Best$_{Cl3}$	**0,5026**
Ethme1_all	0,5147	Ethme1_all	0,5147	LNaDesc2_all	0,4280

Tab. 4.13 – Performances moyennes des systèmes de TREC5 sur les classes de requêtes

Nous pouvons voir dans le tableau 4.13 que l'utilisateur gagnerait dans la pertinence des résultats qui lui sont restitués si le système ETHme1 était choisi pour restituer les documents pour la classe 1 de requêtes, indépendamment de la mesure choisie. Notons que le système ETHme1 correspond aussi au système qui obtient les meilleures performances moyennes sur l'ensemble des requêtes. De même, pour la classe 3, le meilleur système (LNaDesc2) permet d'obtenir de meilleures performances (toutes mesures confondues) que celles obtenues par le même système (LNaDesc2), lorsque l'ensemble des requêtes est considéré. On peut conclure pour ces 2 classes qu'il est possible de choisir le meilleur système à utiliser.

Il en va différemment pour la classe 2. En effet, le système uwgcx0 que nous choisissons comme le système à utiliser permet effectivement d'obtenir de meilleures performances pour les requêtes de la classe 2 que celles que ce même système obtient pour l'ensemble des requêtes. Le meilleur système est choisi en fonction de ses performances en termes de MAP et de R-Précision. Pour les mesures de hautes précisions par contre, les résultats obtenus par le système que nous sélectionnons sont moins bons que ceux que les systèmes obtiennent sur l'ensemble des requêtes. Par exemple, genrl3 (resp. ethme1) est le système que nous avons choisi pour répondre aux requêtes de la classe 2. Le système genrl3 obtient une P@10 de 0,44 contre 0,47 pour l'ensemble des requêtes (resp. ethme1 obtient une valeur de P@15 de 0,3867 contre 0,5147 pour l'ensemble des

requêtes). De même, 4 systèmes obtiennent la meilleure P@5 pour les requêtes de la classe 2. Les performances de ces 4 systèmes sur l'ensemble des requêtes sont toutes supérieures à celles qu'ils obtiennent sur les requêtes de la classe 2. Le fait qu'il y ait plusieurs systèmes qui obtiennent les meilleures performances n'a pas d'incidence sur la manière de choisir le meilleur système. Dans ce cas (plusieurs systèmes ont la meilleure performance), le meilleur système n'est pas associé à un système particulier mais plutôt à une valeur correspondant à la meilleure performance. Pour revenir au cas de la P@5 pour la classe 2, une explication que nous pouvons donner est que les systèmes uwgcx0, uwgxc1, et genrl3 en particulier retrouvent plus de documents pertinents (parmi les 5 premiers documents) en moyenne sur l'ensemble des requêtes que pour les requêtes de la classe 2. Les performances moyennes pour la haute précision sont faibles par rapport à celles obtenues pour la classe 1 et la classe3. Nous ne détectons donc pas pour cette classe le meilleur système à utiliser.

L'analyse que nous avons faite dans cette section est centrée sur un nombre de requêtes déterminé par la CAH. Cependant, pour effectuer une comparaison sur le même nombre de requêtes, nous présentons dans la section suivante l'évaluation dite globale que nous avons réalisée.

4.4.2.2 Évaluation globale

Les résultats présentés dans la section précédente comparent les performances par classe de requêtes avec les performances des systèmes pour toutes les requêtes de TREC. Étant donné que pour une année de TREC les 50 requêtes sont réparties en 3 classes, nous étudions dans cette section les performances que nous obtenons en sélectionnant le meilleur système à utiliser pour l'ensemble des classes (donc sur 50 requêtes) par rapport au meilleur système global (performances initiales sur 50 requêtes).

	Moyenne sur les 3 classes de requêtes (meilleur système pour chaque classe)	Moyenne pondérée en fonction de la taille des classes	Meilleur système en termes de MAP, R-précision, P@5, P@10, et P@15
MAP	0,3353	0,3356	0,3070 (ethme1)
R-Prec	0,3663	0,37	0,3444 (uwgcx0)
P@5	0,6448	0,6709	0,6160 (ethme1)
P@10	0,5700	0,5912	0,5460 (ethme1)
P@15	0,5104	0,5233	0,5147 (ethme1)

TAB. 4.14 – Performances moyennes des systèmes de TREC5 par rapport au meilleur système

Le tableau 4.14 indique en colonne 2, la moyenne par rapport à la classe de requêtes (moyenne de Best_{Cl1}, Best_{Cl2}, et Best_{Cl3} pour la MAP, R-précision, etc.). Dans

la colonne 3, cette moyenne est pondérée en fonction de la taille de la classe considérée. Par exemple, la classe 1 contient 29% de requêtes sur l'ensemble des 200 requêtes. Dans la dernière colonne, nous indiquons les résultats du meilleur système pour la mesure concernée sur l'ensemble des requêtes. Le meilleur système (système entre parenthèses) en termes de MAP, R-précision, P@ pour TREC5 est aussi le meilleur système pour toutes les autres mesures, sauf pour la R-précision. Le tableau 4.14 montre qu'en moyenne les résultats peuvent être améliorés. Par exemple, le choix du meilleur système pour chacune des classes permet d'obtenir une MAP de 0,3353 alors qu'elle était de 0,3070 pour le meilleur des systèmes participants à TREC5, soit une amélioration de 9,22%. En prenant la moyenne pondérée (0,3356), l'amélioration passe de 9,22% à 9,32% pour la MAP. Nous faisons référence à cette moyenne pondérée par l'appelant Best$_{Cl}$_p (cf. tableaux 4.15, 4.16, et 4.17).

MAP		**R-Prec**	
Best$_{Cl}$|Best$_{Cl}$_p		**Best$_{Cl}$|Best$_{Cl}$_p**	
(0,3353 / 0,3356)	***%A***	***(0,3663 / 0,37)***	***%A***
ETHme1_all (0,3070)	(9,22% | 9,32%)	ETHme1_all (0,3305)	(10,83% | 11,95%)
Uwgcx0_all (0,2944)	(13,89% | 14%)	Uwgcx0_all (0,3444)	(6,36% | 7,43%)
LNaDesc2_all (0,2585)	(29,71% | 29,83%)	LNaDesc2_all (0,2929)	(25,06% | 26,32%)

TAB. 4.15 – Comparaison globale des performances des systèmes de TREC5 avec Best$_{Cl}$ - MAP et R-Prec

P@5		**P@10**	
Best$_{Cl}$|Best$_{Cl}$_p		**Best$_{Cl}$|Best$_{Cl}$_p**	
(0,6483/0,6709)	***%A***	***(0,57/0,5912)***	***%A***
ETHme1_all (0,6160)	(5,24% | 8,91%)	ETHme1_all (0,5460)	(4,4% | 8,28%)
uwgcx0_all (0,5560)	(16,6% | 20,66%)	genrl3_all (0,47)	(21,28% | 25,79%)
uwgcx1_all (0,5520)	(17,45% | 21,54%)	LNaDesc2_all (0,4780)	(19,25% | 23,68%)
genrl3_all (0,58)	(11,78%|15,67%)		
genrl4_all (0,5240)	(23,72% | 28,03%)		
LNaDesc2_all (0,4960)	(30,71% | 35,26%)		

TAB. 4.16 – Comparaison globale des performances des systèmes de TREC5 avec Best$_{Cl}$ - P@5 et P@10

P@15 **Best$_{Cl}$|Best$_{Cl}$_p** ***(0,5104 / 0,5233)***	***%A***
ETHme1_all (0,5147)	(-0,8% | 1,67%)
LNaDesc2_all (0,4280)	(19,25% | 22,27%)

TAB. 4.17 – Comparaison globale des performances des systèmes de TREC5 avec Best$_{Cl}$ - P@15

Dans les tableaux 4.15, 4.16, et 4.17, nous détaillons le tableau 4.14 en indiquant le pourcentage d'amélioration que nous obtenons par rapport au meilleur système que nous sélectionnons, ainsi que par rapport aux différentes mesures de performance. Nous comparons le système recommandé par notre approche pour une mesure donnée aux systèmes ayant obtenu la meilleure performance pour cette mesure. Par exemple pour la MAP, Best$_{Cl}$ correspond à la moyenne obtenue par Best$_{C}l_i$, i représentant le numéro de la classe, et Best$_{C}l_i$_p à la moyenne pondérée du système que nous préconisons. Ainsi, nous obtenons une MAP de 0,3353 pour le système recommandé sur l'ensemble des classes de requêtes (0,3356 lorsque la moyenne pondérée est prise en compte). Dans la première colonne, les systèmes ETHme1, uwgcx0, et LNadesc2 correspondent respectivement aux meilleurs systèmes pour la classe 1, la classe 2, et la classe 3. Les chiffres indiqués entre parenthèses représentent les valeurs obtenues par ces systèmes sur l'ensemble des requêtes.

La première ligne donne les performances moyennes (sur les 3 classes de requête) du système que nous recommandons (Best$_{Cl}$), ainsi que les performances de Best$_{C}l_i$_p. %A représente le pourcentage d'amélioration lorsqu'on compare respectivement les performances du système Best$_{Cl}$ et de Best$_{C}l_i$_p avec celles des meilleurs systèmes. Par exemple, en considérant la mesure de R-précision, le système Best$_{Cl}$ obtient une valeur de 0,3663 (0,37 pour Best$_{C}l_i$_p). Nous remarquons que dans ce cas, le meilleur système pour la R-précision (uwgcx0) obtient 0,3444 comme valeur de performance, ce qui correspond à une amélioration de 6,363% lorsqu'on utilise Best$_{Cl}$ à la place de uwgcx0 (7,43% avec Best$_{C}l_i$_p). Nous avons indiqué que le système uwgcx0 est le meilleur pour la R-précision, et le système ETHme1 est le meilleur pour les autres mesures. On remarque dans les tableaux 4.15, 4.16 et 4.17 que Best$_{Cl}$ est meilleur que les meilleurs systèmes et pour toutes les mesures, sauf pour la P@15 où les performances de ETHme1 sont dégradées (-0,8%). L'amélioration obtenue par rapport aux meilleurs systèmes et pour les autres mesures varie de 4,4% à 30,71%. La méthode de sélection du meilleur système semble apporter une amélioration par rapport aux meilleurs systèmes. Pour l'ensemble des mesures, l'amélioration moyenne par rapport aux meilleurs systèmes sélectionnés est supérieure à 14% (soit 17,72% pour la MAP, 14,08% pour la R-précision, 17,58% pour la P@5, 14,98% pour la P@10, et 9,23% pour la P@15). En considérant Best$_{C}l_i$_p, le système que nous proposons permet d'obtenir de meilleures performances

que le meilleur système, indépendamment de la mesure considérée. Les améliorations vont de 7,43% (R-Prec) à 35,26% (P@5).

Les mêmes types d'analyse peuvent être faites pour les autres mesures et les autres années de TREC. Sur les 61 systèmes ayant participé à la campagne TREC5, nous sommes alors capables de détecter à 98,36% (sur un ensemble de systèmes) le meilleur système à utiliser pour une mesure donnée.

Nous remarquons dans le tableau 4.13 que nous recommandons le système ETHME1 pour la P@15, aussi bien pour les requêtes de la classe 1 que celles de la classe 2. La question que l'on peut se poser est de savoir quelle est la meilleure classe pour laquelle ce système doit être utilisé, ou de vérifier que ce système est bien le meilleur à utiliser pour les 2 classes. Pour répondre à ce genre de questions, nous proposons d'analyser dans la section suivante l'évaluation que nous avons effectuée en intégrant des phases d'apprentissage et de test lors de la classification des requêtes. Les systèmes que nous détectons comme étant les meilleurs sont alors analysés sur l'ensemble des itérations que nous avons effectuées.

4.4.3 Affectation des requêtes aux classes

Nos expérimentations se sont déroulées dans un premier temps sans apprentissage (l'ensemble des requêtes est classé) (cf. sections 4.4.1et 4.4.2), et dans un second temps suivant un processus d'apprentissage et de test. Nous procédons dans cette section à un tirage aléatoire de 180 requêtes (requêtes d'entraînement), et de 20 requêtes (requêtes de test). Ce tirage a été croisé dix fois afin de réduire les effets du tirage aléatoire.

Étant donné la représentation vectorielle des requêtes de test, le principe de l'affectation d'une requête à une classe est basé sur le calcul de la distance qui sépare la requête au barycentre de la classe. Plus la distance est faible, plus la requête de test possède des CL similaires avec celles de la classe de requêtes considérée. Pour chaque requête de test, il est alors possible de déterminer la classe à laquelle elle appartient. Nous avons déterminé les classes à partir de 180 requêtes provenant des campagnes d'évaluation TREC3, TREC5, TREC6 et TREC7. Cependant, les systèmes et les requêtes d'une session TREC à l'autre sont différents. Nous avons donc effectué un filtrage des requêtes en fonction de la campagne TREC à laquelle elles appartiennent. Le tableau 4.18 donne la répartition des requêtes pour chaque année de TREC et pour chaque classe de requêtes. Dans cette table Nq représente le nombre de requêtes par classe.

	Nq	TREC3	TREC5	TREC6	TREC7
Classe1	58 (29%)	14	12	17	15
Classe2	34 (17%)	15	10	5	4
Classe3	108 (54%)	21	27	29	31

TAB. 4.18 – Répartition des requêtes TREC dans les classes

Le tableau 4.18 montre par exemple que la classe 3 contient 54% des requêtes et

que 28,7% de ces requêtes proviennent de TREC7. En fonction de ces 3 classes de requêtes, nous avons évalué les performances des systèmes. Comme indiqué dans la sous-section 4.2, nous avons utilisé le programme trec_eval pour calculer les performances des systèmes participants. Ce programme fournit en sortie pour chaque système un ensemble de mesures. Nous avons choisi parmi ces mesures la précision moyenne (MAP), la R-précision, la P@5, P@10, et P@15. Pour chacune de ces mesures, nous avons classé les systèmes en fonction de leur performance, et cela pour chaque année de TREC.

4.4.4 Apprentissage et classification des requêtes

L'analyse présentée dans la section 4.4.2 correspond à une première étape de nos travaux. Les requêtes qui n'ont pas été choisies comme requêtes d'apprentissage constituent dans cette étape l'ensemble des requêtes de test. L'objectif principal de la phase de test est d'arriver à affecter les requêtes de test dans les "bonnes" classes. Pour cela, il est indispensable de pouvoir comparer les requêtes de test aux classes qui ont été définies. Nous avons appliqué la classification sur l'ensemble des requêtes dont nous disposons, et les résultats que nous obtenons ne sont pas forcément généralisables en fonction d'une autre classification. Afin de nous assurer de la conformité de nos résultats, nous avons utilisé une technique d'apprentissage et de test. Durant la phase d'apprentissage, 180 requêtes permettent de déterminer les 3 classes de requêtes. Les 20 requêtes restantes sont alors utilisées pour évaluer la classification. Nous avons réitéré ce processus dix fois et à chaque itération, les requêtes de test sont affectées dans une classe et nous comparons les résultats obtenus pour chaque requête de test (lorsque le système recommandé par notre méthode est utilisé) aux autres résultats obtenus par les autres systèmes.

Dans la figure 4.12, nous présentons pour chacune des 10 itérations la répartition des requêtes d'entraînement dans les classes de requêtes. Il ressort de cette figure que pour chaque itération, les requêtes de la classe 1 sont les plus nombreuses, suivent les requêtes de la classe 2, et les requêtes de la classe 1 sont les moins nombreuses. Il y a en moyenne (sur les 10 itérations) 83 requêtes dans la classe 1, 62 requêtes dans la classe 2, et 35 requêtes dans la classe 3. Dans la figure 4.12, aucune distinction n'est faite entre les requêtes des différentes campagnes TREC que nous avons utilisées.

Comme nous l'avons mentionné en introduction de ce chapitre, nous avons filtré les requêtes et les systèmes en fonction de la campagne TREC à laquelle ils se réfèrent.

	TREC3	TREC5	TREC6	TREC7	Moyenne
Classe1	8,00%	14,00%	12,50%	11,50%	11,50%
Classe2	8,50%	5,50%	10,50%	9,00%	8,38%
Classe3	8,00%	5,50%	3,00%	4,00%	5,125%
Total	24,5%	25%	26%	24,5%	25%

TAB. 4.19 – Répartition des requêtes de test dans les classes (moyenne sur 10 itérations)

Le tableau 4.19 donne la répartition moyenne (sur l'ensemble des 10 itérations) des

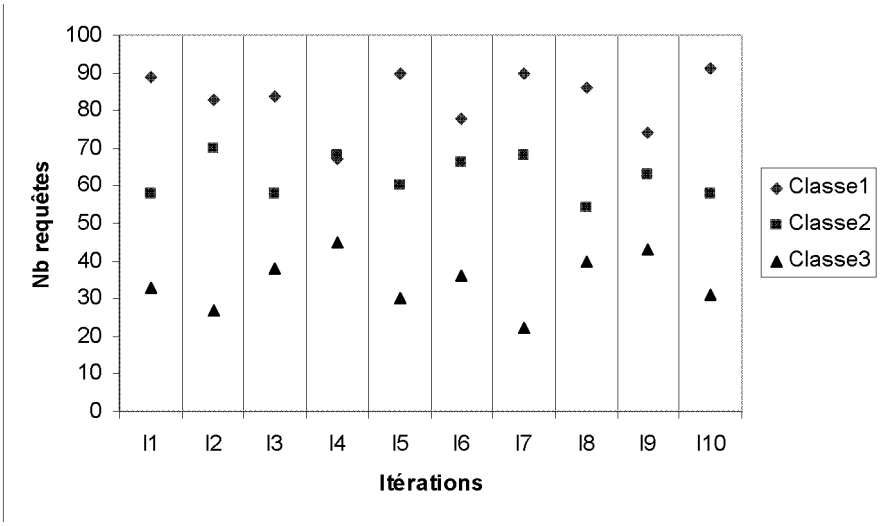

FIG. 4.12 – Répartition des requêtes d'entraînement dans les classes (requêtes de TREC3, TREC5, TREC6, et TREC7 confondues)

20 requêtes de test dans les différentes classes, en fonction des années de TREC. En moyenne, la classe 1 de requêtes comprend le plus grand nombre de requêtes.

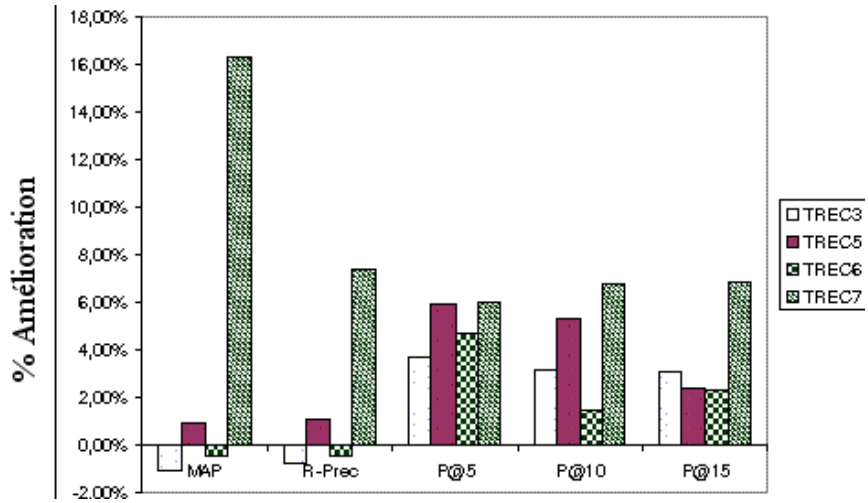

FIG. 4.13 – Comparaison globale des différents pourcentages d'amélioration obtenus en utilisant le système BestCl à la place des meilleurs systèmes de chaque année

Dans la figure 4.13, nous faisons une comparaison globale des meilleurs systèmes avec le système recommandé par notre approche. Nous appelons ici meilleurs systèmes, les systèmes ayant obtenus les meilleures performances pour une mesure donnée et une classe de requêtes. Il peut toutefois arriver que ces meilleurs systèmes soient également ceux qui ont obtenus les meilleures performances pour l'ensemble des requêtes et pour la même mesure. Dans la figure 4.13, nous comparons les améliorations (en pourcentage) que l'on obtient si le système Best_{Cl} est utilisé à la place des meilleurs systèmes de chaque année, pour les requêtes correspondantes. Rappelons que Best_{Cl} correspond au meilleur système que nous détectons.

Best_{Cl} permet d'améliorer les performances par rapport aux meilleurs systèmes de TREC sauf pour la mesure MAP (Best_{Cl} : 0,4181 et inq102 : 0,4226, amélioration : -1,08%) et R-précision (Best_{Cl} : 0,4491 et inq102 : 0,4524, amélioration : -0,72%) de TREC3, ainsi que pour la MAP (Best_{Cl} : 0,4611 et uwmt6a0 : 0,4631, amélioration : -0,43%) et la R-précision (Best_{Cl} : 0,4872 et uwmt6a0 : 0,4893, amélioration : -0,48%) de TREC6. Dans la figure 4.13, nous remarquons des améliorations de performance pour chaque année de TREC pour toutes les mesures de haute précision. Par exemple pour la P@10, les améliorations varient de 1,48% (TREC6) à 6,73% (TREC7). La plus grande amélioration est obtenue pour la MAP et est de 16,27% (TREC7). La plus petite amélioration est de 0,97% pour la MAP et pour TREC5. Le pourcentage d'amélioration moyen sur l'ensemble des campagnes TREC varie de 1,81% (R-précision) à 5,08% (P@5).

4.5 Classification des requêtes en fonction de leur difficulté

Nous avons analysé la classification des requêtes en fonction de leurs CL . Certains travaux dans la littérature s'intéressent à la prédiction de la difficulté des requêtes [MT05], afin de proposer la meilleure stratégie à utiliser. Dans [DHMO04] par exemple, les auteurs définissent les requêtes difficiles comme étant celles pour lesquelles la moyenne des systèmes obtient de faibles performances en terme de précision. Nous cherchons à établir dans cette section un rapprochement entre les classes de requêtes définies à l'aide de leurs CL , et 3 classes de requêtes liées à la difficulté des requêtes. Nous nous inspirons des travaux présentés dans [DHMO04] sur la difficulté des requêtes pour la présente analyse. Nous utilisons dans cette expérimentation les résultats des systèmes en terme de précision.

Nous regroupons les requêtes en trois classes : *difficiles*, *intermédiaires* et *faciles*. Les requêtes faciles sont celles pour lesquelles la moyenne des performances des SRI est la plus élevée. Les requêtes difficiles regroupent quant à elles les requêtes pour lesquelles la majorité des SRI obtiennent des performances moyennes faibles ; les performances étant mesurées par rapport à la précision.

Le tableau 4.20 donne un récapitulatif des performances moyennes obtenues par les systèmes sur les 3 classes de requêtes que nous avons présentées dans les sections précédentes.

	Classe1	Classe2	Classe3	Moyenne
TREC3	0,081	0,129	0,139	0,121
TREC5	0,06	0,03	0,061	0,055
TREC6	0,049	0,062	0,041	0,046
TREC7	0,053	0,07	0,063	0,063
Moyenne	0,061	0,073	0,076	0,07125

TAB. 4.20 – Comparaison des performances moyennes des systèmes (en terme de précision) pour chaque classe de requête

Dans le tableau 4.20, la dernière colonne représente la précision moyenne obtenue pour l'ensemble des requêtes pour une année donnée. En regardant la ligne correspondant à TREC3 par exemple, on constate que les SRI obtiennent une précision moyenne proche de la moyenne des précisions moyennes (0,121) pour les requêtes de la classe2, soit 0,129. En nous référant à la définition donnée dans le paragraphe précédant, on peut dire que les requêtes de la classe2 correspondent à des requêtes *intermédiaires*. La classe3 obtient la plus grande précision moyenne ; on peut dire que la classe3 correspond à la classe des requêtes *faciles*. La classe1 quant à elle contient des requêtes dites *difficiles*. En observant les résultats obtenus pour TREC6, les requêtes *faciles* sont associées à la classe2. Les requêtes *difficiles* correspondent à la classe3, et les requêtes *intermédiaires* à la classe1. Pour TREC5, la classe 1 peut être associée aux requêtes faciles, tandis que la classe 2 correspond aux requêtes difficiles.

Il n'est pas possible de dégager une tendance commune pour chaque classe de requêtes (en fonction de leur difficulté) d'une année à l'autre, car les requêtes sont dans un premier temps catégorisées en fonction de leurs CL , puis pour chaque classe les requêtes sont analysées en fonction des performances moyennes des systèmes. Toutefois, la dernière ligne du tableau nous indique que la classe1 correspond en moyenne à la classe des requêtes *difficiles*.

4.6 Conclusion

Dans les expérimentations présentées, nous avons fait l'hypothèse que les requêtes pouvaient être classées en fonction de critères linguistiques. Nous avons aussi fait l'hypothèse qu'il était possible d'associer un système à utiliser pour chaque classe de requêtes ainsi déterminée. Pour évaluer ces hypothèses, nous avons procédé en deux étapes : nous avons dans un premier temps considéré les résultats que nous aurions obtenus si l'on était capable de choisir le système à utiliser pour une catégorie de requêtes. La sélection des systèmes est basée sur les mesures MAP, Précision, P@5, P@10, et P@15.

Les catégories de requêtes ont été obtenues en utilisant les caractéristiques linguistiques de ces dernières. En supposant que nous sommes capables de détecter le meilleur système à utiliser pour une classe de requête, nous montrons que nous améliorerions les résultats de la recherche. Dans une première approche, les CL de l'ensemble des re-

quêtes nous ont permis de classifier les requêtes en 3 grandes classes. Pour chaque classe, nous proposons d'utiliser le système qui obtient les meilleures performances. Dans la majorité des cas, (sauf pour les mesures de haute précision de la classe 2), nous détectons le meilleur système à utiliser puisque les performances que nous obtenons sont meilleures que celles des meilleurs systèmes. En considérant l'ensemble des requêtes (moyenne sur les 3 classes déterminées), la méthode proposée permet d'obtenir des performances supérieures à celle du meilleur système, pour chacune des mesures utilisées. Ainsi, l'amélioration obtenue est de 9,22% pour la MAP, 6,36% pour la R-précision, 4,66% pour la P@5, 4,4% pour la P@10 ; en revanche, la P@15 est diminuée de 0,84%.

Dans une deuxième étape, nous avons évalué la détection du meilleur système à utiliser pour chaque classe de requêtes en utilisant un apprentissage (phase d'entraînement pour déterminer les classes de requêtes, et phase de test pour voir si l'on classifie correctement les requêtes de test). Nous avons trouvé que dans ce cas aussi, nous améliorons les résultats que les meilleurs systèmes ont obtenus. Nous montrons par exemple que lorsque le système recommandé par notre approche est utilisé à la place du meilleur système (le meilleur système de chaque classe de requêtes est sélectionné par apprentissage en fonction de l'année considérée), les améliorations sur l'ensemble des collections de documents que nous avons utilisées varient de 1,48% à 6,73% pour la P@10. Pour la MAP, la plus grande amélioration est de 16,27%, et de 5,08% pour la P@5.

La catégorisation linguistique des requêtes est donc une piste pour l'adaptation des stratégies de recherche en fonction des requêtes, bien que nous n'ayons pas pu montrer un lien direct entre les catégories de requêtes déterminées par leurs CL et celles déterminées par les performances des systèmes en termes de précision (ref. sous-section 4.5).

Nous proposons dans la suite d'appliquer des techniques de fusion aux classes de requêtes que nous avons déterminées, notamment dans le chapitre 6 où nous proposons de coupler la fusion à la classification. Avant d'appliquer la fusion aux classes basées sur les CL des requêtes, nous avons mené une série d'expérimentations sur la fusion, sans tenir compte dans un premier temps des classes de requêtes que nous avons déterminées. Nous abordons dans le chapitre suivant deux techniques simples de fusion de données. D'autres hypothèses sont faites et vérifiées à travers une série d'expérimentation, notamment l'application de la fusion en fonction de la typologie des requêtes (*faciles*, *intermédiaires* et *difficiles*) que nous avons introduite dans ce chapitre.

Chapitre 5

Fusion de systèmes

5.1 Introduction

Ce chapitre présente une étude que nous avons menée sur la fusion de données. Nous proposons deux approches de fusion. La première, dite technique de fusion systématique des SRI est présentée dans la section 5.2. C'est une méthode simple de fusion qui utilise les opérateurs ensemblistes d' UNION et d'INTERSECTION pour combiner les résultats des systèmes. L'objectif visé est de quantifier l'apport de telles techniques. Nous nous situons dans le cas où aucune information sur le degré de similarité entre la requête et les documents n'est disponible. Dans ce contexte, un document est soit pertinent, soit non pertinent. On peut par exemple prendre le cas des moteurs de recherche sur Internet où le score des documents est rarement connu.

La deuxième méthode de fusion que nous proposons est une technique de fusion adaptative qui fusionne différents systèmes sur la base du taux de chevauchement [Lee97] et de la typologie des requêtes (faciles ou difficiles) (cf. section 5.3). Nous terminons ce chapitre en présentant les différentes expérimentations que nous avons menées ainsi que les résultats que nous avons obtenus (section 5.4.1).

5.2 Fusion systématique des systèmes

Les méthodes de fusion de données proposées dans la littérature s'appuient sur la prise en compte de la similarité entre la requête et les documents, c'est-à-dire le degré de pertinence des documents restitués par les systèmes. Cependant, cette information n'est pas toujours disponible comme par exemple dans la tâche de *détection de la nouveauté* de TREC, où un document doit être considéré par le système soit comme pertinent, soit comme non pertinent. Nous proposons dans cette section une méthode de fusion que nous qualifions de *systématique*, qui ne nécessite pas ce type d'information. Cette méthode restera donc applicable quels que soient les systèmes utilisés. La méthode repose sur le concept très simple d'*union* ou d'*intersection* des listes de documents restitués par les systèmes.

Le principe de l' union que nous utilisons est similaire à celui de la théorie des ensembles. De manière formelle, soit D un document et (L_1, L_2) deux listes de documents. L' union des deux listes s'écrit de la manière suivante :

$$D \in (L_1 \cup L_2) \Leftrightarrow (D \in L_1) \vee (D \in L_2) \tag{5.1}$$

L' union regroupe les documents sélectionnés par les systèmes. Les doublons qui sont éventuellement présents à l'issue de l' union sont supprimés. L' union devrait donc favoriser le *rappel*, dans la mesure où tout document retrouvé par l'un ou l'autre des systèmes est retrouvé après la fusion par union.

L' intersection quant à elle permet de regrouper les documents restitués en commun par les systèmes. Elle se formalise de la manière suivante :

$$D \in (L_1 \cap L_2) \Leftrightarrow (D \in L_1) \wedge (D \in L_2) \tag{5.2}$$

où D est un document et (L_1, L_2) sont des listes de documents restitués par deux systèmes différents. L' intersection quant à elle devrait favoriser la *précision*, car un document ne fera partie de la liste restituée (après la fusion) que si les deux systèmes le retrouvent dans leur liste[1].

Ce principe sert de base dans la méthode que nous proposons dans la section suivante.

5.3 Fusion adaptative des systèmes

La technique de fusion que nous avons présentée dans la section précédente est basée uniquement sur les principes simples d' union et d' intersection. La deuxième technique que nous proposons consiste à fusionner les systèmes en intégrant de nouveaux paramètres tels que le chevauchement des listes de documents, et la difficulté de la requête. Nous essayons de voir si la fusion peut être adaptée en fonction de ces paramètres liés aux requêtes et aux systèmes.

Les travaux présentés dans [DHMO04] montrent que les requêtes pour lesquelles une grande proportion de systèmes ont échoué sont celles qui ont les plus faibles taux de précision (leur rappel est aussi faible, mais pas nécessairement parmi les plus faibles). Nous nous basons sur ce principe pour établir une typologie des requêtes afin d'en étudier l'impact sur les résultats de la recherche. Cette typologie constitue un premier paramètre que nous analysons dans nos expérimentations. Notons qu'aucune phase d'apprentissage n'est nécessaire pour établir la typologie des requêtes. Cette typologie est déterminée de manière empirique.

Les documents restitués par différents systèmes sont différents. Les travaux présentés dans [Lee97] indiquent que le chevauchement entre des ensembles de documents différents (nombre de documents communs entre des ensembles de documents) est un

[1] Cela correspond à l'*effet de choeur* que nous avons présenté dans le chapitre 3

facteur très important à considérer dans la fusion. Nous nous intéressons à ce taux de chevauchement dans nos travaux afin de vérifier les résultats présentés dans [Lee97] qui montrent que plusieurs systèmes ont tendance à retrouver les mêmes documents pertinents mais retournent différents documents non pertinents. Le taux de chevauchement est calculé pour des paires de listes de documents selon la formule suivante.

$$Couv_P = \frac{nb\_doc\_pert\_en\_commun \ * 2}{nb\_doc\_pert\_S_1 \ + nb\_doc\_pert\_S_2} \tag{5.3}$$

Dans la formule 5.3, le taux de chevauchement est calculé pour deux systèmes S_1 et S_2. $Couv_P$ représente le taux de chevauchement (ou de couverture) des listes de documents pertinents, restitués par les SRI S_1 et S_2. Nous nous limitons dans nos travaux au taux de chevauchement des documents pertinents, afin de mesurer le degré de ressemblance des systèmes. Le taux de chevauchement varie entre 0 et 1. Nous avons choisi de calculer le taux de chevauchement pour les N meilleurs systèmes. Nous analysons ensuite l'impact du taux de chevauchement en fonction des typologies de requêtes, en appliquant les techniques de fusion par union et par intersection.

5.4 Évaluation

Les 2 techniques de fusion précédentes sont évaluées en se basant sur les collections de TREC 2002 et 2003 utilisées dans la sous-tâche *détection de passages* de la tâche "nouveauté" (*Novelty*). Les mesures que nous avons choisi d'étudier sont le rappel, la précision, et la F-mesure. Pour chacune de ces deux campagnes de TREC (2002 et 2003), les systèmes sont classés par performance décroissante.

Nous présentons dans cette section les différentes expérimentations que nous avons menées afin d'évaluer les stratégies de fusion que nous proposons. Dans la section 5.4.1 nous nous intéressons à la fusion systématique des systèmes. Dans la section 5.4.2, nous détaillons les expérimentations effectuées dans le cadre de la fusion adaptative des systèmes basée sur des critères de chevauchement des documents et la difficulté des requêtes.

5.4.1 Expérimentations sur la fusion *systématique* des systèmes

Cette section présente les différentes expérimentations que nous avons effectuées pour évaluer la stratégie de fusion *systématique* des systèmes. La section 5.4.1.1 présente les différentes spécificités des données que nous utilisons. Dans la section 5.4.1.2, nous analysons l'impact de la fusion systématique des systèmes sur les performances du meilleur système. Dans la section 5.4.1.3, nous proposons deux stratégies de sélection des systèmes que nous utilisons dans la fusion et nous comparons les résultats obtenus par chacune de ces stratégies.

5.4.1.1 Analyse préalable

L'objectif de cette analyse préalable est de détailler les caractéristiques des collections et des systèmes que nous utilisons dans nos expérimentations.

Dans la figure 5.1, nous analysons la distribution des performances des systèmes (en terme de rappel) ayant participé à la campagne d'évaluation TREC Novelty de 2002 et de 2003. Nous utilisons à cet effet une représentation des données sous forme de boîte à moustaches [Tuk77].

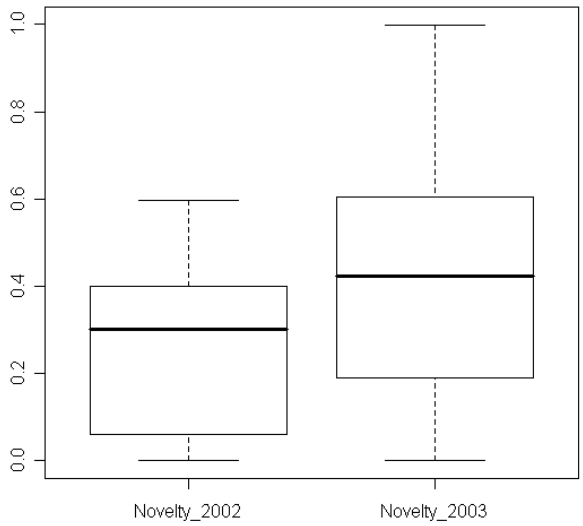

FIG. 5.1 – Diagramme en boîte représentant la répartition du rappel pour les campagnes TREC/Novelty 2002 et 2003

Une boîte à moustaches permet de représenter une distribution de données. La boîte à moustaches utilise 5 valeurs qui résument des données : le minimum, les 3 quartiles Q1, Q2 (médiane), Q3, et le maximum. Les quartiles jouent un rôle important dans l'interprétation de la boîte à moustaches. La médiane Q2 divise la série en deux groupes d'effectifs égaux. Le premier quartile divise le groupe de données situé en dessous de la médiane en deux groupes d'effectifs égaux. Le troisième quartile divise quand à lui le groupe situé au delà de la médiane en deux groupes de taille égale. Les quartiles représentent 25% (Q1), 50% (Q2) et 75% (Q3) des données analysées.

Dans la figure 5.1, la première boîte à moustaches décrit la répartition des mesures de rappel pour la campagne de 2002. Les valeurs de rappel qui sont présentées correspondent à la moyenne que chaque système obtient sur l'ensemble des requêtes pour

lesquelles le système est évalué. En 2002 la valeur maximale de rappel est 0,597 (et de 0,999 en 2003) et 75% des valeurs de rappel sont inférieures à 0,4. En 2003, plus de la moitié des systèmes obtiennent une valeur de rappel supérieure à 0,42.

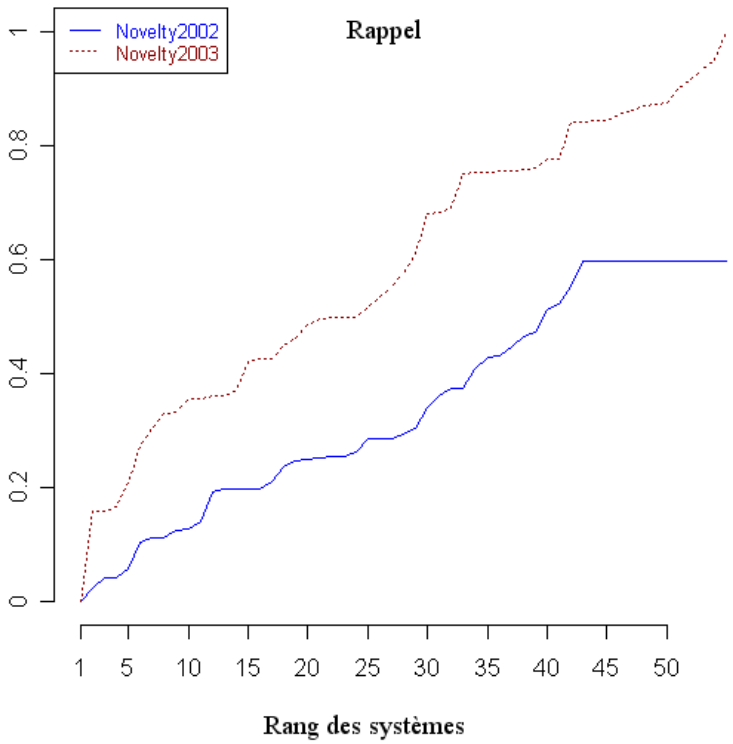

FIG. 5.2 – Variation entre le rappel du meilleur système et les autres systèmes

La figure 5.2 complète l'interprétation des boîtes à moustaches. Elle représente les variations entre le système qui obtient la plus grande valeur de rappel et les autres systèmes. Dans cette figure, les différences de rappel entre deux systèmes de rangs consécutifs ne sont pas grandes. Un certain nombre de paliers sont toutefois à remarquer aussi bien pour 2002 que pour 2003. Par exemple, en 2003, une grande différence de rappel existe entre le premier système et le deuxième système. Entre le cinquième et le sixième système en 2003, une variation importante de rappel est observée soit 0,066 (idem pour 2002 avec une variation entre le cinquième et sixième système de 0,049).

Dans les figures 5.3 et 5.4, on constate que quelques systèmes obtiennent des valeurs de précision en dessous de 0,5 en 2003. Il s'agit des systèmes "atypiques" suivants :

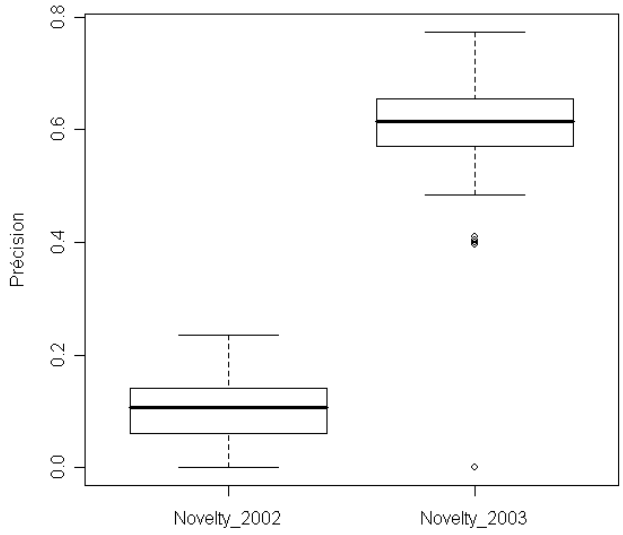

FIG. 5.3 – Répartition des valeurs initiales de précision

Système	Précision
lexiclone03	0,484
ISIALL03	0,411
ISIRAND03	0,41
umbcrun2	0,405
umbcrun3	0,4
umbcrun1	0,396

Le système ISINONE03 obtient quant à lui une valeur de précision nulle en 2003.

Les performances des systèmes en 2003 sont très élevées par rapport à 2002, comme on peut le constater dans les figures 5.3 et 5.4 pour la précision et la F-mesure. Dans la figure 5.5, la différence entre la précision du meilleur système et celle des autres systèmes est inférieure à 0,2 pour les 40 premiers systèmes de 2002 et 2003. Les plus grandes variations sont dues aux systèmes "atypiques" présentés dans le paragraphe précédent. De plus, les différences entre les performances des meilleurs systèmes et ceux qui obtiennent de mauvais résultats sont grandes. Par exemple la différence entre le meilleur et le dernier système en termes de rappel en 2003 est de 0,999. La figure 5.5 représente les variations entre la précision du meilleur système et celles des autres systèmes étudiés. Les variations de la précision sont assez faibles et sont inférieures à 0,2 pour les 40 meilleurs systèmes.

L'analyse préalable que nous venons de mener nous indique que les performances

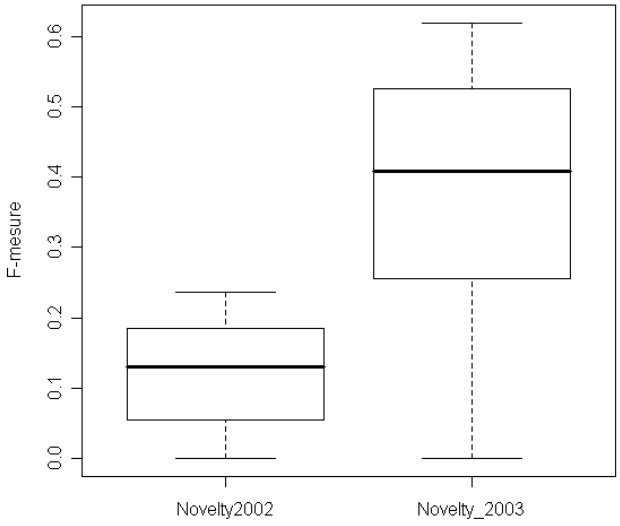

FIG. 5.4 – Répartition des valeurs initiales de F-mesure

entre les meilleurs systèmes sont proches, indépendamment des stratégies de recherche utilisées par les systèmes. La fusion de données est une technique que nous avons envisagée pour améliorer les résultats des meilleurs systèmes. Nous focalisons notre analyse dans les sections suivantes sur les 10 meilleurs systèmes de chaque année. Ce choix s'explique par le fait que fusionner un système ayant obtenu de très bonnes performances et un système ayant des performances faibles s'avère souvent bénéfique pour le plus mauvais système, mais pas toujours pour le meilleur. Nous étudions dans la section suivante l'impact de nos stratégies de fusion sur les N meilleurs systèmes.

5.4.1.2 Impact de la fusion systématique sur les performances du meilleur système

Nous nous intéressons à l'impact de la fusion par union et par intersection sur les performances des systèmes. Nous procédons dans un premier temps à la sélection des systèmes à fusionner, puis nous analysons l'impact de la fusion des systèmes sélectionnés sur les performances du meilleur système. Les systèmes utilisés pour la fusion sont sélectionnés à partir de leurs performances en termes de F-mesure. Le meilleur système est celui qui obtient la plus grande F-mesure. La F-mesure est calculée selon la formule 5.4.

$$F - mesure = \frac{2 * Rappel * Précision}{Rappel + Précision} \tag{5.4}$$

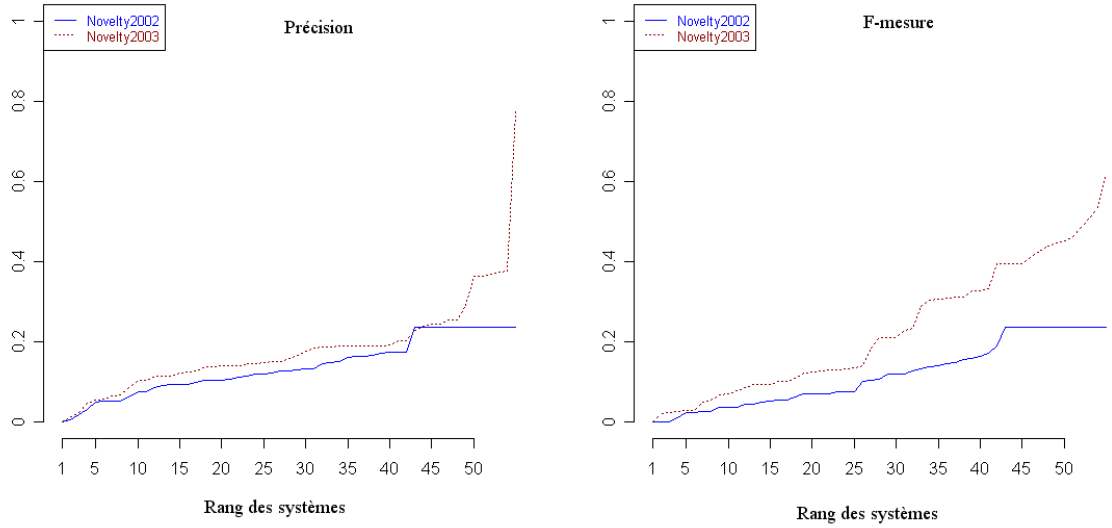

FIG. 5.5 – Variation entre la précision et la F-mesure du meilleur système et les autres systèmes

Nous choisissons la F-mesure pour ne pas favoriser le rappel ou la précision car la F-mesure leur fait jouer un rôle symétrique. Nous focalisons notre attention sur la dernière colonne des tableaux 5.1 et 5.2 dans laquelle les systèmes sont ordonnés par F-mesure décroissante.

Dans les tableaux 5.1 et 5.2, nous donnons pour chacun des 10 meilleurs systèmes les valeurs de rappel, précision et F-mesure qu'ils ont obtenues officiellement à TREC, ordonnés par F-mesure décroissante. Ainsi, en 2002, le système ayant obtenu les meilleures performances est le système thunv3 avec 0,237 (F-mesure) avec un rappel de 0,404 et une précision de 0,204. En 2003, le meilleur système (THUIRnv0315) obtient une F-mesure de 0,619 (rappel 0,792 et précision 0,597).

Dans les tableaux 5.3 et 5.4, les chiffres en gras indiquent des fusions qui n'apportent aucune amélioration par rapport aux résultats du meilleur système. Les chiffres soulignés indiquent une amélioration des résultats par rapport au meilleur système. Les chiffres en italique quant à eux correspondent à une dégradation des performances par rapport aux résultats du meilleur système.

Dans le tableau 5.3 par exemple, la fusion par union de thunv1 et thunv3 n'apporte aucune amélioration par rapport au rappel initial de thunv3. Par contre, la fusion par intersection de thunv1 et thunv3 dégrade les performances de thunv3 (en termes de rappel et F-mesure) et améliore la précision initiale de thunv3. Ce tableau nous indique

	Rappel	Précision	F-mesure
Thunv3	0,404	0,204	0,237
Thunv1	0,341	0,229	0,236
Thunv2	0,341	0,236	0,236
Thunv4	0,335	0,216	0,226
CIIR02tfkl	0,556	0,141	0,213
CIIR02tfnew	0,556	0,141	0,213
pircs2N01	0,486	0,161	0,211
pircs2N02	0,486	0,161	0,211
pircs2N03	0,4	0,184	0,199
pircs2N04	0,4	0,184	0,199

TAB. 5.1 – Performances initiales des 10 meilleurs systèmes (F-mesure) en 2002

	Rappel	Précision	F-mesure
THUIRnv0315	0,792	0,597	0,619
ISIDSCm203	0,832	0,534	0,597
Ulowa03Nov01	0,696	0,636	0,594
THUIRnv0.11	0,726	0,606	0,593
MeijiHilF13	0,84	0,52	0,589
MeijiHilF14	0,84	0,52	0,589
Ulowa03Nov02	0,637	0,649	0,568
THUIRnv0312	0,665	0,624	0,564
THUIRnv0313	0,637	0,633	0,552
THUIRnv0314	0,628	0,632	0,548

TAB. 5.2 – Performances initiales des 10 meilleurs systèmes (F-mesure) en 2003

	Union			Intersection		
Systèmes fusionnés	Rappel	Précision	F-mesure	Rappel	Précision	F-mesure
thunv3 initial	(0,404)	(0,204)	(0,237)	(0,404)	(0,204)	(0,237)
ciir02tfkl-thunv3	0,597	*0,138*	*0,212*	*0,365*	0,213	0,24
ciir02tfnew-thunv3	0,597	*0,138*	*0,212*	*0,365*	0,213	0,24
pircs2n01-thunv3	0,543	*0,147*	*0,211*	*0,349*	0,226	0,24
pircs2n02-thunv3	0,543	*0,147*	*0,211*	*0,349*	0,226	0,24
pircs2n03-thunv3	0,496	*0,16*	*0,216*	*0,31*	0,232	*0,221*
pircs2n04-thunv3	0,496	*0,16*	*0,216*	*0,31*	0,232	*0,221*
thunv1-thunv3	**0,404**	**0,204**	**0,237**	*0,341*	0,228	*0,235*
thunv2-thunv3	**0,404**	**0,204**	**0,237**	*0,341*	0,234	*0,235*
thunv3-thunv4	**0,404**	**0,204**	**0,237**	*0,335*	0,216	*0,226*

TAB. 5.3 – Performance des meilleurs systèmes en 2002 fusionnés avec Thunv3

que la fusion par union de thunv3 avec les autres versions du même système (thunv1, thunv2, et thunv4) ne modifie pas les performances obtenues par thunv3 pour les 3 mesures utilisées. En analysant le tableau 5.1, on peut faire l'hypothèse que les réponses des 4 versions du système *thunv* constituent des sous-ensembles des réponses du système thunv3. La conséquence de cette hypothèse est que l' union de thunv3 avec les autres versions du système correspond exactement aux résultats de thunv3.

En effet, l' intersection de thunv3 avec thunv1 correspond en fait au résultat de thunv1 (les valeurs après intersection de rappel, précision, et F-mesure sont les mêmes que celles de thunv1). La même remarque peut être faite avec les autres versions de *thunv*.

Les résultats sont sensiblement différents lorsque l'on considère les systèmes des autres participants. Par exemple, la fusion par union ou intersection de thunv3 avec ciir02tfnew ou avec pircs2n01, montre bien que les réponses de ces systèmes sont différentes.

De façon générale, le tableau 5.3 montre que lorsque nous considérons les 10 meilleurs systèmes, l'impact de la fusion par union sur la F-mesure est négatif (de 0 à -10,5%). Cependant, l'impact de la fusion par intersection sur la F-mesure est positif (+1,3% pour les 4 premières fusions) lorsque la fusion n'est pas effectuée avec des versions du système *thunv*.

Les mêmes expérimentations ont été réalisées pour la campagne de 2003.

	Union			Intersection		
Systèmes fusionnés	Rappel	Précision	F-mesure	Rappel	Précision	F-mesure
thuirnv0315 initial	(0,792)	(0,597)	(0,619)	(0,792)	(0,597)	(0,619)
isidscm203-thuirnv0315	0,921	*0,528*	*0,618*	*0,703*	0,615	*0,595*
meijihilf13-thuirnv0315	0,938	*0,517*	*0,613*	*0,694*	0,618	*0,592*
meijihilf14-thuirnv0315	0,938	*0,517*	*0,613*	*0,694*	0,618	*0,592*
thuirnv0311-thuirnv0315	0,82	*0,585*	0,623	*0,698*	0,622	*0,588*
thuirnv0312-thuirnv0315	0,804	*0,594*	0,623	*0,654*	0,632	*0,559*
thuirnv0313-thuirnv0315	**0,792**	**0,597**	**0,619**	*0,638*	0,636	*0,552*
thuirnv0314-thuirnv0315	**0,792**	**0,597**	**0,619**	*0,628*	0,635	*0,549*
thuirnv0315-uiowa03nov01	0,797	*0,595*	0,62	*0,691*	0,639	*0,593*
thuirnv0315-uiowa03nov02	0,793	**0,597**	**0,619**	*0,636*	0,65	*0,568*

TAB. 5.4 – Performance des meilleurs systèmes en 2003 fusionnés avec THUIRnv0315

Le tableau 5.4 indique que pour 2003 la fusion avec THUIRnv0315 dégrade les performances de la F-mesure lors de l' intersection. Nous remarquons aussi que les versions du système *THUIRnv* sont assez différentes les unes des autres, comme en témoignent leurs performances initiales (cf. tableau 5.2). De la même manière qu'en 2002, les résultats obtenus par les versions thuirnv0313 et thuirnv0314 sont des sous-ensembles de ceux de thuirnv0315. La fusion par union améliore le rappel du système

thuirnv0315 tout en dégradant ses performances en termes de précision, tandis que la fusion par intersection dégrade le rappel tout en améliorant la précision. La précision des systèmes fusionnés est supérieure à celle du meilleur système, alors que le rappel dégrade les performances de thuirnv0315.

Nous avons choisi les meilleurs systèmes uniquement sur la base de leur performance en termes de F-mesure. Cependant, le système qui obtient la meilleure F-mesure n'est pas forcément celui qui obtient des performances maximales pour le rappel et la précision (et inversement). Nous détaillons dans la section suivante les expérimentations que nous avons effectuées ainsi que les 2 stratégies que nous avons choisies pour sélectionner, pour chaque mesure, les 10 meilleurs systèmes à utiliser lors des fusions.

5.4.1.3 Analyse globale de la fusion des systèmes

L'analyse que nous présentons à présent est dite globale car elle n'est pas centrée sur les performances du meilleur système.

Le tableau 5.5 dresse un récapitulatif des meilleurs systèmes par mesure de performance pour les campagnes TREC Novelty 2002 et 2003.

	Systèmes	Rappel	Systèmes	Précision	Systèmes	F-mesure
2002	nttcslabnvr2	0,597	thunv2	0,236	thunv3	0,237
	UIowa02Nov4	0,574	thunv1	0,229	thunv1	0,236
	CIIR02tfkl	0,556	thunv4	0,216	thunv2	0,236
	CIIR02tfnew	0,556	thunv3	0,204	thunv4	0,226
	nttcslabnvp	0,539	thunv5	0,187	CIIR02tfkl	0,213
	dumbrun	0,493	pircs2N03	0,184	CIIR02tfnew	0,213
	pircs2N01	0,486	pircs2N04	0,184	pircs2N01	0,211
	pircs2N02	0,486	pircs2N05	0,184	pircs2N02	0,211
	ntu1	0,472	cmu02t300rAs	0,174	pircs2N03	0,199
	ntu2	0,469	pircs2N01	0,161	pircs2N04	0,199
2003	ISIALL03	0,999	NLPR03n1w3	0,774	THUIRnv0315	0,619
	MeijiHilF13	0,84	NLPR03n1w2	0,761	ISIDSCm203	0,597
	MeijiHilF14	0,84	NLPR03n1f2	0,751	UIowa03Nov01	0,594
	ISIDSCm203	0,832	NLPR03n1f1	0,726	THUIRnv0311	0,593
	THUIRnv0315	0,792	clr03n1d	0,718	MeijiHilF13	0,589
	THUIRnv0311	0,726	clr03n1n3	0,716	MeijiHilF14	0,589
	UIowa03Nov01	0,696	clr03n1t	0,709	UIowa03Nov02	0,568
	ISIDSm203	0,67	NLPR03n1w1	0,705	THUIRnv0312	0,564
	THUIRnv0312	0,665	clr03n1n2	0,688	THUIRnv0313	0,552
	MeijiHilF11	0,643	ccsummeoqr	0,67	THUIRnv0314	0,548

TAB. 5.5 – Performances initiales des 10 meilleurs systèmes en 2002 et 2003 par mesure de performance

On peut remarquer dans le tableau 5.5 que plusieurs versions de différents systèmes obtiennent les meilleures performances en 2002 et en 2003. Par exemple, 3 systèmes et leurs versions obtiennent les 10 plus grandes valeurs de précision en 2003 (il s'agit des versions des systèmes NLPPR03n et clr03n1, ainsi que le système ccsummeoqr). Cette remarque peut être faite pour l'ensemble des mesures présentées.

Rappelons que nous nous situons dans le contexte de la campagne d'évaluation TREC. Nous disposons donc, à l'issue de l'évaluation des systèmes ayant participé aux tâches Novelty2002 et Novelty2003, d'un ensemble de mesures de performances permettant de classer les systèmes ainsi que la connaissance du nombre de documents pertinents dans la collection utilisée. Les méthodes que nous évaluons dans cette section doivent être ramenées à un contexte expérimental.

Nous avons présenté dans le tableau 5.5 les performances des 10 meilleurs systèmes pour chaque année de TREC analysée. Nous savons d'autres part que les systèmes n'obtiennent pas les mêmes performances pour un ensemble de requêtes donné. Nous supposons alors qu'il est possible de sélectionner les 10 meilleurs systèmes en fonction de leurs performances. La figure 5.6 compare les performances obtenues si les 10 meilleurs systèmes sont sélectionnés et lorsque l'ensemble des systèmes est sélectionné. L'objectif est de quantifier l'amélioration obtenue par le choix des 10 meilleurs systèmes. Dans la sous-section 5.4.1.4, nous poursuivons notre analyse sur l'impact de la fusion en présentant les résultats que nous obtenons lorsque nous n'avons aucune connaissance *a priori* sur les performances des systèmes (cf. tableau 5.10).

Dans la figure 5.6, pour chacune des années de TREC que nous avons analysée, on remarque que pour chaque mesure (Rappel, Précision et F-mesure), le choix des 10 meilleurs systèmes permet d'obtenir des performances moyennes supérieures à celles obtenues en considérant l'ensemble des systèmes participants. La figure 5.6 illustre les variations de performance qu'il ya entre les 10 meilleurs systèmes et l'ensemble des systèmes. Ces variations permettent, d'une certaine manière, de mesurer l'homogénéité des systèmes (en terme de performance). Lorsque les variations de performance sont grandes entre les meilleurs systèmes et les autres, il apparaît pertinent d'utiliser les meilleurs systèmes dans nos expérimentations. Par exemple, en 2002, le rappel moyen obtenu par les 10 meilleurs systèmes est de 0,5228 contre 0,3356 lorsque l'ensemble des systèmes est pris en compte (d'où une différence de 0,1872 par rapport à la moyenne sur l'ensemble des systèmes). En 2003, la différence entre le rappel moyen des 10 meilleurs systèmes (0,7703) et l'ensemble des systèmes (0,4083) atteint 0,360. Pour la précision, la différence est de 0,067 en 2002 et de 0,129 en 2003.

Nous proposons de sélectionner les 10 meilleurs systèmes de chaque année, afin d'appliquer 2 stratégies de fusion. Les 10 systèmes qui sont sélectionnés sont combinés deux à deux, et le résultat de la fusion par union et par intersection est analysé. Nous présentons ci-dessous les résultats obtenus pour chaque type de fusion (union, intersection), chaque mesure (rappel, précision, F-mesure), et chaque année (2002, 2003). Nous avons

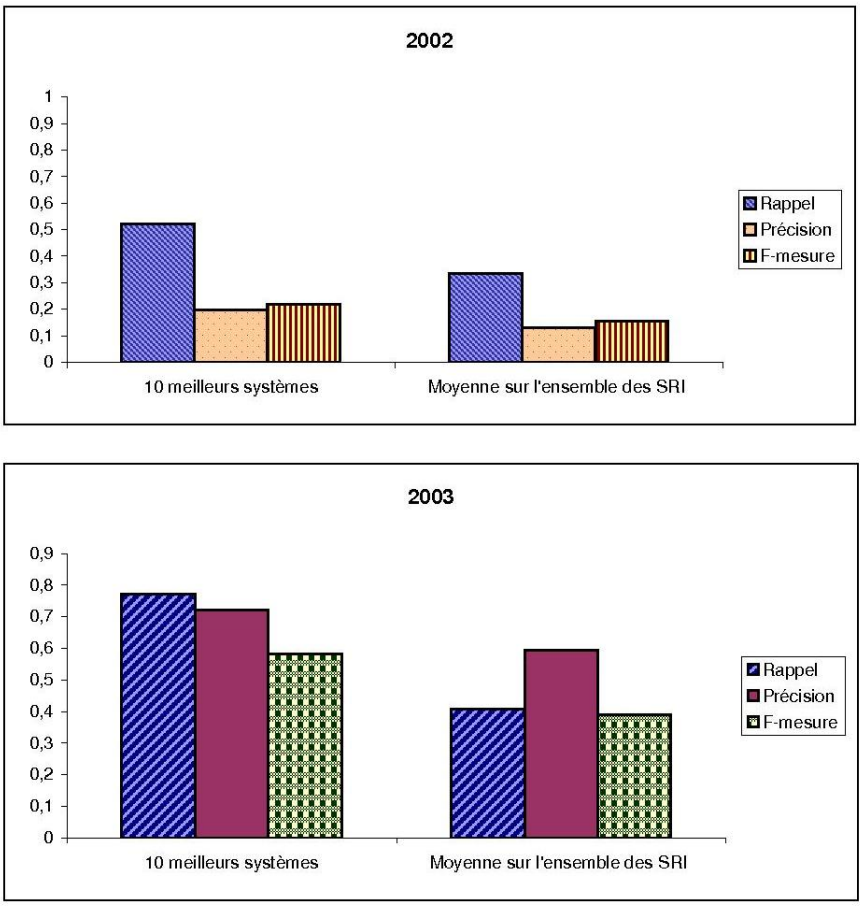

FIG. 5.6 – Comparaison des performances des 10 meilleurs systèmes par rapport aux performances moyennes de l'ensemble des SRI

adopté deux stratégies différentes pour la sélection des 10 meilleurs systèmes.

Dans la première stratégie (nommée stratégie1), nous utilisons la F-mesure comme base de sélection. Les 10 systèmes qui obtiennent les meilleures F-mesures sont sélectionnés.

Dans le cas où l'on ne souhaite pas faire jouer un rôle symétrique aux mesures de rappel et précision, nous proposons une deuxième stratégie (nommée stratégie2). Dans la stratégie2, les 10 meilleurs systèmes sont sélectionnés en fonction de la mesure que l'on souhaite analyser. Par exemple, on choisira les 10 systèmes qui obtiennent le meilleur rappel lorsque nous nous intéressons au rappel. Dans le cas où la précision est utilisée, la sélection des systèmes se base sur les 10 meilleurs systèmes en termes de précision. Ainsi, les 10 meilleurs systèmes sélectionnés pour le rappel ne sont pas les mêmes que ceux sélectionnés pour la précision dans la stratégie2, alors que dans la stratégie1 les

systèmes sélectionnés par la valeur de leur F-mesure, sont aussi utilisés dans le calcul du rappel et de la précision.

Notons toutefois que lorsque la F-mesure est souhaitée, la stratégie1 et la stratégie2 sont équivalentes car elles permettent de sélectionner les mêmes systèmes.

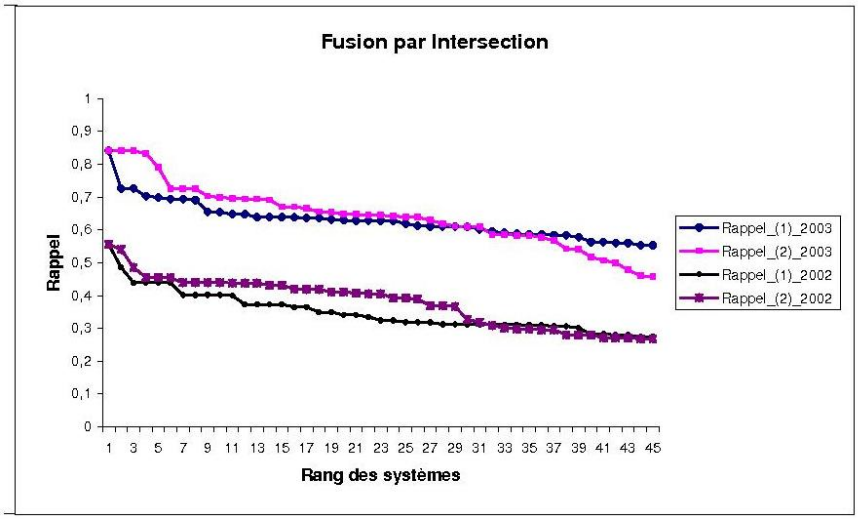

FIG. 5.7 – Comparaison des mesures de rappel obtenues par chaque stratégie de fusion pour la fusion par **intersection**

La figure 5.7 compare pour chacune des collections utilisées les résultats obtenus (en termes de rappel) avec les 2 stratégies que nous proposons. Nous retenons pour chaque stratégie les 45 valeurs, issues de la combinaison des 10 meilleurs systèmes deux à deux, de rappel que nous comparons. Dans la figure 5.7, les valeurs en abscisses correspondent au classement de la valeur de rappel considérée par rapport aux autres valeurs de rappel. Nous remarquons pour l'année 2002 que sur l'ensemble des résultats de la fusion, la stratégie 2 permet d'obtenir de meilleurs résultats que la stratégie 1. La stratégie1 obtient cependant de meilleurs résultats que la stratégie2 pour les 15 dernières combinaisons en 2003 (13 dernières combinaisons en 2002). Le point de basculement de la tendance des courbes se situe lors de la fusion des systèmes thunv1/thunv2 (stratégie1), et des systèmes nttcslabnvr2/ntu1 (stratégie2).

La figure 5.8 est interprétée de la même manière que la figure 5.7. Dans la figure 5.8, les résultats qui sont présentés correspondent à ceux que nous obtenons après application de la fusion par union. Nous remarquons une nette séparation des courbes pour chacune des années. Les 9 premiers résultats obtenus avec la stratégie2 en 2003 obtiennent un rappel égal à 1. La stratégie 2 reste toujours la meilleure stratégie à utiliser.

Le tableau 5.6 indique la valeur de l'amélioration moyenne obtenue lors des fusions,

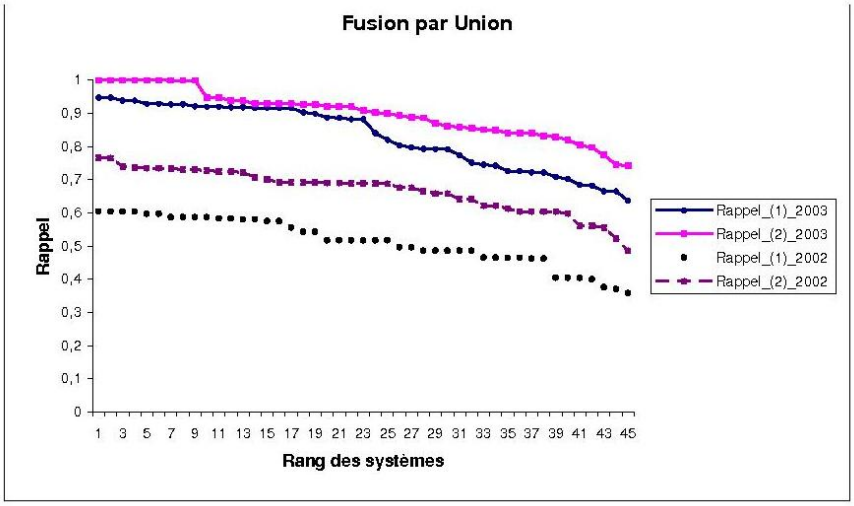

FIG. 5.8 – Comparaison des mesures de rappel obtenues par chaque stratégie de fusion pour la fusion par **union**

et complète l'analyse des courbes 5.8 et 5.7. Dans ce tableau, pour la stratégie1 (sélection des meilleurs systèmes basée sur la mesure F), nous comparons la moyenne sur l'ensemble des combinaisons 2 à 2 des systèmes (45 couples de systèmes formés par la fusion des 10 systèmes sélectionnés avec la stratégie1) avec les performances du meilleur système (rappel du système étant le meilleur par rapport à la mesure F) et avec la moyenne du rappel obtenu par les systèmes utilisés isolément. La moyenne des systèmes utilisés isolément revient à calculer le rappel moyen obtenu par les 10 systèmes sélectionnés par la stratégie1. Les performances de la stratégie2, qui sélectionne les meilleurs systèmes en termes de rappel sont comparées d'une part avec celles du meilleur système (sélectionné de la même façon) et avec celles obtenues en moyenne par les systèmes sélectionnés pris séparément.

		2002	2003
Stratégie1	Meilleur système simple	0,404	0,792
	Moyenne systèmes simples	0,4305	0,7293
	Moyenne fusion 2 à 2	0,5133	0,8305
Stratégie2	Meilleur système simple	0,597	0,999
	Moyenne systèmes simples	0,5228	0,7703
	Moyenne fusion 2 à 2	0,6679	0,8997

TAB. 5.6 – Valeurs moyennes de rappel pour la fusion par union des systèmes 2 à 2

Dans le tableau 5.6, pour la stratégie1, on remarque que le rappel moyen obtenu par les systèmes simples (0,4305) est supérieur au rappel du meilleur système sélectionné

par la stratégie1 (0,404). Cela s'explique par le fait que le système détecté comme étant le meilleur avec la stratégie1 (Thunv3) est classé en 12ème position par rapport au rappel des autres systèmes. Dans ce cas, en appliquant la fusion par union sur les systèmes sélectionnés avec la stratégie1, on obtient une amélioration du rappel moyen de 19,23% par rapport au rappel moyen des systèmes simples. En 2003, le rappel moyen des systèmes simples est inférieur au rappel du meilleur système avec la stratégie1 (ce système est classé en 5ème position par rapport au rappel des autres systèmes).

Pour la stratégie2, le meilleur système obtient un rappel supérieur au rappel moyen des systèmes simples. On constate alors en 2003 que le rappel moyen à l'issue de la fusion des 10 meilleurs systèmes est inférieur au rappel du meilleur système. La conclusion que l'on peut tirer est que les 9 autres meilleurs systèmes retrouvent tous des sous-ensembles de l'ensemble des documents pertinents que le meilleur système restitue (le deuxième meilleur système obtient un rappel de 0,84 contre 0,999 pour le meilleur système).

La stratégie 2 permet d'améliorer en moyenne les résultats plus que la fusion des meilleurs systèmes sur la base de la mesure F (27,75% contre 19,23% en 2002, et 16,8% contre 13,9% en 2003). Ce résultat n'était pas attendu : en effet, intuitivement, il paraît plus facile d'améliorer les performances de "mauvais" systèmes que de "bons" systèmes (même si l'amélioration de "mauvais" systèmes par combinaison avec de meilleurs systèmes n'a pas vraiment d'intérêt).

Les mêmes expérimentations sont reproduites pour la précision. Dans la figure 5.9, les résultats obtenus montrent une faible différence entre les performances de la stratégie1 et de la stratégie2 en 2002 à l'issue de la fusion. Cette faible différence s'explique par le fait que 80% des systèmes sélectionnés à travers leur valeur de précision et ceux sélectionnés grâce à leur valeur de F-mesure sont identiques (cf. tableau 5.5). De plus, l' intersection marque un accord entre les systèmes sur les documents retrouvés. Dans la figure 5.10, l'analyse précédente est plus mitigée pour 2002, les performances obtenues étant sensiblement égales.

		2002	2003
Stratégie1	Meilleur système simple	0,204	0,597
	Moyenne systèmes simples	0,1857	0,5951
	Moyenne fusion 2 à 2	0,2169	0,6396
Stratégie2	Meilleur système simple	0,236	0,774
	Moyenne systèmes simples	0,1959	0,7218
	Moyenne fusion 2 à 2	0,2268	0,7598

TAB. 5.7 – Valeurs moyennes de précision pour la fusion par intersection des systèmes 2 à 2

Le tableau 5.7 contient le même type de comparaisons que la tableau 6, mais en termes de précision.

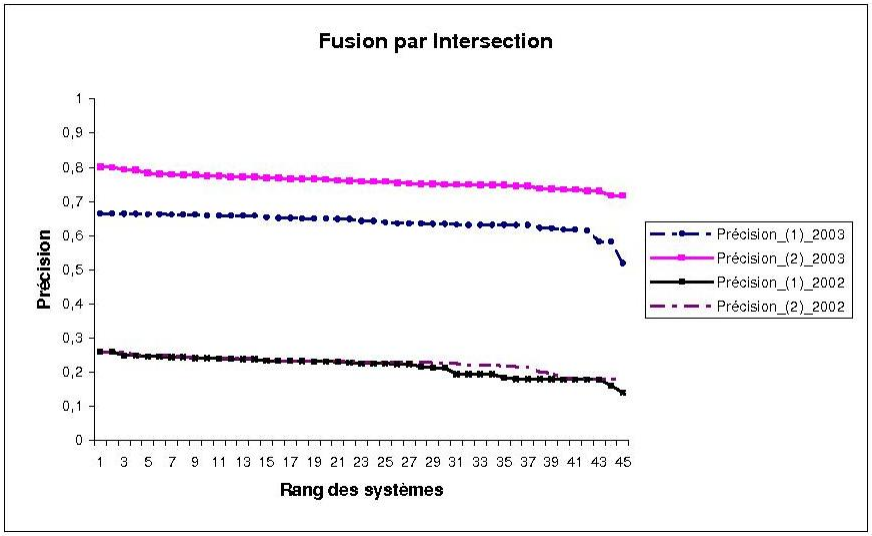

FIG. 5.9 – Comparaison des mesures de précision obtenues par chaque stratégie de fusion pour la fusion par **intersection**

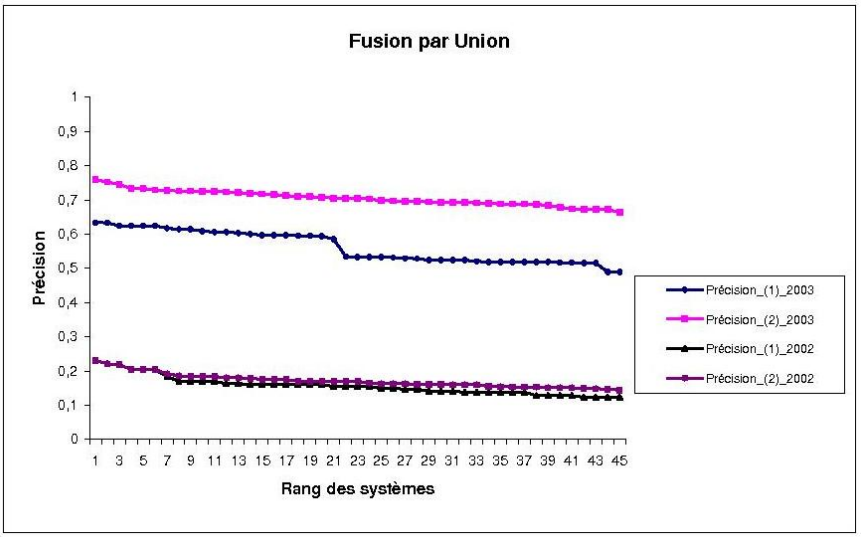

FIG. 5.10 – Comparaison des mesures de précision obtenues par chaque stratégie de fusion pour la fusion par **union**

5.4.1.4 Discussions

Les résultats des expérimentations que nous avons présentées montrent quels sont les impacts des fusions par union et par intersection. Nous avons montré qu'il était

possible, par cette simple stratégie d'améliorer fortement le rappel et donc de constituer un ensemble important de documents liés à un sujet donné.

Parallèlement, nous avons montré qu'il était aussi possible d'améliorer fortement la précision. En 2002, la fusion du meilleur système avec un des meilleurs systèmes en termes de mesure F permet d'augmenter la précision de 15% (en 9% 2003). En se basant sur les meilleurs systèmes en termes de précision, l'augmentation est portée à 1% en 2002 et 20% en 2003.

Il faut cependant noter que le rappel et la précision sont des mesures corrélées.

Stratégie1

			Rappel	Précision	F mesure
2002	Meilleur système	thunv3	0,404	0,204	0,237
	Union	thunv1-thunv2	*0,37*	0,231	0,244
	Intersection	ciir02tfkl-thunv1	*0,324*	0,238	0,241
		ciir02tfnew-thunv1	*0,324*	0,238	0,241
2003	Meilleur système	THUIRnv315	0,792	0,597	0,619
	Union	thuirnv0311-thuirnv0315	0,82	*0,585*	0,623
		thuirnv0312-thuirnv0315	0,804	*0,594*	0,623
	Intersection	meijihilf13-thuirnv0311	*0,703*	0,615	*0,595*

Tab. 5.8 – Comparaison entre les performances du meilleur système et les meilleures combinaisons de la stratégie1 de fusion systématique

En fonction de la stratégie utilisée, nous comparons les résultats que nous obtenons avec ceux du meilleur système. Dans le tableau 5.8, les valeurs des meilleurs systèmes, sans fusion, sont représentées dans la première ligne de chaque année. Les valeurs en italique indiquent une dégradation par rapport aux performances du meilleur système, et les valeurs soulignées une amélioration par rapport aux performances initiales. Dans ce tableau, le meilleur système est déterminé par la stratégie1 c'est à dire que le meilleur système a la plus grande valeur de F-mesure.

Nous constatons que le choix de la stratégie1 permet d'améliorer les performances du meilleur système en terme de F-mesure pour les 2 années étudiées, lors de la fusion par union. Dans le tableau 5.8, les meilleurs résultats sont obtenus en 2002 en fusionnant des versions de thunv3 (en l'occurrence thunv1 et thunv2) avec d'autres systèmes. Par exemple pour la précision, la fusion par union de thunv2 et thunv1 permet d'améliorer les performances de thunv3 de 13,23%. Ce résultat n'était pas attendu car l' union est censé améliorer le rappel. Dans le cas présent, l' union de thunv1 et thunv2 dégrade le rappel (-8,42%). Pour l' intersection par contre, la précision de thunv3 est améliorée de 16,67%. En 2003 par contre, l' union avec thuirnv315 permet d'améliorer son rappel de 1,52% et 3,54% ; L' intersection permet d'améliorer la précision de 3,18%. Les résultats obtenus en 2002 montrent par exemple que la précision joue un rôle plus important que le rappel dans les améliorations obtenues par la fusion par union, tandis qu'en 2003 le rappel permet d'obtenir des améliorations. Cette situation s'explique par le fait que la stratégie1 est basée sur la F-mesure qui fait intervenir le rappel et la précision dans son calcul. Nous nous focalisons donc sur les résultats obtenus en terme de F-mesure. On

peut alors remarquer que pour chacune des 2 années étudiées, la fusion par union avec la stratégie1 permet d'améliorer la F-mesure. La fusion par union permet d'obtenir une amélioration de la F-mesure de l'ordre de 2,95% en 2002 et de 0,64% en 2003.

Stratégie2

		Systèmes	Rappel	Systèmes	Précision
2002	Meilleur système	nttcslabnvr2	0,597	thunv2	0,236
	Union	nttcslabnvr2-ntu2	0,767	thunv1-thunv2	*0,231*
	Intersection	ciir02tfkl-ciir02tfnew	*0,556*	pircs2n03-thunv2	0,26
				pircs2n04-thunv2	0,26
				pircs2n05-thunv2	0,26
2003	Meilleur système	ISIALL03	0,999	NLPR03n1w3	0,774
	Union	isiall03-isidscm203	1	nlpr03n1w2-nlpr03n1w3	*0,759*
		isiall03-isidsm203	1		
		isiall03-thuirnv0311	1		
		isiall03-thuirnv0312	1		
		isiall03-thuirnv0315	1		
		isiall03-uiowa03nov01	1		
	Intersection	isiall03-meijihilf13	*0,84*	clr03n1t-nlpr03n1w3	0,802
		isiall03-meijihilf14	*0,84*		
		meijihilf13-meijihilf14	*0,84*		

TAB. 5.9 – Comparaison entre les performances du meilleur système et les meilleures combinaisons de la stratégie2 de fusion systématique

Le choix de la stratégie2 permet d'obtenir les résultats que nous présentons dans le tableau 5.9. Nous ne présentons pas la F-mesure dans ce tableau car les résultats sont les mêmes que ceux de la stratégie1 (les mêmes systèmes sont choisis pour la fusion).

La lecture du tableau peut se faire selon le rappel ou selon la précision. On remarque que deux versions d'un même système obtiennent le meilleur rappel en 2002 (CIIR02tfkl et CIIR02tfnew). Nous remarquons dans le tableau que la majorité des meilleurs résultats obtenus avec la stratégie2 fusionne des versions de systèmes avec le meilleur système. Il apparaît dans ce tableau que la fusion par union permet d'améliorer le rappel du meilleur système (28,47% en 2002, et 0,1% en 2003) et la fusion par intersection améliore la précision du meilleur système (10,2% d'amélioration en 2002 contre 3,62% en 2003). Dans le tableau 5.9, deux situations particulières se produisent. En 2002, lors de la fusion par intersection, la précision du meilleur système (thunv2) est améliorée de 10,17% lorsque thunv2 est fusionné avec 3 versions du système pircs2n (pircs2n03, pircs2n04 et pircs2n05). Il en est de même en 2003 avec la fusion par union. Le rappel du meilleur système (ISIALL03) est augmenté lorsqu'il est fusionné avec les 2 versions du système isids (isidscm203 et isidsm3203) ou avec les 3 versions du système thuirnv (thuirnv0311, thuirnv0312 et thuirnv0315). Dans ces deux situations, la question que l'on peut se poser est comment choisir à priori le bon couplage ? Dans le

cas présent, nous considérons qu'il n'ya pas un "bon" couplage, mais plusieurs ("bons" couplages).

Pour aller plus loin dans la réflexion liée à la question précédente, on peut se demander si l'amélioration des performances est due à la stratégie de sélection des systèmes ou à l'application de la fusion elle même. Il est établi que la fusion par union améliore le rappel et que la fusion par intersection a des effets positifs sur la précision. Les stratégies de fusion que nous avons proposées sont applicables dans un cadre expérimental bien défini, nous offrant les éléments nécessaires à la sélection des meilleurs systèmes. Dans des conditions d'exploitation réelles, sans connaissance *a priori* sur la manière de sélectionner les systèmes, il est toutefois possible d'appliquer les techniques de fusion par union et par intersection. Nous donnons à titre d'exemple les résultats obtenus lors de l'application de la fusion, sans sélection préalable des meilleurs systèmes.

		Rappel	Précision	F-mesure
2002	Union	0,5028 (49,82%)	*0,1124* (-12,84%)	0,1653 (6,8%)
	Intersection	*0,1735* (-48,3%)	0,1700 (31,86%)	*0,1356* (-12,39%)
	Moyenne sans fusion	0,3356	0,1289	0,1547
2003	Union	0,6073 (48,74%)	*0,5800* (-2,16%)	0,5075 (30,47%)
	Intersection	*0,2114* (-48,22%)	0,6792 (14,58%)	*0,2519* (-35,24%)
	Moyenne sans fusion	0,4083	0,5928	0,3890

TAB. 5.10 – Mesure de l'impact de la fusion sur les performances initiales

Dans le tableau 5.10, nous avons appliqué la fusion par union et intersection sur l'ensemble des systèmes. Les systèmes sont couplés deux à deux lors de la fusion, indépendamment de leurs performances initiales. La moyenne obtenue à l'issue de la fusion est alors comparée à la moyenne des performances officielles obtenues par les systèmes lors de leur participation à TREC. Cette comparaison nous permet de quantifier l'apport des stratégies de fusion par union et par intersection. Les résultats du tableau 5.10 confirment que la fusion par union augmente le rappel, tandis que la fusion par intersection augmente la précision. En 2002, l'amélioration obtenue sur le rappel par la fusion par union s'élève à 49,82% (48,74% en 2003). La précision moyenne sans application de la fusion subit quant à elle une diminution de -12,84% en 2002 (-2,16% en 2003). En ce qui concerne la précision, l'amélioration est de 31,86% en 2002 et de 14,58% en 2003 lorsque la fusion par intersection est appliquée. Le rappel subit une dégradation avec la fusion par intersection de l'ordre de -48,3% en 2002 et -48,22% en 2003. L' union et l' intersection sont des méthodes qui sont utilisables en situation réelle, sans conditions supplémentaires.

Dans les conditions expérimentales que nous avons définies dans la section 5.4.1.3, nous avons montré à travers nos expérimentations que la stratégie2 est plus efficace que la stratégie1, quel que soit le type de fusion appliqué. La stratégie2 est meilleure que la stratégie1 pour le rappel de l'ordre de 24,2% en 2002, et de 6,66% en 2003. Pour

la précision, la stratégie2 obtient de meilleurs résultats que la stratégie1 de l'ordre de 1,17% en 2002 contre 18,57% en 2003. Nous proposons de retenir la stratégie2 pour la suite des expérimentations que nous présentons dans la section 5.4.2.

5.4.2 Expérimentation sur la fusion adaptative des systèmes

L'application des techniques de fusion par union ou intersection permet d'obtenir des améliorations de performance par rapport aux performances initiales des systèmes. Dans les expérimentations précédentes, les requêtes ne jouent pas un rôle particulier dans la fusion. Nous introduisons donc un paramètre supplémentaire dans la fusion (la requête) car les systèmes n'obtiennent pas les mêmes performances pour toutes les requêtes. Nous pensons qu'en fonction de certaines caractéristiques des requêtes, il est possible de déterminer de manière plus fine la technique de fusion à appliquer. Les caractéristiques que nous mettons en avant sont d'une part la typologie des requêtes (section 5.4.2.1) (requêtes faciles, intermédiaires, difficiles), et le chevauchement des listes de documents restitués par les SRI en fonction de la typologie des requêtes (section 5.4.2.2).

5.4.2.1 Impact de la typologie des requêtes

Nous établissons une typologie de requêtes afin d'étudier son impact sur les résultats de la fusion. Nous avons déterminé 3 catégories de requêtes. La première catégorie regroupe les requêtes que nous considérons comme difficiles. Ce sont les requêtes pour lesquelles la précision moyenne des systèmes est faible. La deuxième catégorie regroupe les requêtes dites faciles, c'est à dire les requêtes pour lesquelles la moyenne des précisions est élevée. Enfin, la dernière catégorie contient les requêtes dites intermédiaires, pour lesquelles la moyenne des précisions est la plus proche de la moyenne des précisions moyennes. Dans la figure 5.11, nous donnons le diagramme en boîtes des performances moyennes que les SRI ont obtenues pour les requêtes de chaque année.

La répartition des performances dans ce diagramme nous indique qu'un grand nombre de systèmes obtiennent des performances moyennes pour l'ensemble des requêtes. Par exemple en 2002, 50% des requêtes sont satisfaites avec une précision moyenne comprise entre 0,05 et 0,18. Nous avons déterminé manuellement le nombre de requêtes et nous avons choisi 4 requêtes faciles, 4 requêtes intermédiaires et 4 requêtes difficiles. Ainsi, notre analyse sur la typologie des requêtes se déroule sur un ensemble de 12 requêtes. Les 4 requêtes faciles sont celles pour lesquelles la précision moyenne des SRI est parmi les plus grandes. Les 4 requêtes difficiles quant à elles sont choisies par rapport à la faible précision moyenne obtenue sur l'ensemble des systèmes. Les requêtes intermédiaires sont les 4 requêtes qui obtiennent la plus faible différence avec la moyenne des précisions moyennes. Les 4 requêtes dites intermédiaires sont composées des 2 requêtes les plus plus proches en valeur inférieure de la moyenne des moyennes, et des 2 requêtes les plus proches en valeur supérieure à la moyenne des moyennes.

Le tableau 5.11 récapitule pour chaque année, les 14 requêtes que nous avons utilisées dans notre étude. Nous avons choisi d'utiliser la deuxième stratégie de fusion pour

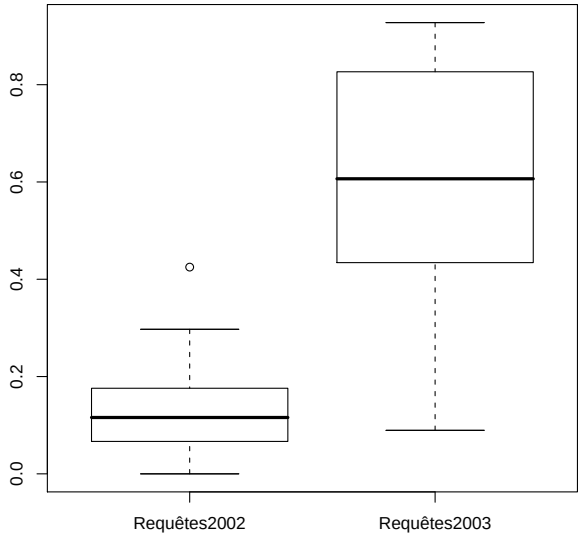

FIG. 5.11 – Répartition des performances moyennes des systèmes (précision moyenne) pour les requêtes de chaque année

	requêtes faciles	requêtes intermédiaires	requêtes difficiles
2002	368, 397 355, 358	405, 342 416, 409	377, 420 312, 432
2003	15, 26 27, 31	16, 21 28, 39	19, 20 25, 48

TAB. 5.11 – Typologie des requêtes utilisées

analyser l'impact de la typologie des requêtes sur les résultats.

Dans le tableau 5.12, nous présentons les performances moyennes obtenues par les 10 systèmes sélectionnés par la stratégie2. Chaque valeur du tableau correspond à la moyenne sur l'ensemble des requêtes faciles des performances moyennes des 10 meilleurs systèmes sélectionnés. Dans ce tableau, les valeurs soulignées correspondent à une amélioration des performances par rapport aux performances obtenues par les systèmes lorsqu'aucune fusion n'est appliquée. Les valeurs en italiques indiquent des performances moins bonnes à l'issue de la fusion que celles obtenues sans fusion. En prenant par exemple la mesure du rappel, le tableau 5.12 montre que lorsque la fusion par union est appliquée, on obtient une différence de rappel de 0,2465 en 2002 et de 0,1698 en 2003. Lorsque la fusion par intersection est utilisée, la différence de rappel est de -0,016 en 2002 et de -0,1463 en 2003. Pour la précision, l' intersection permet d'obtenir de

		Année	
	Type Fusion	2002	2003
Rappel	Sans fusion	0,4573	0,6648
	Union	0,7038	0,8346
	Intersection	*0,4413*	*0,5185*
Précision	Sans fusion	0,4870	0,9853
	Union	*0,4293*	*0,9852*
	Intersection	0,5429	0,9874
F-mesure	Sans fusion	0,4263	0,7387
	Union	0,4343	0,8194
	Intersection	0,4295	*0,6480*

Tab. 5.12 – Analyse des performances moyennes des systèmes pour les requêtes faciles par année

meilleures performances (différence de 0,0559 en 2002 et de 0,0021 en 2003). Pour la F-mesure, la fusion par union entraîne de meilleurs résultats (0,4343 en 2002 et 0,8194 en 2003) que les résultats sans fusion (0,4263 en 2002 et 0,7387 en 2003). L' intersection permet d'obtenir une amélioration de la F-mesure de 0,75% en 2002 et une dégradation de la F-mesure de -12,28% en 2003. La conclusion que l'on peut tirer est que l' union améliore la F-mesure pour les 2 années étudiées.

Les tableaux 5.13, et 5.14 se lisent comme le tableau 5.12. Dans le tableau 5.13 le taux de précision moyenne est inférieur à la précision initiale lorsque la fusion par union est appliquée en 2002 et en 2003 pour les requêtes intermédiaires. En 2003, la fusion par union entraîne une dégradation de la précision moyenne de -1,67% (contre -14,94% en 2002).

		Année	
	Type Fusion	2002	2003
Rappel	Sans fusion	0,3365	0,7493
	Union	0,6200	0,8871
	Intersection	0,3386	*0,6144*
Précision	Sans fusion	0,2028	0,7240
	Union	*0,1725*	*0,7072*
	Intersection	0,2512	0,7682
F-mesure	Sans fusion	0,2102	0,6235
	Union	0,2276	0,6352
	Intersection	*0,2081*	*0,6084*

Tab. 5.13 – Analyse des performances moyennes des systèmes pour les requêtes intermédiaires par année

Le tableau 5.14 contient les résultats obtenus après l'application de la fusion sur les

requêtes difficiles. Dans le cas des requêtes difficiles, les résultats vont dans le même sens que ceux présentés dans le tableau 5.12. Pour la précision, nous obtenons un résultat qui n'était pas attendu. En effet, l'application de la fusion par union en 2002 permet d'obtenir de meilleures performances en terme de précision comparé aux résultats sans fusion. Une explication pourrait être que plusieurs couples de systèmes, parmi les couples de fusion des 10 meilleurs sélectionnés, retrouvent les mêmes documents non pertinents en commun, et des documents pertinents différents pour les 4 requêtes difficiles que nous avons utilisées.

		Année	
	Type Fusion	2002	2003
Rappel	Sans fusion	0,1873	0,8348
	Union	0,3694	0,9462
	Intersection	*0,1509*	*0,7655*
Précision	Sans fusion	0,0190	0,2203
	Union	0,0203	*0,2008*
	Intersection	0,0217	0,2546
F-mesure	Sans fusion	0,0305	0,2057
	Union	0,0309	*0,1886*
	Intersection	*0,0236*	*0,2288*

TAB. 5.14 – Analyse des performances moyennes des systèmes pour les requêtes difficiles par année

Les résultats que nous obtenons confirment que la fusion par union a un impact positif sur le rappel, et la fusion par intersection sur la précision. Nous résumons dans le tableau 5.15 le pourcentage d'amélioration que l'on obtient lorsque nous comparons les performances moyennes obtenues par union et par intersection aux performances initiales moyennes par type de requêtes. Nous avons utilisé à cet effet la F-mesure comme mesure de performance. Les conclusions que nous tirons sont donc valables pour les 2 stratégies que nous avons proposées.

Type requêtes	Année	Union	Intersection
Faciles	2002	1,88%	0,75%
	2003	10,92%	-12,28%
Difficiles	2002	1,31%	-22,62%
	2003	-8,31%	11,30%

TAB. 5.15 – Récapitulatif des pourcentages d'amélioration des performances moyennes obtenues. La mesure utilisée est la F-mesure

Dans le tableau 5.15, on constate que quel que soit le type de requête et l'année, la fusion par union permet d'obtenir une amélioration des performances pour les requêtes faciles (1,88% en 2002 et 10,92% en 2003). Pour les requêtes difficiles, aucune conclusion ne peut être prise car la fusion par union améliore la F-mesure initiale (sans fusion) en

2002 de 1,31%, et dégrade la F-mesure iniale en 2003 (-8,31%). Les résultats obtenus par la fusion par intersection montrent quant à eux une dégradation des performances en 2002 (-22,62%) et une amélioration de la F-mesure initiale en 2003 (11,30%).

Nous récapitulons dans le tableau suivant 5.16 les améliorations obtenues pour les requêtes faciles et difficiles lorsque le rappel et la précision sont utilisés comme mesures de performance. Dans ce tableau, nous utilisons le rappel lorsqu'il s'agit de la fusion par union, et la précision lorsque la fusion par intersection est utilisée.

Type requêtes	Année	Union	Intersection
Faciles	2002	53,90%	11,48%
	2003	25,54%	0,21%
Difficiles	2002	97,12%	14,21%
	2003	13,34%	15,60%

TAB. 5.16 – Récapitulatif des pourcentages d'amélioration des performances moyennes obtenues. Le rappel est utilisé pour l' union et la précision pour l'intersection

Nous notons dans le tableau 5.16 que les améliorations vont de 0,21% à 53,90% (fusion par union et par intersection confondues) pour les requêtes faciles. Pour les requêtes difficiles, l'amélioration va de 13,34% à 97,22%. Il semble d'après ce tableau plus facile d'améliorer les performances des requêtes difficiles que celles des requêtes faciles. Ce résultat n'était pas attendu : en effet, intuitivement, il paraît plus facile d'améliorer les performances des requêtes "faciles" que des requêtes "difficiles".

Nous souhaitons poursuivre l'analyse sur l'impact du nombre de documents pertinents dans la fusion. Nous proposons dans la section suivante d'analyser l'impact du taux de chevauchement des listes de documents restitués lors de la fusion, la fusion étant réalisée en fonction des typologies des requêtes que nous avons présentées dans cette section.

5.4.2.2 Impact du taux de chevauchement

Le taux de chevauchement dans les listes restituées par les systèmes permet de mesurer à quel degré les systèmes s'accordent sur les documents qu'ils jugent pertinents. L'analyse que nous effectuons a pour but de montrer les liens existant entre la typologie des requêtes et le degré de chevauchement des listes restituées dans la fusion. Nous utiliserons le terme fusion dans cette section pour faire référence à la fusion par intersection qui permet de mesurer le degré d'accord entre les listes restituées par les systèmes. Dans le tableau 5.17, nous récapitulons pour chaque année et chaque type de requêtes, le nombre de documents jugés pertinents. Rappelons que la colonne NB_doc_pert représente pour une requête, le nombre de documents pertinents définis comme tels par les juges de TREC (cf. chapitre 4). Notons que ces jugements de pertinence pour une même requête peuvent être différents selon les juges et l'instant du jugement, ainsi que selon la difficulté de la requête.

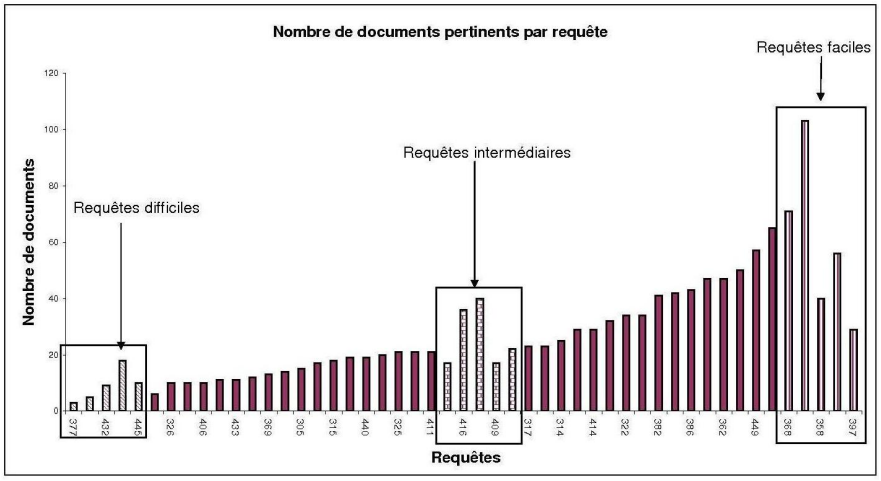

FIG. 5.12 – Répartition du nombre de documents pertinents par requête en 2002

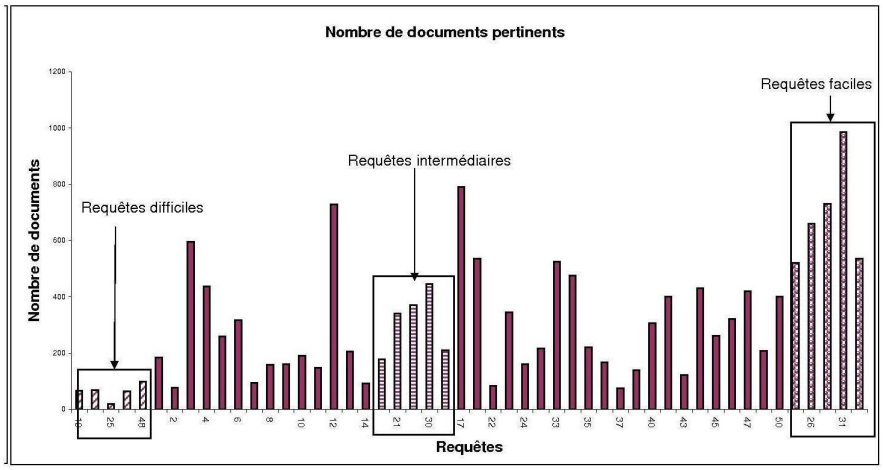

FIG. 5.13 – Répartition du nombre de documents pertinents par requête en 2003

Dans les figures 5.12 et 5.13, nous présentons pour chaque type de requête le nombre de documents pertinents définis par les juges en 2002 et 2003. On remarque que pour chacune des années, le groupe des requêtes faciles contient la requête possédant le plus grand nombre de documents pertinents, et inversement pour les requêtes difficiles.

Dans le tableau 5.17 les requêtes difficiles possèdent en moyenne moins de documents pertinents que les requêtes faciles. En 2002 par exemple, les requêtes faciles comptent en moyenne 60,75 documents pertinents contre 8,75 documents en moyenne pour les requêtes difficiles ; la différence est plus nette en 2003 (le nombre de documents pertinents varie entre 19 et 985) où pour l'ensemble des requêtes difficiles on dispose en moyenne

	2002		2003	
	requêtes	NB_doc_ pert	requêtes	NB_doc_pert
Faciles	355	103	15	522
	358	40	26	661
	368	71	27	730
	397	29	31	985
	Moyenne	**60,75**	Moyenne	**724,5**
Intermédiaires	405	40	16	179
	409	17	21	340
	416	36	28	371
	342	17	30	445
	Moyenne	**27,5**	Moyenne	**311,2**
Difficiles	377	3	19	62
	312	5	20	69
	420	18	25	19
	432	9	48	98
	Moyenne	**8,75**	Moyenne	**62**

TAB. 5.17 – Nombre de documents pertinents par requête en 2002 et 2003

de 62 documents pertinents et de près de 725 documents pertinents pour les requêtes faciles.

Le taux de chevauchement est calculé pour chaque type de requêtes et chaque année. Un taux de chevauchement égal à 1 indique que les deux systèmes pour lesquels ce taux est calculé, retrouvent les mêmes documents pertinents.

De même, un taux de chevauchement qui est égal à 0 indique que les systèmes ne retrouvent aucun document en commun. Dans les figures 5.14 et 5.15 nous donnons une représentation du taux de chevauchement pour chaque requête facile et difficile.

La figure 5.14 indique pour chaque requête facile le taux de chevauchement obtenu lorsque la fusion est effectuée sur cette catégorie de requêtes. On remarque en moyenne que le taux de chevauchement en 2002 est supérieur à celui de 2003 pour les requêtes faciles. Pour la requête 358 par exemple, plus de 75% des systèmes fusionnés obtiennent un taux de chevauchement compris entre 0,96 et 1, c'est à dire que les systèmes retrouvent un grand nombre de documents pertinents communs, relatifs à la requête 358.

Pour la requête 312 par exemple (figure 5.15), un taux de chevauchement de 1 est obtenu pour 8,79% des systèmes fusionnés ; seulement 2,19% des systèmes obtiennent un taux de chevauchement de 0,67. La grande majorité des systèmes (89,02%) n'ont aucun document pertinent en commun pour la requête 312 qui comprend 5 documents pertinents (cf. tableau 5.17).

Afin d'avoir une idée globale sur les relations qui existent entre la typologie des requêtes et le taux de chevauchement, nous présentons dans les figures 5.16 et 5.17 la répartition des taux de chevauchement en fonction du type de requête.

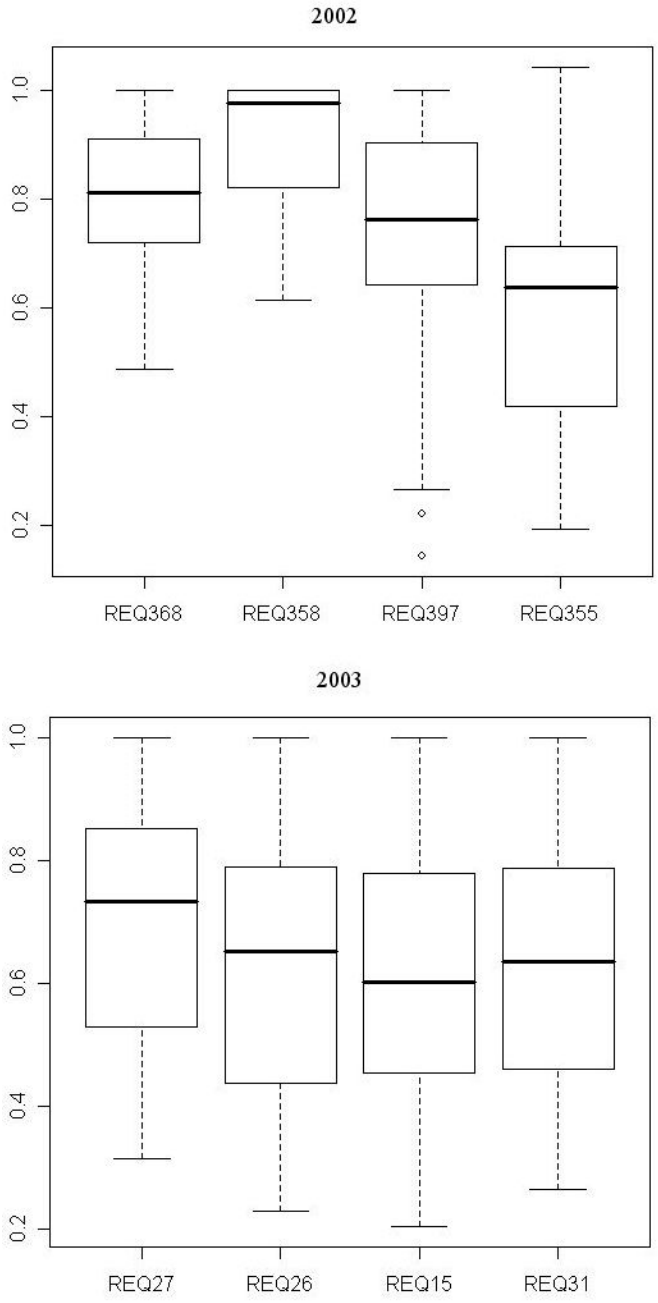

FIG. 5.14 – Taux de chevauchement des requêtes faciles

Dans ces deux figures, la fusion est réalisée pour chaque couple de systèmes, et le taux de chevauchement correspondant est calculé. L'allure des courbes montrent le taux de chevauchement pour chaque type de requête, et chaque année. Les abscisses

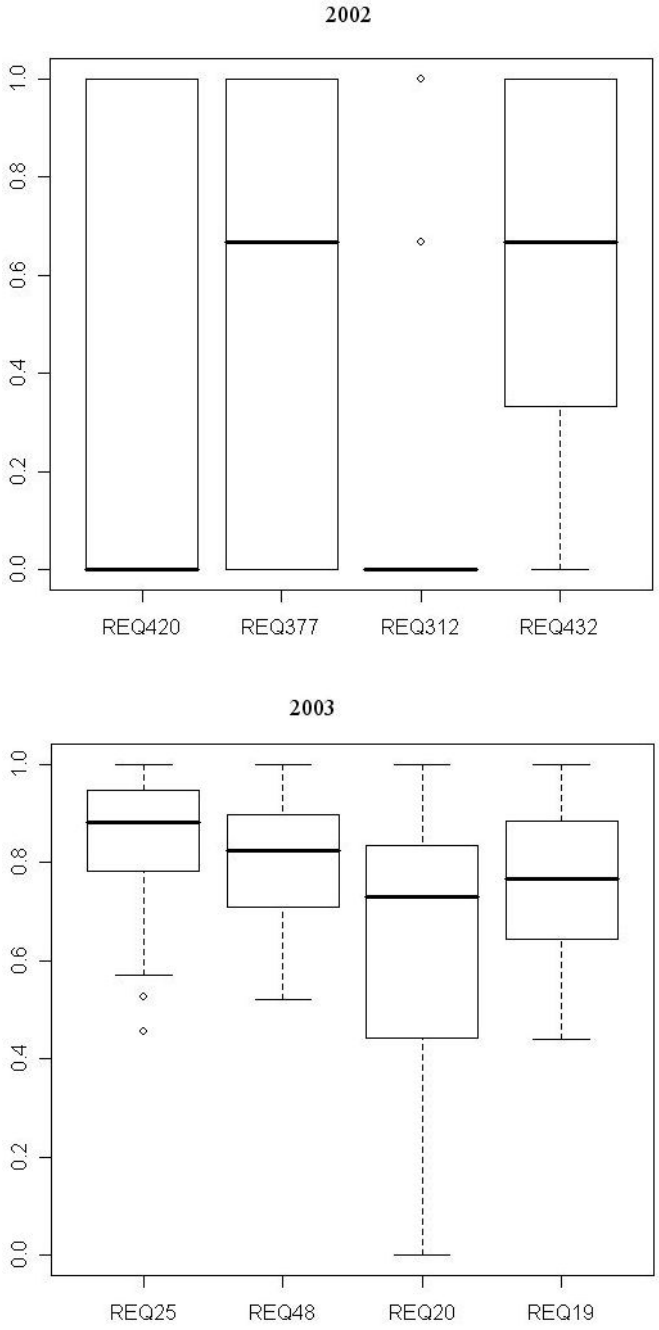

FIG. 5.15 – Taux de chevauchement des requêtes difficiles

correspondent aux rang des systèmes fusionnés (les différentes fusions sont ordonnées par taux de chevauchement décroissant), et les ordonnées indiquent les différents taux

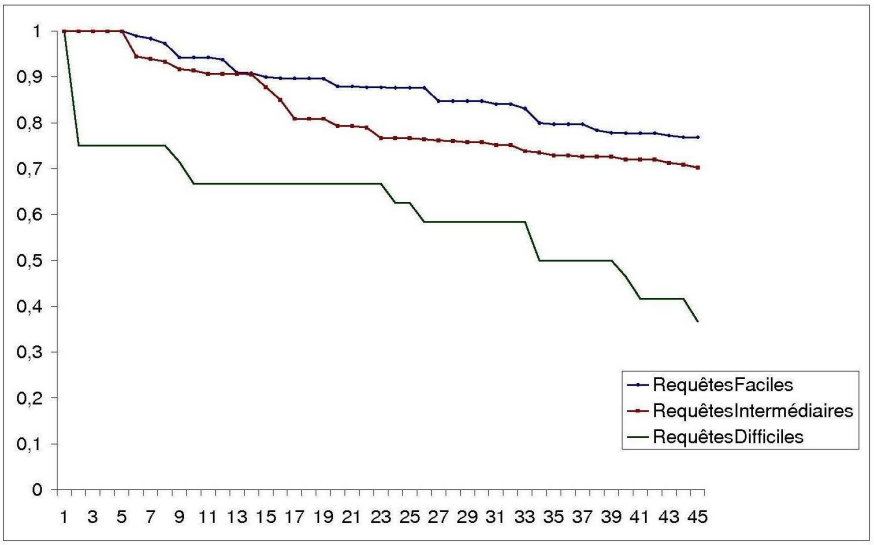

FIG. 5.16 – Représentation des taux de chevauchement moyens en 2002, par type de requête

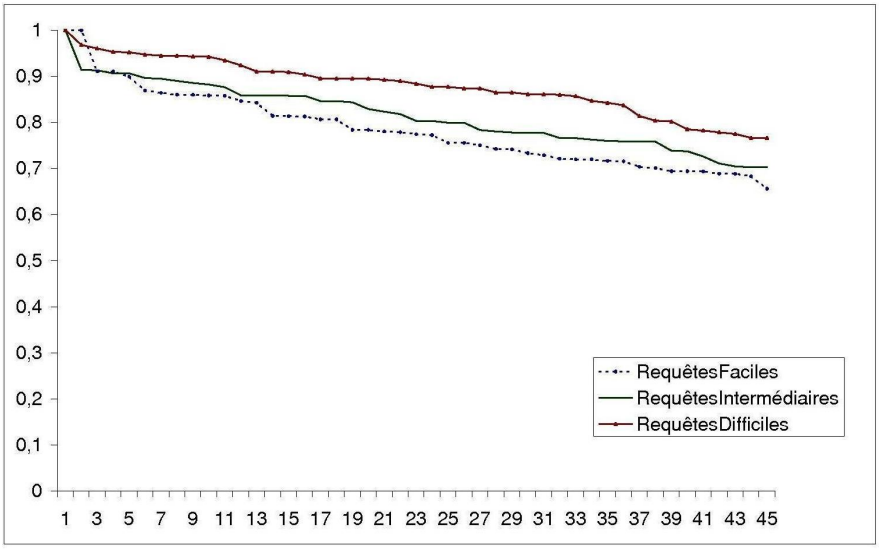

FIG. 5.17 – Représentation des taux de chevauchement moyens en 2003, par type de requête

de chevauchement obtenus.

L'observation de ces 2 figures (5.16, et 5.17) nous montre qu'en 2002, la courbe représentant les taux de chevauchement pour les requêtes faciles est située au dessus

de la courbe représentant le taux de chevauchement moyen pour les requêtes difficiles. Cela signifie qu'en moyenne le taux de chevauchement est plus important pour les requêtes faciles que pour les requêtes difficiles. En 2002, les couples de systèmes fusionnés retrouvent plus de documents pertinents en commun pour les requêtes faciles. En 2003, les systèmes retrouvent plus de documents pertinents pour les requêtes difficiles.

La figure 5.18 quantifie la répartition des différents taux de chevauchement par type de requête et par année. Dans cette figure, les abscisses correspondent aux différents intervalles de chevauchement, et les ordonnées indiquent le nombre de couples de systèmes obtenant un taux de chevauchement dans l'intervalle considéré. Par exemple, l'abscisse 0,1 correspond à l'intervalle [0..0,9999] et l'ordonnée correspondante comptabilise le nombre de couples de systèmes obtenant un taux de chevauchement appartenant à cet intervalle. En 2003, aucun couple de systèmes fusionnés n'obtient un taux de chevauchement à l'abscisse 0,1, quel que soit le type de requêtes traité. Par contre, pour la même année 2003, 13 couples de systèmes obtiennent un taux de chevauchement compris entre 0,5 et 0,59999 pour les requêtes faciles. En abscisse, la valeur 1 compte le nombre de fusions ayant un taux de chevauchement de 1. La lecture de la figure 5.18 nous indique que pour les taux de chevauchement inférieurs à 0,8, les couples de systèmes fusionnés restituent plus de documents pertinents communs pour les requêtes faciles que pour les requêtes difficiles. Les taux de chevauchement supérieurs à 0,7 montrent par exemple en 2002 que les systèmes retrouvent beaucoup plus de documents pertinents en commun pour les requêtes faciles que pour les requêtes difficiles. En 2003, les couples de systèmes fusionnés retrouvent plus de documents pertinents en commun pour les requêtes difficiles comparé au nombre de documents communs retrouvés pour les requêtes faciles.

5.5 Conclusion

Dans ce chapitre, nous nous sommes intéressés aux méthodes de fusion de systèmes [FS94] qui s'appuient sur la variabilité des performances des systèmes afin de proposer des résultats issus de la combinaison de plusieurs systèmes. Ces méthodes nécessitent en général la connaissance des jugements de pertinence indiquant les documents pertinents à retrouver. Cependant, cette information est rarement disponible. Les moteurs du web par exemple ne la fournissent pas en même temps que les documents.

Dans ce chapitre, nous avons proposé deux stratégies de fusion simples basées sur l' union et l' intersection, et ne nécessitant pas ce type d'information. Une méthode simple permettant d'augmenter le rappel consiste à appliquer une fusion par union des résultats obtenus. Il est bien établi que la fusion par union ne peut qu'augmenter le rappel (ou le maintenir constant si le couple de systèmes fusionnés retrouve les mêmes documents pertinents). De la même façon, L' intersection permet d'augmenter la précision : un document retrouvé par un couple de systèmes a plus de chances d'être pertinent, car il correspond à un accord entre les systèmes. Il est cependant bien connu qu'appliquer une méthode qui augmente le rappel dégrade généralement la précision et inversement. Nous avons cherché dans ce chapitre à mesurer les effets de la fusion par union et intersection sur le rappel et la précision.

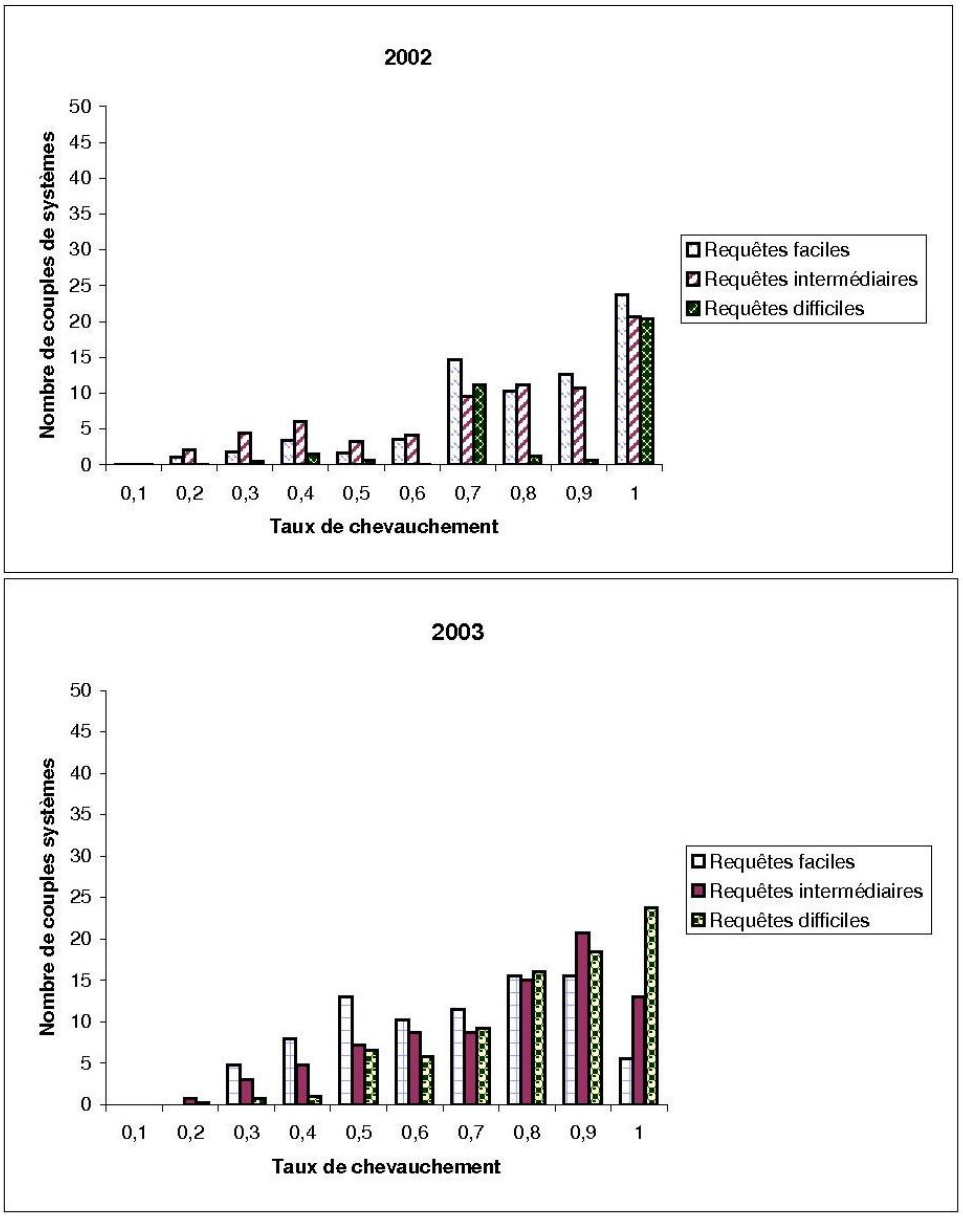

FIG. 5.18 – Répartition des taux de chevauchement par type de requête et pour chaque année

Au cours de nos expérimentations, nous avons évalué ce que nous avons appelé fusion *systématique* des systèmes à travers 2 stratégies différentes de fusion. Dans la première stratégie, les 10 systèmes qui obtiennent les plus grandes valeurs de F-mesure sont utilisés pour réaliser la fusion par union et par intersection avec le meilleur système. La

deuxième stratégie (stratégie2) que nous avons analysée consiste à choisir pour chaque mesure (rappel, précision, et F-mesure) les 10 systèmes qui obtiennent les meilleures performances. Cette stratégie s'avère meilleure que la première (stratégie1). Les résultats montrent que la stratégie 2 permet d'améliorer en moyenne les résultats plus que la fusion des meilleurs systèmes sur la base de la mesure F (27,75% contre 19,23% en 2002, et 16,8% contre 13,9% en 2003 pour le rappel). Ce résultat n'était pas attendu : en effet, intuitivement, il paraît plus facile d'améliorer les performances de " mauvais " systèmes que de " bons " systèmes (même si l'amélioration de " mauvais " systèmes par combinaison avec de meilleurs systèmes n'a pas vraiment d'intérêt).

Cependant, il est bon de rappeler que ces stratégies de fusion ne peuvent être mises en œuvre sans la connaissance de la performance des systèmes, donc du nombre de documents pertinents dans la collection utilisée ainsi que des scores des systèmes. Ces données sont disponibles dans le cadre de la campagne TREC sur laquelle nous nous sommes basé pour nos expérimentations. Nous avons également cherché à mesurer l'impact de la fusion par union et par intersection, lorsqu'aucune connaissance *a priori* sur les systèmes n'est disponible. Cette méthode de fusion, simple à implémenter, est applicable dans des conditions d'exploitation réelles. Les résultats montrent que les performances initiales des systèmes subissent une augmentation de 49,82% pour le rappel après application de la fusion par union en 2002 (48,74% en 2003). Pour la précision, l'amélioration des performances initiales est de l'ordre de 31,36% en 2002 avec la fusion par intersection (14,58% en 2003). La mesure des améliorations ainsi obtenue permet de retenir par exemple la fusion par union dans le cadre de constitution de corpus. La constitution de corpus à de nombreux domaines d'application, que ce soit dans la veille scientifique ([HM04]) ou dans la construction automatisée d'ontologie de domaine ([HCHM06]).

Dans le cadre expérimental de TREC, nous avons ensuite étudié une combinaison moins systématique que l' intersection et l' union telle que nous les avons appliquées. Nous avons retenu la stratégie2 pour la suite de nos expérimentations. Dans l'approche que nous avons proposée pour la fusion systématique des systèmes, les requêtes jouent toutes un rôle identique. En effet, la fusion est appliquée sur l'ensemble des requêtes, sans tenir compte de leurs spécificités. Nous avons donc dans une deuxième partie analysé l'impact de certaines caractéristiques liées aux requêtes lorsque nous utilisons la stratégie2 de fusion. Les caractéristiques que nous avons analysées correspondent à la typologie des requêtes (faciles, intermédiaires ou difficiles) et au taux de chevauchement des documents pertinents restitués par les systèmes.

La typologie des requêtes est déclinée en 3 catégories : les requêtes faciles, les requêtes difficiles, et les requêtes intermédiaires. Nous avons évalué l'impact de cette typologie de requêtes sur la fusion. Nos résultats montrent que contrairement à ce à quoi l'on pourrait s'attendre, il est plus facile d'améliorer les performances obtenues pour les requêtes faciles que celles des requêtes difficiles sous certaines conditions. Par exemple, l'amélioration du rappel lors de la fusion par union des requêtes faciles est de l'ordre de 53,90% en 2002 et de 25,54% en 2003. L'augmentation maximale du rappel est de

53,90% pour les requêtes faciles, et de 97,12% pour les requêtes difficiles avec la fusion par union. La fusion par intersection permet d'améliorer les performances moyennes des systèmes à hauteur de 11,48% pour les requêtes faciles, et jusqu'à 15,60% pour les requêtes difficiles.

L'analyse du taux de chevauchement complète l'analyse de la typologie des requêtes. Nous avons, à travers les expérimentations sur le taux de chevauchement, montré qu'il existe un lien entre les types de requêtes et le taux de chevauchement des documents pertinents. Il est donc possible d'utiliser le taux de chevauchement pour avoir une idée de la typologie des requêtes (faciles, intermédiaires et difficiles) utilisées sous certaines hypothèses. Par exemple, 17,4% des systèmes fusionnés obtiennent un taux de chevauchement compris entre 0,8 et 0,9 pour les requêtes faciles en 2002 (et 2,6% pour les requêtes difficiles), et 49,2% des systèmes obtiennent un taux de chevauchement de 1 (17,4% pour les requêtes difficiles). En 2003, 24,4% des sytèmes fusionnés obtiennent un taux de chevauchement de 1 pour les requêtes difficiles, et 10,2% des systèmes obtiennent un taux de chevauchement de 1 pour les requêtes faciles.

S'il est vrai que le taux de chevauchement semble être lié à la collection utilisée, le taux de chevauchement peut être vu comme un indicateur sur la difficulté de la requête. Cela permet en situation d'exploitation réelle d'avoir un moyen pour catégoriser les requêtes.

Les conclusions que nous tirons à l'issue de nos expérimentations nécessitent d'être validées sur plusieurs années. En effet, nous avons utilisé 2 années dans ce chapitre, et les résultats auxquels nous aboutissons constituent une piste intéressante qui devra être complétée. Il en est de même pour les expérimentations portant sur la typologie des requêtes, ainsi que sur le taux de chevauchement. Notre échantillon de requêtes est constitué de 12 requêtes et pourra être étendu afin de pouvoir généraliser les résultats. Nous songeons notamment à utiliser un sous-ensemble de SRI ayant participé à plusieurs années de TREC comme perspective à court terme de nos travaux, afin de compléter nos analyses.

Les propositions de fusion que nous avons faites dans ce chapitre sont basées sur des couples de systèmes. Dans le chapitre suivant, nous intégrons un nombre plus important de systèmes dans la fusion en proposant deux algorithmes de fusion probabilistes : MaxProb et MaxProbSeg. Nous avons abordé dans le chapitre 4 une typologie de requêtes basée sur les CL. Cette typologie de requêtes peut être envisagée en situation réelle, car la procédure de calcul des CL des requêtes est connue et réalisable. Nous n'avons cependant pas évalué le temps machine nécessaire pour déterminer les CL d'une requête, mais nous pensons que cette méthode est applicable en situation d'exploitation réelle. Nous focalisons donc dans le chapitre suivant sur l'application des techniques de fusion probabilistes que nous proposons sur les classes définies par les CL des requêtes (cf. chapitre 4).

Chapitre 6

Méthode adaptative en fonction du contexte linguistique de la requête

6.1 Introduction

Les travaux que nous présentons dans ce chapitre sont complémentaires de ceux que nous avons introduits respectivement dans les chapitres 4 et 5 de ce manuscrit.

Notre intuition de départ du chapitre 4 est que les requêtes peuvent être regroupées en fonction de leurs typologies linguistiques, afin de faciliter le choix de la meilleure stratégie à utiliser. Nous avons montré à travers une série d'expérimentations que les résultats que nous obtenons sont satisfaisants, ce qui laisse penser que le "profilage" linguistique des requêtes est une méthode intéressante.

Dans le chapitre 5, nous avons mené une étude sur la fusion de données en proposant des méthodes faciles à mettre en œuvre, basées sur 2 opérateurs de fusion qui sont l' union et l' intersection. Plusieurs pistes ont été explorées et nous avons notamment étudié l'impact d'une typologie des requêtes sur les résultats de la fusion, ainsi que l'impact du taux de chevauchement des documents restitués lors de la fusion.

L'objet de ce chapitre est d'établir un rapprochement entre la classification linguistique des requêtes (chapitre 4) et les techniques de fusion (chapitre 5). L'intuition que nous avons est qu'en établissant le couplage classification linguistique/ fusion, nous sommes à même de proposer des stratégies de recherche capables de s'adapter en fonction du contexte linguistique de la requête. Nous proposons dans ce chapitre deux méthodes de fusion probabilistes (MaxProb et MaxProbSeg) permettant de mettre en place une phase d'apprentissage pendant laquelle la meilleure stratégie à utiliser en fonction des CL des requêtes est déterminée. Cette stratégie est ensuite appliquée sur les données lors de la phase de test.

Nous présentons ces méthodes de fusion dans la section 6.2. Dans la section 6.3, nous présentons en détail le processus de fusion, appliqué aux catégories linguistiques de requêtes à travers un exemple. Nous terminons ce chapitre en présentant les expéri-

mentations que nous avons menées ainsi que les résultats que nous avons obtenus dans la section 6.4.

6.2 MaxProb et MaxProbSeg : deux algorithmes probabilistes de fusion de données

Nous présentons dans cette section deux algorithmes que nous avons proposés et utilisés dans nos travaux [KM07]. Nous nous sommes inspirés de l'algorithme ProbFuse présenté dans [LTP+06]. Dans ces travaux, les auteurs estiment pour chaque SRI *une probabilité de pertinence* pour chacun des *segments*[1] de la liste de documents retrouvés par un système en leur affectant un score. La liste finale qui est restituée à l'utilisateur est composée des documents restitués par tous les SRI. Les documents sont choisis en fonction d'un certain nombre de paramètres tels que, la position des documents dans les listes, et la probabilité de pertinence des segments des listes de documents restitués par les systèmes.

Certaines techniques de fusion utilisent la notion de score dans leur traitement [VH00], tandis que d'autres se basent sur la notion de rang des documents. Nos travaux se situent dans ces deux catégories d'algorithmes. Les algorithmes MaxProb et MaxProbSeg exploitent le rang des documents dans les listes restituées. Nous utilisons la notion de rang car, les scores attribués par les SRI ne sont pas forcément disponibles. Il faut toutefois noter que le rang et le score des documents sont corrélés. La croissance du rang équivaut à une décroissance du score du document. Nous établissons un score pour chaque système que nous utilisons dans notre approche.

L'algorithme de base (ProbFuse) combine les documents en fonction de leur position dans les listes de documents restitués par les différents SRI. D'après notre analyse du chapitre précédent, la sélection d'un certain nombre de systèmes permet d'obtenir de bonnes performances lors de la fusion (la fusion n'est pas forcément bénéfique lorsque tous les systèmes sont utilisés). Les deux algorithmes que nous proposons sont des adaptations de l'algorithme ProbFuse, et fonctionnent dans un contexte d'apprentissage et de test. La première adaptation que nous proposons (MaxProb) diffère de ProbFuse essentiellement lors de la phase de test. En effet, ProbFuse attribue à tous les documents un score à partir d'une combinaison des scores attribués par les autres systèmes. MaxProb quant à lui se base sur les performances passées des systèmes pour sélectionner par segment le système qui sera utilisé. La liste finale est alors constituée de documents issus de différents segments et de différents systèmes. MaxProbSeg réalise la fusion en choisissant les N meilleurs systèmes.

Les algorithmes MaxProb et MaxProbSeg sont décrits ci-dessous.

[1] Un segment correspond à un ensemble de X documents successifs dans la liste de documents restitués par les SRI.

MaxProb

Phase d'entraînement

```
Pour chaque requête d'entraînement faire
  Soumettre la requête aux SRI
  Pour chaque SRI faire
    Segmenter la liste de documents restitués
    Pour chaque segment faire
      Calculer la probabilité de pertinence des documents du segment
    Fin Pour
  Fin Pour
Fin Pour
Identifier pour chaque segment le SRI qui obtient le plus grand score de probabilité
```

Phase de test

```
Pour chaque requête de test faire
  Soumettre la requête aux SRI
  Pour chaque SRI, restituer les documents appartenant au segment pour lequel ce
  SRI obtient le plus grand score de pertinence
  Fin Pour
Fin Pour
```

MaxProbSeg

L'algorithme MaxProbSeg est une variante de l'algorithme MaxProb. La différence entre les deux algorithmes se situe au niveau de leur phase de test. La phase de test de MaxProbSeg est décrite comme suit :

Phase d'entraînement

Idem que MaxProb

Phase de test

Pour chaque requête de test faire
Soumettre la requête aux N meilleurs SRI
Récupérer la liste des documents restitués par les N SRI (les doublons sont supprimés)
Pour chaque document de la liste précédente faire
Pour chacun des N SRI faire
Identifier le segment auquel le document appartient
Mettre à jour le score du document en fonction de la probabilité de pertinence du segment
Fin Pour
Ordonner les documents en fonction de leur score, et restituer la liste ainsi obtenue en résultat
Fin Pour
Fin Pour

Les deux algorithmes que nous venons de présenter fonctionnent sur la base d'une phase d'apprentissage et d'une phase de test.

Durant l'étape d'apprentissage, les performances des SRI sont étudiées, afin de pouvoir estimer la probabilité de pertinence de chaque système. Cette probabilité de pertinence est calculée pour chaque segment de la liste de documents restitués par les SRI. La formule de calcul de cette probabilité de pertinence (reprise de l'algorithme ProbFuse) est la suivante :

$$P(D_{seg_k}/SRI) = \frac{\sum_{q=1}^{Q} \frac{|R_{k_q}|}{|k|}}{|Q|}. \tag{6.1}$$

Dans la formule 6.1, D_{seg_k} représente un document qui se trouve dans le segment nommé k. Un segment [LTP+05] correspond à une partie de la liste de documents ordonnée retrouvés par un SRI. Le découpage de la liste s'établit en définissant un certain seuil qui correspond au nombre de documents successifs que doit contenir un segment. La probabilité qu'un document retrouvé par un SRI dans le segment k, et pour une requête q, soit pertinent est exprimée sous forme d'une fraction. Le numérateur correspond au nombre de documents dans le segment k qui sont pertinents pour la requête q ($|R_{k_q}|$). Le dénominateur quant à lui représente le nombre de documents présents dans le segment k ($|k|$). Pour un SRI donné, la probabilité de pertinence d'un document dans un segment est obtenue en faisant une moyenne des probabilités de pertinence pour le même SRI, obtenues pour le même numéro de segment. Cette moyenne est calculée par rapport au nombre total de requêtes qui sont utilisées ($|Q|$). Ainsi, tous les documents d'un même segment possèdent la même probabilité de pertinence, indépendamment de leurs positions dans le segment.

À l'issue de la phase d'apprentissage des deux algorithmes, chaque SRI possède un ensemble de probabilités de pertinence, indiquant la probabilité qu'un document dans un segment donné soit pertinent. Ces probabilités de pertinence ainsi calculées sont

utilisées lors des différentes phases de test des algorithmes.

Phase de test de MaxProb

Durant la phase de test de l'algorithme MaxProb, les différentes probabilités de pertinence des systèmes sont analysées segment par segment. Ainsi, pour un numéro de segment donné, le système qui a obtenu la plus grande probabilité ($\max \mathrm{P}(\mathrm{D}_{seg_k}|\mathrm{SRI})$) de pertinence est retenu, et les documents qu'il retrouve dans ce segment sont récupérés. La liste de documents finale qui est restituée en réponse à une requête de test, est constituée (segment par segment) à partir des résultats des SRI ayant les plus grandes probabilités de pertinence pour chaque segment. En cas de doublons dans la liste de documents, le numéro de segments dans lesquels se trouvent les documents permet de déterminer le document qui sera gardé dans la liste. Ainsi, seul le document qui se trouve dans le segment dont le numéro est le plus petit est conservé. Les doublons sont alors supprimés de la liste. Par exemple, si un document est restitué par deux SRI respectivement dans le segment 3 et dans le segment 8, seul le document du segment 3 est retenu. Cela permet d'accorder une certaine importance aux segments, en fonction de leur numéro d'ordre. En d'autres termes, les documents dans le segment 1 sont supposés plus pertinents que ceux dans les segments dont le numéro est plus grand. Cependant, cette notion d'importance du segment est relative aux autres segments. Les documents contenus dans un même segment ont quant à eux la même probabilité de pertinence, peu importe la position qu'ils occupent dans le segment. MaxProb favorise l'effet d'écrémage (ou *skimming effet*), car il ne permet de sélectionner qu'un certain nombre de documents (segment par segment) provenant de différents SRI.

Phase de test de MaxProbSeg

Contrairement à MaxProb qui se contente de récupérer les documents issus de différents SRI pour constituer une liste finale, MaxProbSeg calcule un score de pertinence pour chaque document. Dans le cas où l'ordre des documents restitués a une importance dans la liste finale, l'algorithme MaxProb atteint ses limites. En effet, MaxProb ne prend pas en compte le rang des documents une fois que la liste finale est constituée, chacun des documents dans un segment étant considéré avec la même importance. Pour pallier cette situation, MaxProbSeg permet d'établir un classement des documents à partir de leur score de pertinence. Ce score est calculé par la formule suivante :

$$Score_D = \sum_{SRI_j \in S} \frac{P(D_{seg_k}/SRI_j)}{k}. \tag{6.2}$$

Dans l'équation 6.2, le score d'un document dépend des scores que les N meilleurs SRI lui ont attribués. Dans cette équation, S représente l'ensemble des N SRI qui sont utilisés lors de la recherche. La détermination des N systèmes utilisés dans l'algorithme MaxProbSeg est fonction de la probabilité de pertinence de la segmentation effectuée dans les listes restituées par les systèmes. La moyenne des probabilités de pertinence

pour tous les segments est calculée pour chaque SRI et chaque itération de l'apprentissage. Une valeur correspondant à la moyenne des moyennes de probabilités de pertinence est alors associée aux SRI. Cette pondération permet de classer les différents systèmes sélectionnés. Les N systèmes qui obtiennent la plus grande valeur sont sélectionnés et utilisés dans le calcul du score final qui sera attribué aux documents. Ce nombre (N) est déterminé manuellement dans nos expérimentations.

Le fait de diviser la probabilité de pertinence par le numéro de segment permet de favoriser les documents trouvés dans les segments ayant un petit numéro. Par exemple, le segment 1 est favorisé par rapport au segment 2 et ainsi de suite pour les autres segments. MaxProbSeg favorise l'effet de choeur (ou *chorus effect*) car, lorsqu'un document est retrouvé par différents SRI, son score ne dépend pas seulement d'un seul SRI, mais de sa position dans les listes de documents restitués par les autres SRI. Lorsqu'un document n'est pas restitué par un SRI, sa probabilité de pertinence pour ce SRI est égale à 0, afin d'éviter de favoriser certains documents. Le même principe est appliqué lorsque le nombre de segments du SRI_i est inférieur au nombre maximal de segments déterminé par rapport à l'ensemble des systèmes.

6.3 Exemple de fonctionnement des algorithmes MaxProb et MaxProbSeg

Dans cette section, nous montrons à travers un exemple comment nous adaptons la fusion probabiliste des données à la classification des requêtes. Supposons que les éléments suivants soient disponibles :

5 requêtes d'apprentissage sont choisies : ($Q_1,\ Q_2,\ Q_3,\ Q_4,\ et\ Q_5$)
1 requête de test est disponible : Q_T
2 classes de requêtes sont déterminées à l'issue de la phase d'apprentissage des CL des requêtes : classe$_1$($Q_1,\ Q_3$) et classe$_2$ ($Q_2,\ Q_4,\ Q_5$)
2 SRI dans le modèle adaptatif : $SRI_1\ et\ SRI_2$
1 liste de documents pertinents pour la requête Q_1. Pour des raisons de simplicité de la présentation de la méthode, nous déroulons l'exemple que pour les requêtes de la classe 1 :
$L_{pert1} = (D_{23},\ D_5,\ D_{15},\ D_{17},\ D_1,\ D_{43})$
1 liste de documents pertinents pour la requête Q_3 :
$L_{pert3} = (D_2,\ D_{33},\ D_{17},\ D_{10},\ D_{22},\ D_7,\ D_4,\ D_{43})$
Nous supposons qu'un segment contient 3 documents (|k| = 3)

La première phase du processus consiste à établir une classification des requêtes, basée sur leurs CL . À l'issue de cette classification, les deux classes de requêtes C_1 et C_2 sont définies lors de la phase d'apprentissage. Dans la suite de l'exemple et pour des raisons de simplification, nous utilisons les requêtes de la classe C_1 pour nos calculs. La figure 6.1 présente les résultats obtenus respectivement par les 2 SRI (SRI_1 et SRI_2) lorsque les requêtes d'apprentissage sont soumises.

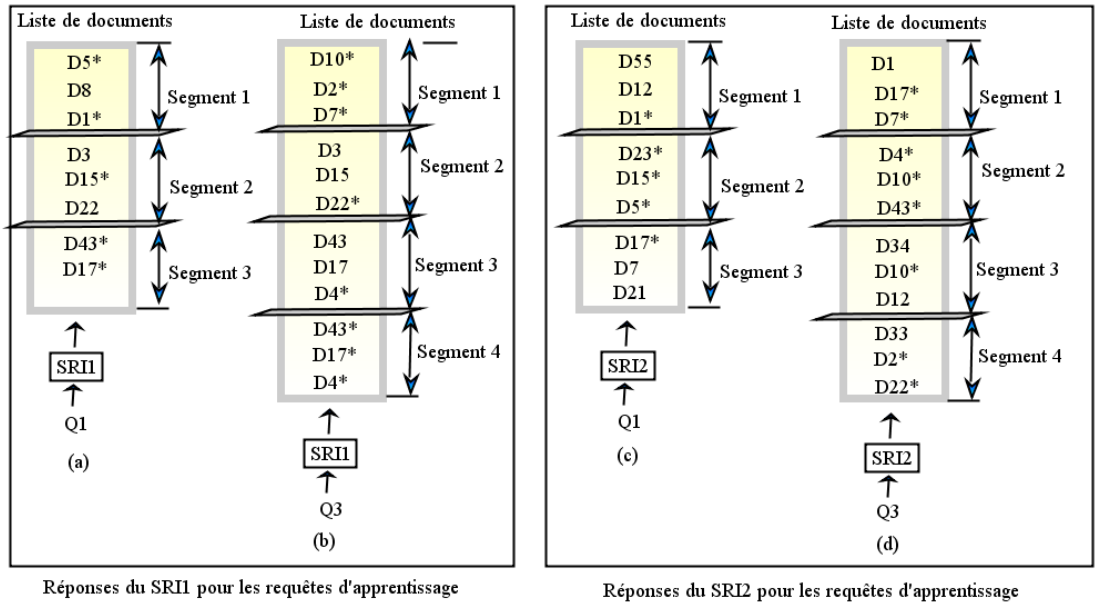

FIG. 6.1 – Exemple de liste de documents restituée par 2 SRI, en réponse aux requêtes Q_1 et Q_3.

Dans la figure 6.1, les documents de la forme D_i* sont considérés comme pertinents pour la requête. Ainsi, dans la figure 6.1(a), le SRI_1 restitue 8 documents dont 5 documents pertinents. Le deuxième segment de ce système contient 1 seul document pertinent sur les 3 documents du segment. En consultant les listes restituées par les 2 SRI, il est alors possible de déterminer la probabilité de pertinence de chaque segment dans les listes de documents. En appliquant la formule 6.1, on obtient les probabilités suivantes :

Segment	Probabilités de pertinence du SRI_1	Probabilités de pertinence du SRI_2
1	$\text{P}(\text{D}_{seg_1}\|\text{SRI}_1)_{classe_1} = 0{,}83$	$\text{P}(\text{D}_{seg_1}\|\text{SRI}_2)_{classe_1} = 0{,}5$
2	$\text{P}(\text{D}_{seg_2}\|\text{SRI}_1)_{classe_1} = 0{,}33$	$\text{P}(\text{D}_{seg_2}\|\text{SRI}_2)_{classe_1} = 1$
3	$\text{P}(\text{D}_{seg_3}\|\text{SRI}_1)_{classe_1} = 0{,}50$	$\text{P}(\text{D}_{seg_3}\|\text{SRI}_2)_{classe_1} = 0{,}33$
4	$\text{P}(\text{D}_{seg_4}\|\text{SRI}_1)_{classe_1} = 0{,}50$	$\text{P}(\text{D}_{seg_4}\|\text{SRI}_2)_{classe_1} = 0{,}33$

TAB. 6.1 – Exemple de calcul de probabilité de pertinence pour les requêtes de la classe_1

Les données dans le tableau 6.1 résument pour chaque SRI la probabilité de pertinence qu'il obtient pour chaque segment, sachant que seules les requêtes de la classe C_1 sont utilisées (par exemple, $\text{P}(\text{D}_{seg_1}|\text{SRI}_1)_{classe_1}$ représente la probabilité de pertinence d'un document situé dans le segment 1, sachant que le SRI_1 sera utilisé pour répondre aux requêtes de la classe_1). On remarque par exemple que le SRI_2 obtient une probabilité de pertinence égale à 1 pour le segment 2 car, tous les documents retrouvés dans le

segment 2 par ce système sont pertinents.

En appliquant l'algorithme MaxProb, on obtient les résultats décrits dans le tableau 6.2 à l'issue de la phase d'apprentissage.

Segment	SRI sélectionné
1	SRI_1
2	SRI_2
3	SRI_1
4	SRI_1

TAB. 6.2 – Stratégie de fusion d'après l'algorithme MaxProb pour les requêtes appartenant à la $classe_1$

Le tableau 6.2 préconise l'utilisation du SRI ayant obtenu la probabilité de pertinence maximale par rapport à un segment donné. Ainsi, en accord avec le calcul effectué dans le tableau 6.1, le SRI_2 est utilisé pour restituer les documents situés dans le segment 2. Nous supposons que lors de la phase de test, la requête Q_T est affectée dans la $classe_1$ des requêtes. La stratégie de fusion du tableau 6.2 est alors utilisée pour réaliser la fusion. Cette stratégie détermine pour quels segments les SRI doivent être utilisés.

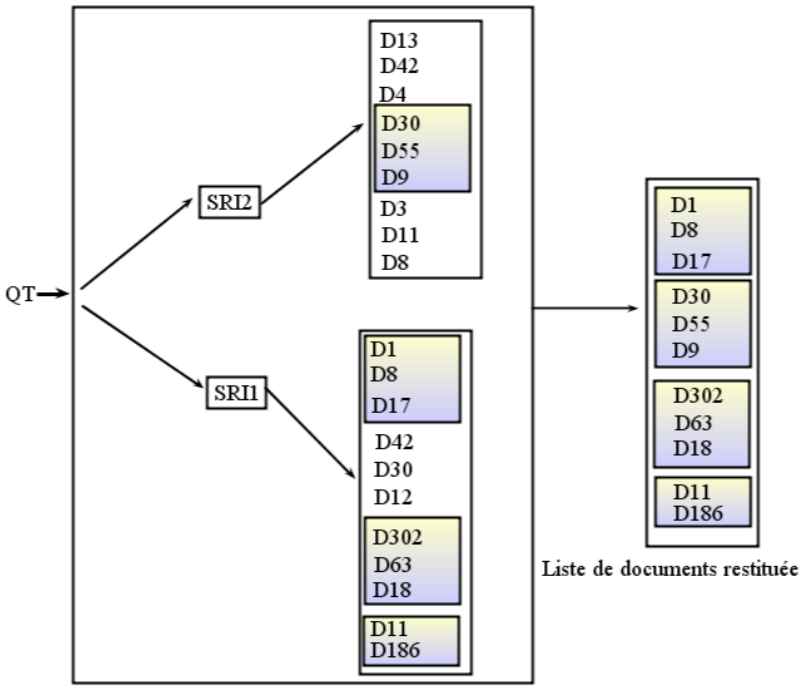

FIG. 6.2 – Processus de fusion de MaxProb

Dans la figure 6.2, nous montrons comment la liste de documents finale est constituée, segment par segment. Comme nous l'avions mentionné dans la section 6.2, les documents D_8, D_{17}, et D_1 ont la même importance dans la liste restituée. Dans le cas où l'utilisateur décide de ne consulter qu'un seul document dans la liste, un de ces trois (D_8, D_{17}, ou D_1) documents pourra être restitué à l'utilisateur.

En nous basant toujours sur l'exemple présenté en introduction de cette section, nous décrivons en détail le fonctionnement de l'algorithme MaxProbSeg, lors de sa phase de test. Nous supposons que l'ensemble des N SRI est composé du SRI_1 et du SRI_2 (ils obtiennent tous les 2 une valeur moyenne de pertinence égale à 0,54). Les probabilités de pertinence des segments correspondent à celles présentées dans le tableau 6.1, pour chacun des 2 SRI. Le processus de fusion de MaxProbSeg est illustré dans la figure 6.3.

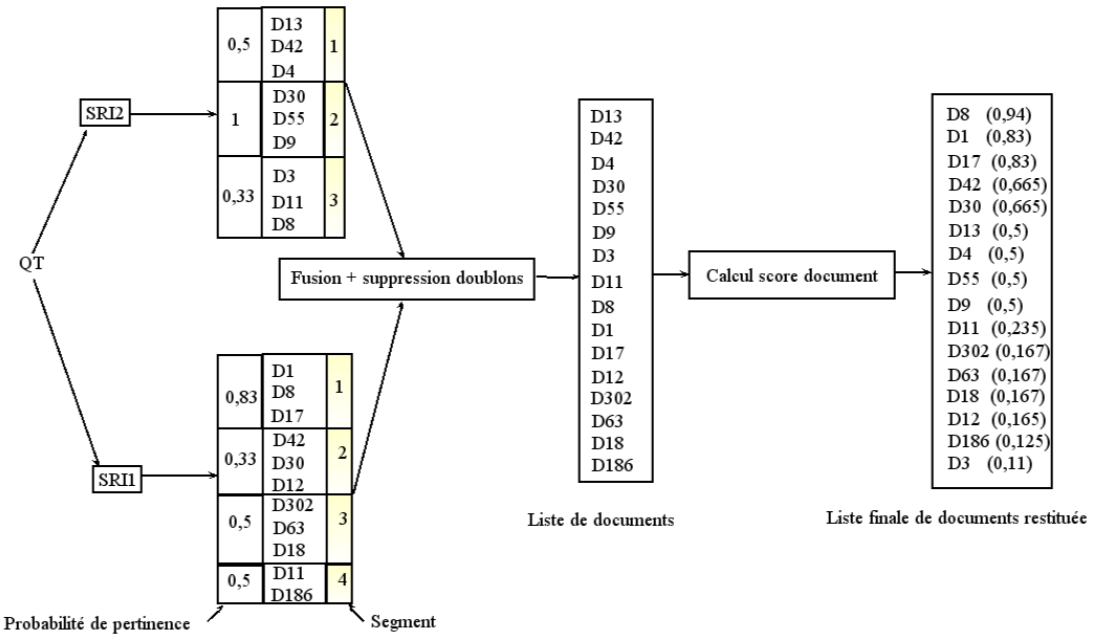

FIG. 6.3 – Processus de fusion de MaxProbSeg

Dans la figure 6.3, la liste finale des documents qui seront restitués à l'utilisateur est constituée en deux étapes. Dans une première phase, tous les documents retrouvés par le SRI_1 et le SRI_2 sont regroupés pour former une liste unique de documents. Dans cette liste, les documents en double sont supprimés. La deuxième phase utilise les documents ainsi regroupés, et en appliquant la formule de calcul de score, chaque document de la liste est associé à un score indiquant sa pertinence supposée. Nous rappelons que plus un document est retrouvé par plusieurs systèmes, plus ce document augmente ses chances d'être restitué. Le segment dans lequel le document est retrouvé permet aussi de favoriser ou non ce document dans la liste finale. Si nous reprenons le choix d'un utilisateur qui souhaite consulter un seul document dans la liste finale, le document D_8 sera cette fois

ci restitué, sachant qu'il obtient un score de pertinence supérieur au document D_1, qui lui même sera préféré au document D_{17} (contrairement à l'algorithme MaxProb). En comparant les listes restituées respectivement par MaxProb et MaxProbSeg, on constate par exemple que les 3 premiers documents sont les mêmes (si on ne tient pas compte de leur rang). Cependant, à partir du deuxième segment une différence est observée dans les listes. Par exemple, la liste constituée par MaxProb peut être découpée en 4 segments tandis que celle obtenue avec MaxProbSeg contient 6 segments. En poussant un peu plus l'observation, certains documents changent de segment comme les documents D_9 et D_{55} qui passent du segment 2 (MaxProb) au segment 3 (MaxProbSeg). Certains documents font aussi leur apparition dans les segments. C'est le cas par exemple des documents D_{42} et D_{13} (figure 6.3) qui sont considérés plus pertinents que les documents D_9 et D_{55} (figure 6.2). Dans la cas de l'utilisation de MaxProbSeg, il peut arriver que des documents obtiennent le même score de pertinence. Nous avons choisi de ne pas mettre en place un traitement spécifique pour les ex æquo. En effet, il n'ya pas d'incidence des ex æquo sur les résultats de la fusion avec MaxProbSeg car le calcul du score de pertinence ne tient pas compte du rang du document mais du score obtenu par le document. Par exemple, pour la P@5, si les documents situés en 5ème (D_5) et 6ème (D_6) position sont ex æquo, le calcul de la P@5 donnerait le même résultat si l'on décidait de classer le document D_5 en 6ème position et le document D_6 en 5ème position.

6.4 Évaluation

Les évaluations que nous avons effectuées sont basées sur les collections de TREC3, TREC5, TREC6, et TREC7. Nous avons utilisé pour nos expérimentations 5 mesures de performance qui sont la MAP, la R-précision, la P@5, la P@10, et la P@15. Nous avons implémenté les algorithmes MaxProb et MaxProbSeg en utilisant le langage *Perl*, et le programme *trec_ eval* fourni par TREC.

Nous donnons dans la section 6.4.1 les résultats préliminaires que nous avons obtenus. Nous présentons les résultats que nous avons obtenus avec l'algorithme MaxProbSeg dans les sections 6.4.2 et 6.4.3. Nous terminons la présentation de nos expérimentations dans la section 6.4.4, en donnant les résultats des tests statistiques que nous avons appliqués à l'issus de nos travaux.

6.4.1 Résultats préliminaires avec MaxProb

Les performances initiales des systèmes sont données pour chaque année de TREC et chaque mesure. Un exemple de performances initiales est présenté dans les tableaux 6.3 et 6.4. Les performances initiales correspondent aux performances officielles que les SRI ont obtenues lors de leur participation à TREC. Ces performances sont classées par ordre décroissant pour chaque mesure considérée.

On peut remarquer dans les tableaux 6.3 et 6.4 que le système inq102 est celui qui obtient les meilleures performances pour TREC3 sauf pour la P@5. Les 5 meilleurs systèmes de TREC3 en termes de MAP, font également partie (à 80%) des 5 meilleurs

		MAP		R-précision
TREC3	inq102	0,4226	inq102	0,4524
	citya1	0,4012	citya1	0,4217
	brkly7	0,3714	brkly7	0,4152
	inq101	0,3659	inq101	0,4088
	assctv2	0,3539	assctv2	0,3999
TREC5	ETHme1	0,3070	uwgcx0	0,3444
	uwgcx1	0,2954	uwgcx1	0,3392
	uwgcx0	0,2944	ETHme1	0,3305
	Cor5M2rf	0,2832	LNmFull2	0,3115
	LNmFull2	0,2809	LNmFull1	0,3013

Tab. 6.3 – Performances initiales des 5 meilleurs systèmes – Exemple de TREC3 et TREC5 (MAP et R-précision)

		P@5		P@10		P@15
TREC3	brkly7	0,7600	inq102	0,7220	inq102	0,6867
	inq102	0,7440	brkly7	0,7120	brkly7	0,6813
	citya1	0,7400	citya1	0,7120	citya1	0,6787
	citya2	0,7320	citya2	0,6940	citya2	0,6560
	assctv2	0,7280	assctv2	0,6760	assctv2	0,6453
TREC5	ETHme1	0,6160	ETHme1	0,5460	ETHme1	0,5147
	genrl3	0,5800	Cor5M2rf	0,5080	uwgcx1	0,4707
	Cor5M2rf	0,5760	uwgcx1	0,5000	uwgcx0	0,4627
	uwgcx0	0,5560	uwgcx0	0,4880	Cor5M2rf	0,4547
	uwgcx1	0,5520	LNmFull2	0,4780	LNmFull2	0,4333

Tab. 6.4 – Performances initiales des 5 meilleurs systèmes – Exemple de TREC3 et TREC5 (Haute précision)

systèmes pour les autres mesures, mais à des rangs différents (absence de inq101 pour la MAP et R-précision, et citya2 pour les mesure P@).

Pour TREC5, le système ETHme1 obtient les meilleures performances pour la MAP, et les mesures de haute précision (P@5, P@10, et P@15). Pour la R-précision, le meilleur système est le système uwgcx0. Les 3 meilleurs systèmes pour la MAP font aussi partie des meilleurs systèmes pour les autres mesures.

6.4.1.1 Analyse locale des résultats

Nous avons mené une série d'expérimentations pour évaluer les méthodes que nous proposons. Lors des 10 itérations successives que nous avons effectuées (phase d'apprentissage + phase de test), les résultats que nous avons obtenus lors de la phase d'apprentissage ont permis de déterminer les probabilités de pertinence pour chacun des segments des listes restituées par les systèmes. La taille des segments a été fixée à

25 dans nos expérimentations comme préconisé dans la littérature ([LTP+06]), c'est à dire que nous considérons que chaque segment contient 25 documents. Dans le tableau 6.5, nous donnons un exemple de résultat issu de la première itération avec TREC3. Ces résultats correspondent à l'utilisation des requêtes de test affectées à la classe 1. Nous présentons dans ce tableau les résultats obtenus avec MaxProb pour chaque requête de test. Nous comparons les résultats de MaxProb et ceux du meilleur système que nous avons associé à la classe 1 de requêtes (cf. chapitre 4, section 4.4.3). Nous appelons ce système Best_Cl1.

requêtes	Mesures	MaxProb	Best_Cl1
Req161 (305)	Nb-Pert	232	280
	map	0,3776	0,5921
	R-prec	0,4525	0,5803
	P5	1	0,8
	P10	1	0,8
	P15	0,9333	0,8667
Req183 (228)	Nb-Pert	208	225
	map	0,5082	0,5302
	R-prec	0,5263	0,557
	P5	0,8	1
	P10	0,8	0,8
	P15	0,8	0,6
Req194 (123)	Nb-Pert	82	48
	map	0,222	0,064
	R-prec	0,2846	0,1545
	P5	0,8	1
	P10	0,8	0,3
	P15	0,6667	0,2

TAB. 6.5 – Première itération sur la classe 1 de requêtes de test – TREC3

Dans le tableau 6.5, la première colonne représente les requêtes de test de la classe 1. Les chiffres entre parenthèses correspondent au nombre de documents pertinents qu'un système idéal devrait retrouver pour chaque requête. Par exemple, le nombre de documents pertinents associés à la requête *Req194* est égal à 123. On constate par exemple que pour les deux requêtes de test 183 et 161, les résultats obtenus en terme de MAP et de R-précision sont meilleurs lorsque Best_Cl1 est utilisé (contrairement à MaxProb qui permet d'obtenir les meilleures MAP et R-précision pour la requête 194). L'approche MaxProb est toutefois plus indiquée (pour les trois requêtes) lorsque les mesures de haute précision sont utilisées (P@10 et P@15). Ces résultats nous indiquent que MaxProb permet de retrouver un nombre de documents pertinents plus élevé (parmi les 15 premiers documents) que Best_Cl1.

En observant de plus près le tableau 6.5, on remarque que pour la requête 194, MaxProb permet de retrouver plus de documents pertinents que le système Best_Cl1 (cf

lignes *Nb-pert* du tableau). En prenant la mesure de R-précision, le système Best_Cl1 permet d'obtenir de meilleurs résultats pour les requêtes 161 et 183. L'analyse des requêtes de la classe 1 pour la première itération nous montre les faibles performances de MaxProb par rapport à Best_Cl1 lorsque la MAP et la R-précision sont considérées (requêtes 161 et 183). Il n'en demeure pas moins que la répartition des documents pertinents est satisfaisante pour MaxProb qui semble apporter de meilleures performances pour les mesures de haute précision.

Dans les campagnes TREC, chaque système participant restitue une liste de 1000 documents qui sont évalués par l'intermédiaire du programme trec_eval. Chaque liste restituée est formée de 40 segments différents. En l'occurrence, la taille du segment vaut 25, et le nombre de documents restitués vaut 1000 (soit 1000/25).

Avant de présenter les résultats que nous avons obtenus, nous donnons dans le tableau 6.6 la liste des systèmes que nous avons utilisés dans l'algorithme MaxProb avec la collection TREC3.

segments	systèmes	segments	systèmes	segments	systèmes
1	**inq102**	15	crnlla	29	crnlla
2	inq102	16	crnlla	30	crnlla
3	assctv2	17	crnlla	31	crnlla
4	inq102	18	siems2	32	crnlla
5	citya2	19	dortd2	33	crnlla
6	citya1	20	crnlea	34	crnlla
7	brkly7	21	crnlla	35	crnlla
8	citya2	22	crnlla	36	crnlla
9	vtc2s2	23	crnlla	37	crnlla
10	citya1	24	crnlla	38	crnlla
11	crnlea	25	crnlla	39	crnlla
12	inq102	26	crnlla	40	crnlla
13	crnlla	27	crnlea		
14	westp1	28	crnlla		

Tab. 6.6 – Meilleurs systèmes par segment selon MaxProb pour les requêtes d'entraînement de issues de la classe 1 de TREC3

Dans le tableau 6.6, chaque segment est associé au système ayant obtenu la meilleure probabilité de pertinence. Par exemple, le système inq102 est utilisé pour restituer les documents du premier segment. Ce système fait partie des 2 meilleurs systèmes de TREC3, indépendamment de la mesure choisie. On peut aussi noter que le système crnlla est utilisé pour restituer les documents de plusieurs segments différents, alors qu'il ne figure pas dans la liste des 5 meilleurs SRI (choisis en fonction de leur performance moyenne sur l'ensemble des documents restitués en réponse aux requêtes) pour aucune mesure. En tout, 12 systèmes sont utilisés pour restituer les documents répondant aux

requêtes de test.

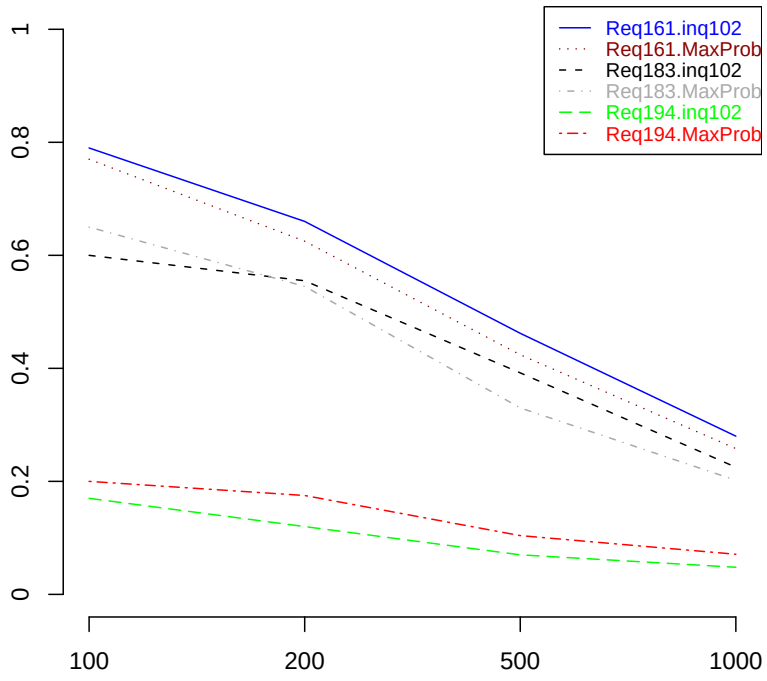

FIG. 6.4 – Répartition des documents pertinents pour les requêtes 194, 183, et 161

Dans la figure 6.4, nous comparons la répartition des documents pertinents pour les 3 requêtes de test choisies lors de la première itération (requête 194, 183, et 161). Les indices en abscisse représentent le nombre de documents pour lesquels la précision est mesurée ; par exemple, la mesure de la précision lorsque 5 documents sont retrouvés nous indique que le système Best_Cl1 retrouve 2 documents pertinents (soit une P@5 de 0,4) tandis que MaxProb permet de retrouver 4 documents pertinents (soit une P@5 de 0,8). Dans la figure 6.4, best_Cl1 correspond au système inq102. On peut aussi noter d'après le tableau 6.6 que inq102 est utilisé par MaxProb pour restituer les documents des segments 1, 2, 4 et 12. Les résultats obtenus par MaxProb et Best_Cl1 sont donc identiques pour les 50 premiers documents. Les 2 méthodes peuvent donc être utilisées indifféremment pour les 50 premiers documents pour les requêtes 194, 183 et 161. Nous nous intéressons donc pour ces 3 requêtes, à la répartition des 100, 200, 500, et 1000 documents lorsque les 2 méthodes que nous avons proposées sont utilisées.

Nous avons présenté dans cette section les résultats obtenus lors de la première

itération sur la classe 1 des requêtes de TREC3. Nous analysons dans la section suivante les résultats obtenus pour toutes les collections de TREC que nous avons utilisées, ainsi que pour toutes les itérations que nous avons réalisées dans nos expérimentations. Nous appelons itération une phase d'apprentissage suivie d'une phase de test. Durant chaque itération il est possible de d'établir une classification des requêtes et de calculer la probabilité de pertinence des SRI.

6.4.1.2 Analyse globale des résultats

La mesure de haute précision P@15 permet de calculer la précision lorsque les 15 premiers documents sont retrouvés. La taille d'un segment étant fixée à 25 documents, seul le premier segment est pris en compte par ces mesures. Ainsi, les performances de MaxProb pour les mesures de hautes précision dépendent de celles du système inq102. Le meilleur système en termes de P@5 est brkly7 et inq102 est le meilleur système pour les autres mesures. Comparer les performances de MaxProb à celles de best_sys (en termes de haute précision) revient à comparer les résultats du système inq102 et de best_sys. Best_sys correspond à la moyenne pondérée sur les 3 classes de requêtes des systèmes Best_Cli, i représentant le numéro de la classe pour laquelle le système est le meilleur. Best_sys est uniquement déterminé en fonction des requêtes d'entraînement.

L'exemple d'un jeu de 10 itérations sur la collection de documents de TREC3 est donné dans les tableaux 6.7, 6.8, et 6.9.

Dans les tableaux 6.7, 6.8, et 6.9, on remarque que pour certaines itérations les performances obtenues par MaxProb et Best_sys sont nulles. Par exemple lors de la 4ème et 6ème itération (cf. tableau 6.7), aucune des requêtes de test n'appartenait à la classe 1. Dans la mesure où les lignes qui contiennent des valeurs égales à 0 indiquent qu'aucune requête de test n'appartient à la classe considérée, ces lignes ne sont pas prises en compte lors du calcul de la moyenne indiquée dans la dernière ligne du tableau. En analysant la moyenne obtenue sur l'ensemble des 10 itérations et pour chaque mesure, on peut dire que MaxProb est plus adapté pour les mesures de haute précision, tandis que Best_sys est indiqué lorsque la MAP ou la R-précision sont utilisées.

Nous présentons dans la figure 6.5 les résultats que nous avons obtenus pour la MAP pour chaque année de TREC. Les résultats de chaque itération correspondent à la moyenne des résultats obtenus pour l'ensemble des 3 classes de requêtes déterminées à l'aide des CL des requêtes.

Dans la figure 6.5, excepté pour TREC3, MaxProb permet en moyenne (sur un ensemble de 10 itérations) d'obtenir de meilleures performances en termes de MAP que Best_sys. Ceci est vérifié pour TREC5, TREC6, et TREC7.

Le tableau 6.10 apporte des informations complémentaires à celles de la figure 6.5. Nous comparons les performances moyennes sur l'ensemble des itérations sur les 3 classes de requêtes. Nous pouvons conclure d'après ce tableau que l'algorithme MaxProb permet d'obtenir de meilleurs résultats en terme de MAP, P@5 et P10 que le meilleur

Classe 1

		map	R-prec	P5	P10	P15
Itération1	MaxProb	0,3693	0,4211	0,8667	0,8667	0,8000
	Best_Sys	0,3954	0,4306	0,9333	0,6333	0,5556
Itération2	MaxProb	0,3556	0,3955	0,8500	0,7500	0,6834
	Best_Sys	0,4907	0,5132	1,0000	0,7500	0,6834
Itération3	MaxProb	0,6135	0,5785	0,9000	0,9000	0,8667
	Best_Sys	0,5735	0,5518	0,9000	0,9000	0,8667
Itération4	MaxProb	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
	Best_Sys	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
Itération5	MaxProb	0,3300	0,3684	1,0000	1,0000	1,0000
	Best_Sys	0,3006	0,3895	0,8000	0,9000	0,8667
Itération6	MaxProb	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
	Best_Sys	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
Itération7	MaxProb	0,2100	0,2869	0,6000	0,7000	0,5333
	Best_Sys	0,2132	0,2623	0,4000	0,3000	0,2667
Itération8	MaxProb	0,1437	0,2222	0,2000	0,4000	0,2667
	Best_Sys	0,2768	0,2963	0,6000	0,4000	0,2667
Itération9	MaxProb	0,4934	0,5000	0,8000	0,8500	0,8333
	Best_Sys	0,4750	0,4765	0,9000	0,8500	0,8667
Itération10	MaxProb	0,3381	0,3469	0,9000	0,7000	0,6667
	Best_Sys	0,3998	0,4259	0,8000	0,7500	0,6667
Moyenne	**MaxProb**	**0,3567**	**0,3899**	**0,7646**	**0,7708**	**0,7063**
	Best_Sys	**0,3906**	**0,4183**	**0,7917**	**0,6854**	**0,6299**

TAB. 6.7 – Résumé des itérations sur la classe 1 de TREC3

système pour TREC3 et TREC5. Les résultats en terme de R-précision sont plus défavorables pour toutes les années, idem pour la MAP pour TREC3. Seules les collections de TREC7 permettent d'obtenir de meilleurs résultats avec MaxProb pour la R-précision. Cependant, MaxProb est moins performant que le meilleur système de TREC7 pour les mesures de haute précision.

6.4.2 Analyse locale des résultats obtenus avec MaxProbSeg

Nous avons présenté dans la section précédente les résultats préliminaires issus de la mise œuvre de l'algorithme MaxProb. Dans cette section, nous décrivons les résultats que nous avons obtenus lorsque nous utilisons l'algorithme MaxProbSeg. Notre intuition est que MaxProbSeg devrait permettre d'obtenir de meilleurs résultats que MaxProb. Une des spécificités de MaxProbSeg est liée au choix des systèmes qui interviennent dans la fusion. Nous avons fait l'hypothèse que tous les systèmes ne gagnaient pas à être fusionnés. Nous avons sélectionné les N meilleurs systèmes pour réaliser la fusion. Nous appelons "meilleur système" un système qui obtient en moyenne la meilleure probabilité de pertinence, pour une année donnée, lors de la phase d'apprentissage.

Classe 2

		map	R-prec	P5	P10	P15
Itération1	MaxProb	0,0693	0,1212	0,4000	0,3500	0,2334
	Best_Sys	0,3347	0,2556	0,6000	0,3500	0,2334
Itération2	MaxProb	0,6251	0,6412	0,9333	0,8667	0,8889
	Best_Sys	0,6674	0,6525	0,9333	0,8667	0,8889
Itération3	MaxProb	0,1717	0,2204	0,3000	0,2000	0,2000
	Best_Sys	0,1703	0,2512	0,3000	0,1500	0,2334
Itération4	MaxProb	0,4902	0,5524	0,8000	0,8000	0,7333
	Best_Sys	0,4752	0,5277	0,6000	0,7000	0,7167
Itération5	MaxProb	0,2749	0,3494	0,6000	0,7500	0,6667
	Best_Sys	0,3160	0,3906	0,6000	0,7500	0,4667
Itération6	MaxProb	0,3662	0,3913	0,8000	0,5000	0,4667
	Best_Sys	0,3731	0,3043	0,8000	0,5000	0,4667
Itération7	MaxProb	0,2647	0,3348	0,7000	0,6500	0,6334
	Best_Sys	0,2679	0,3593	0,7000	0,6500	0,6334
Itération8	MaxProb	0,4443	0,5152	0,6000	0,8000	0,7333
	Best_Sys	0,3481	0,4242	0,6000	0,8000	0,4000
Itération9	MaxProb	0,4965	0,4615	1,0000	0,9000	0,8000
	Best_Sys	0,5864	0,5769	1,0000	0,9000	0,8000
Itération10	MaxProb	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
	Best_Sys	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
Moyenne	**MaxProb**	**0,3559**	**0,3986**	**0,6815**	**0,6463**	**0,5951**
	Best_Sys	**0,3932**	**0,4158**	**0,6815**	**0,6296**	**0,5377**

Tab. 6.8 – Résumé des itérations sur la classe 2 de TREC3

La valeur moyenne des probabilités de pertinence d'un système est calculée dans un premier temps sur l'ensemble des 40 segments de la liste de documents qu'il restitue lors d'une itération durant la phase d'apprentissage. La moyenne de ces différentes valeurs moyennes de pertinence permet de classer les systèmes. Dans le tableau 6.11, nous donnons pour la collection de TREC3 le classement des 5 meilleurs systèmes en fonction de leurs probabilités de pertinence. Le classement complet est fourni en annexe.

Dans le tableau 6.11, la deuxième colonne indique que la valeur moyenne de probabilité de pertinence des systèmes est calculée sur l'ensemble des requêtes, indépendamment de la classe à laquelle elles appartiennent. Cela nous donne une indication sur l'impact de notre méthode lorsque aucune classification n'est effectuée. Les colonnes *classe 1*, *classe 2*, et *classe 3* contiennent les meilleurs systèmes pour chaque classe de requête. On peut par exemple remarquer que les 4 meilleurs systèmes de la classe 3 sont les mêmes lorsqu'aucune classification n'est considérée.

Dans le tableau 6.12, les 5 meilleurs systèmes sont différents pour chaque classe et lorsqu'aucune classification n'est établie. Comme pour le tableau 6.11, les meilleurs systèmes dans le tableau 6.13 sont identiques pour la classe 2 et la classe 3, ainsi que

Classe 3

		map	R-prec	P5	P10	P15
Itération1	MaxProb	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
	Best_Sys	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
Itération2	MaxProb	0,3925	0,4496	0,8667	0,8667	0,8222
	Best_Sys	0,4432	0,5163	0,6667	0,7667	0,7111
Itération3	MaxProb	0,6768	0,6386	1,0000	1,0000	0,9334
	Best_Sys	0,7286	0,6940	1,0000	0,9500	0,9334
Itération4	MaxProb	0,1992	0,2743	0,7333	0,6333	0,6445
	Best_Sys	0,2655	0,3248	0,8000	0,4667	0,6222
Itération5	MaxProb	0,3025	0,3589	0,8000	0,8750	0,8333
	Best_Sys	0,3083	0,3901	0,7000	0,8250	0,6667
Itération6	MaxProb	0,2165	0,3163	0,7000	0,5500	0,5667
	Best_Sys	0,1657	0,2896	0,6000	0,5500	0,3334
Itération7	MaxProb	0,3282	0,3741	0,6667	0,6333	0,6000
	Best_Sys	0,4062	0,4151	0,6667	0,6333	0,7333
Itération8	MaxProb	0,3217	0,4251	0,2000	0,2000	0,3333
	Best_Sys	0,3311	0,4774	0,2000	0,2000	0,3333
Itération9	MaxProb	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
	Best_Sys	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
Itération10	MaxProb	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
	Best_Sys	**0,0000**	**0,0000**	**0,0000**	**0,0000**	**0,0000**
Moyenne	**MaxProb**	**0,3482**	**0,4053**	**0,7095**	**0,6798**	**0,6762**
	Best_Sys	**0,3784**	**0,4439**	**0,6619**	**0,6274**	**0,6191**

TAB. 6.9 – Résumé des itérations sur la classe 3 de TREC3

		MAP	R-précision	P@5	P@10	P@15
TREC3	MaxProb	*0,3536*	*0,3979*	0,7185	0,6989	0,6592
	Best_sys	0,3874	0,4260	0,7117	0,6475	0,5955
TREC5	MaxProb	0,3128	*0,3500*	0,6297	0,5509	*0,5319*
	Best_sys	0,2682	0,3660	0,5803	0,5249	0,5376
TREC6	MaxProb	0,4688	*0,4744*	0,7423	0,6703	*0,6093*
	Best_sys	0,4507	0,4747	0,6710	0,6228	0,6180
TREC7	MaxProb	0,3990	0,4402	*0,7012*	*0,6756*	*0,6441*
	Best_sys	0,3635	0,4150	0,7400	0,6884	0,6556

TAB. 6.10 – Comparaison des performances moyennes de MaxProb et Best_sys pour chaque année de TREC

lorsqu'aucune classification n'est faite. Le tableau 6.14 présente lui aussi des variabilités dans la liste des meilleurs systèmes pour chacune des 3 classes.

Nous avons mesuré les performances de l'algorithme MaxProbSeg lorsque le nombre

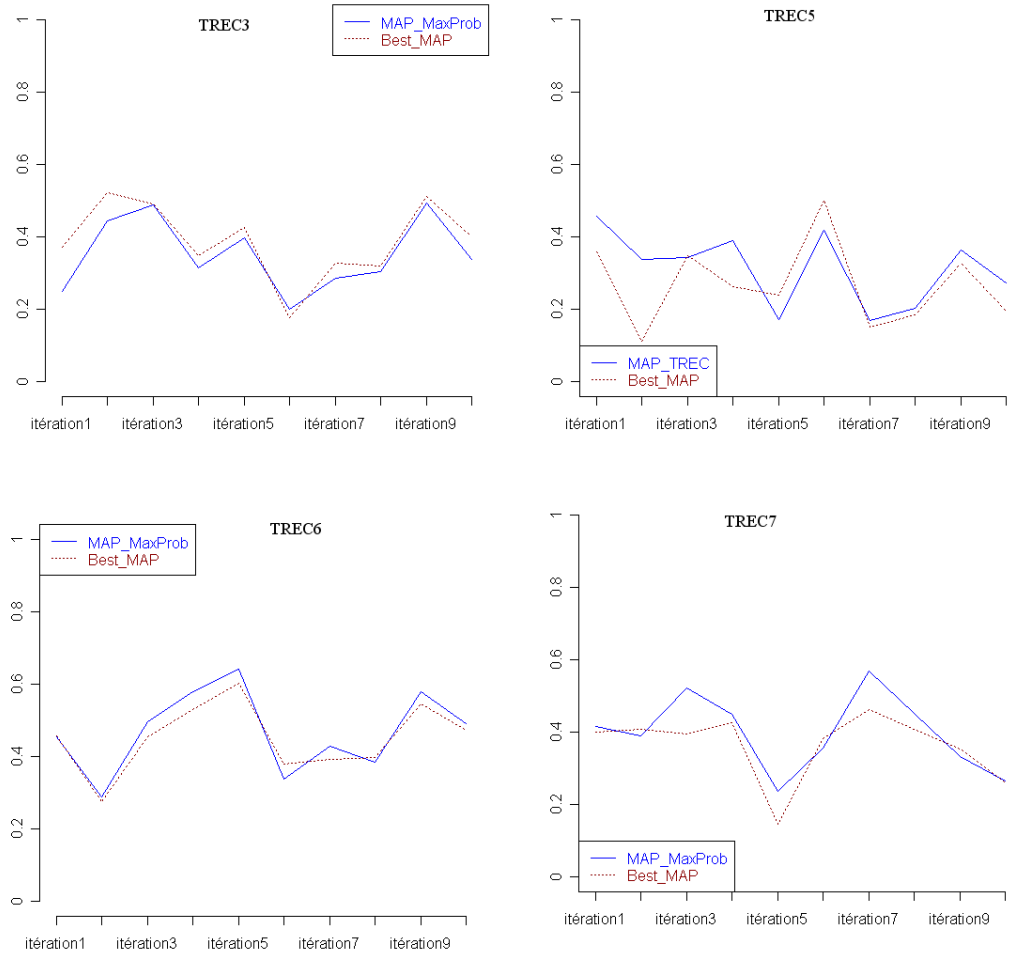

FIG. 6.5 – Comparaison de la MAP

TREC3				
Rang	**All**	**Classe 1**	**Classe 2**	**Classe 3**
1	crnlea	inq102	inq102	crnlea
2	citya1	citya1	crnlea	citya1
3	inq102	crnlea	citya1	inq102
4	inq101	crnlla	inq101	inq101
5	crnlla	assctv2	brkly7	assctv1

TAB. 6.11 – Classement des 5 meilleurs systèmes de TREC3

TREC5				
Rang	**All**	**Classe 1**	**Classe 2**	**Classe 3**
1	genrl4	uwgcx0	lnadesc2	cor5a1se
2	clclus	uwgcx1	lnadesc1	cor5a2cr
3	genrl3	ethme1	lnmfull2	clclus
4	ethme1	clthes	lnmfull1	brkly17
5	lnmfull2	genrl4	cor5m1le	genrl3

TAB. 6.12 – Classement des 5 meilleurs systèmes de TREC5

TREC6				
Rang	**All**	**Classe 1**	**Classe 2**	**Classe 3**
1	claug	anu6min1	uwmt6a0	claug
2	uwmt6a0	anu6alo1	claug	clrel
3	clrel	claug	clrel	uwmt6a0
4	anu6min1	uwmt6a0	iss97man	iss97man
5	iss97man	lnmshort	anu6min1	anu6min1

TAB. 6.13 – Classement des 5 meilleurs systèmes de TREC6

TREC7				
Rang	**All**	**Classe 1**	**Classe 2**	**Classe 3**
1	clarit98comb	ok7ax	clarit98comb	clarit98comb
2	clarit98clus	ok7am	clarit98clus	clarit98clus
3	ok7ax	bbn1	clarit98r rank	att98atdc
4	tno7exp1	tno7exp1	tno7exp1	att98atde
5	att98atdc	mds98td	inq502	clarit98rank

TAB. 6.14 – Classement des 5 meilleurs systèmes de TREC7

de systèmes utilisés varie (on obtient ainsi différentes versions de MaxProbSeg). Par exemple, best_all(5) représente une version de MaxProbSeg dans laquelle les 5 systèmes ayant obtenus les meilleures probabilités de pertinence sont utilisés. Les versions best_all(n) sont appliquées sur l'ensemble des requêtes sans classification.

Dans la section suivante, nous généralisons notre analyse à toutes les requêtes, et nous effectuons une comparaison des résultats que nous obtenons avec ceux de l'algorithme ProbFuse.

6.4.3 Analyse globale des résultats

Nous comparons dans cette section les performances obtenues par chacun des algorithmes de fusion que nous avons proposés. L'analyse globale suppose qu'aucune classification des requêtes n'est effectuée. Les performances de référence sont celles obtenues

par l'algorithme ProbFuse initialement proposé dans [LTP+06].

Dans un premier temps, nous montrons à l'aide du tableau 6.15 que le nombre optimal de systèmes à sélectionner pour MaxProbSeg correspond à 10. En effet, le tableau 6.15 récapitule pour chaque version[2] de MaxProbSeg les performances obtenues pour chacune des mesures que nous avons utilisées. Les performances présentées dans ce tableau correspondent à la moyenne des performances sur les 10 itérations.

	MAP	R-prec	P5	P10	P15
best_all(5)	0,4080	**0,4417**	0,7363	0,7170	0,6951
best_all(10)	**0,4134**	0,4322	**0,7679**	0,7367	**0,7152**
best_all(15)	0,3827	0,4089	0,7492	**0,7509**	0,7091
best_all(20)	0,3677	0,3946	0,7496	0,7415	0,6921
best_all(30)	0,3662	0,4118	0,7520	0,7274	0,6948
best_all(50)	0,3758	0,4146	0,7511	0,7273	0,6954

TAB. 6.15 – Performances globales des versions de MaxProbSeg

La première colonne indique la version de MaxProbSeg qui est utilisée. Les différentes versions de MaxProbSeg sont obtenues en choisissant un nombre différent de systèmes à fusionner. Ces versions sont déclinées sous le nom best_all(n), avec n systèmes utilisés dans la fusion. Les chiffres en gras indiquent pour une mesure donnée la version MaxProbSeg la plus performante. Par exemple pour la R-précision, la version best_all(5) est la meilleure stratégie, alors que pour la P@10 c'est la version best_all(15) qui est la plus performante. On peut toutefois noter que pour ces deux mesures, la version best_all(10) se classe en deuxième ou troisième position.

En comparant les performances des différentes versions de MaxProbSeg avec celles des 5 meilleurs systèmes de TREC3, mesure par mesure, on obtient les tableaux 6.16 et 6.17. Dans ces tableaux, les systèmes sont classés par performance décroissante. On observe un changement de classement parmi les 5 meilleurs systèmes ayant participé à TREC3 lorsque les versions de MaxProbSeg sont prises en compte. Pour la MAP, best_all(10) se classe en deuxième position et best_all(5) en troisième position. Le système citya1 qui était initialement classé deuxième se retrouve 4ème. Pour ces deux mesures, le système inq102 reste le meilleur système. Notons que les meilleurs systèmes pour chaque mesure sont représentés en gras dans les deux tableaux.

En observant le classement pour les mesures de haute précision, le meilleur système est une version de MaxProbSeg (best_all(10) pour la P@5 et la P@15, best_all(15) pour la P@10). On remarque que pour la P@10 et la P@15, les versions de MaxProbSeg obtiennent de meilleures performances que les 5 meilleurs systèmes d'après le classement de TREC. Même si aucune version de MaxProbSeg ne permet d'améliorer les résultats du meilleur système en terme de MAP et R-précision, on constate que MaxProbSeg

[2]Les versions de MaxProbSeg correspondent dans notre cas l'utilisation d'un nombre différent de systèmes. Par exemple, choisir 5 ou 10 systèmes revient à tester 2 versions de l'algorithme MaxProbSeg.

Systèmes	MAP	Systèmes	R-précision
inq102	**0,4226**	**inq102**	**0,4524**
best_all(10)	0,4134	best_all(5)	0,4417
best_all(5)	0,4080	best_all(10)	0,4322
citya1	0,4012	citya1	0,4217
best_all(15)	0,3827	brkly7	0,4152
best_all(50)	0,3758	best_all(50)	0,4146
brkly7	0,3714	best_all(30)	0,4118
best_all(20)	0,3677	best_all(15)	0,4089
best_all(30)	0,3662	inq101	0,4088
inq101	0,3659	assctv2	0,3999
assctv2	0,3539	best_all(20)	0,3946

TAB. 6.16 – Comparaison des performances des versions de MaxProbSeg et des 5 meilleurs systèmes de TREC3 pour la Map et la R-précision

est plus efficace pour les mesures de haute précision. On remarque par exemple que la meilleure version de MaxProbSeg obtient une amélioration (par rapport au meilleur système) de 1,05% pour la P@5, 4% pour la P@10, et 4,15% pour la P@15. Les mesures de haute précision permettent d'évaluer la capacité du système à restituer le maximum de documents pertinents parmi les premiers documents que l'utilisateur consulte. Ces mesures sont par exemple privilégiées lors de la recherche d'information sur Internet. Dans ce contexte, notre méthode permet de répondre de manière plus satisfaisante (par rapport au meilleur système) à la requête de l'utilisateur.

Systèmes	P@5	Systèmes	P@10	Systèmes	P@15
best_all(10)	0,7679	best_all(15)	0,7509	best_all(10)	0,7152
brkly7	**0,7600**	best_all(20)	0,7415	best_all(15)	0,7091
best_all(30)	0,7520	best_all(10)	0,7367	best_all(50)	0,6954
best_all(50)	0,7511	best_all(30)	0,7274	best_all(5)	0,6951
best_all(20)	0,7496	best_all(50)	0,7273	best_all(30)	0,6948
best_all(15)	0,7492	**inq102**	**0,7220**	best_all(20)	0,6921
inq102	0,7440	best_all(5)	0,7170	**inq102**	**0,6867**
citya1	0,7400	brkly7	0,7120	brkly7	0,6813
best_all(5)	0,7363	citya1	0,7120	citya1	0,6787
citya2	0,7320	citya2	0,6940	citya2	0,6560
assctv2	0,7280	assctv2	0,6760	assctv2	0,6453

TAB. 6.17 – Comparaison des performances des versions de MaxProbSeg et des 5 meilleurs systèmes de TREC3 pour les mesures de haute précision

Notre choix s'est porté sur la version best_all(10) de l'algorithme MaxProbSeg, du fait des performances qu'elle permet d'obtenir par rapport aux autres variantes de l'algorithme.

Nous comparons dans le tableau 6.18 les performances de MaxProb, MaxProbSeg et ProbFuse sur la collection de documents de TREC3. La version best_all(10) de MaxProbSeg est utilisée.

	best_all(10)	MaxProb	ProbFuse
MAP	0,4134 (+13,67%)	0,3516 (-3,32%)	0,3637
R-prec	0,4322 (+11,07%)	0,3889 (-0,06%)	0,3891
P@5	0,7679 (+7,04%)	0,7043 (-1,83%)	0,7174
P@10	0,7367 (+9,93%)	0,6722 (-0,29%)	0,6702
P@15	0,7152 (+12,28%)	0,6367 (-0,04%)	0,6370

TAB. 6.18 – Comparaison des performances de MaxProbSeg, maxProb, versus ProbFuse - TREC3

Dans ce tableau, ProbFuse correspond au système de référence. Les pourcentages indiqués entre parenthèses correspondent à l'amélioration obtenue par rapport aux performances de l'algorithme ProbFuse. D'après ce tableau, l'algorithme MaxProb est moins performant que ProbFuse. On remarque cependant que les différences de performances entre MaxProb et ProbFuse sont faibles. MaxProb permet d'avoir des performances assez proches de celles de ProbFuse sur l'ensemble des requêtes.

La version best_all(10) permet quant à elle d'obtenir des résultats supérieurs à ProbFuse et à MaxProb. On observe par exemple une amélioration de la MAP de 13,67% par rapport à ProbFuse, et de 17,6% par rapport à MaxProb. ProbFuse combine les résultats obtenus par tous les systèmes. Les résultats du tableau 6.18 nous montrent que ce type de combinaison est moins performant que le choix des 10 meilleurs systèmes (meilleurs par rapport à leurs probabilité de pertinence).

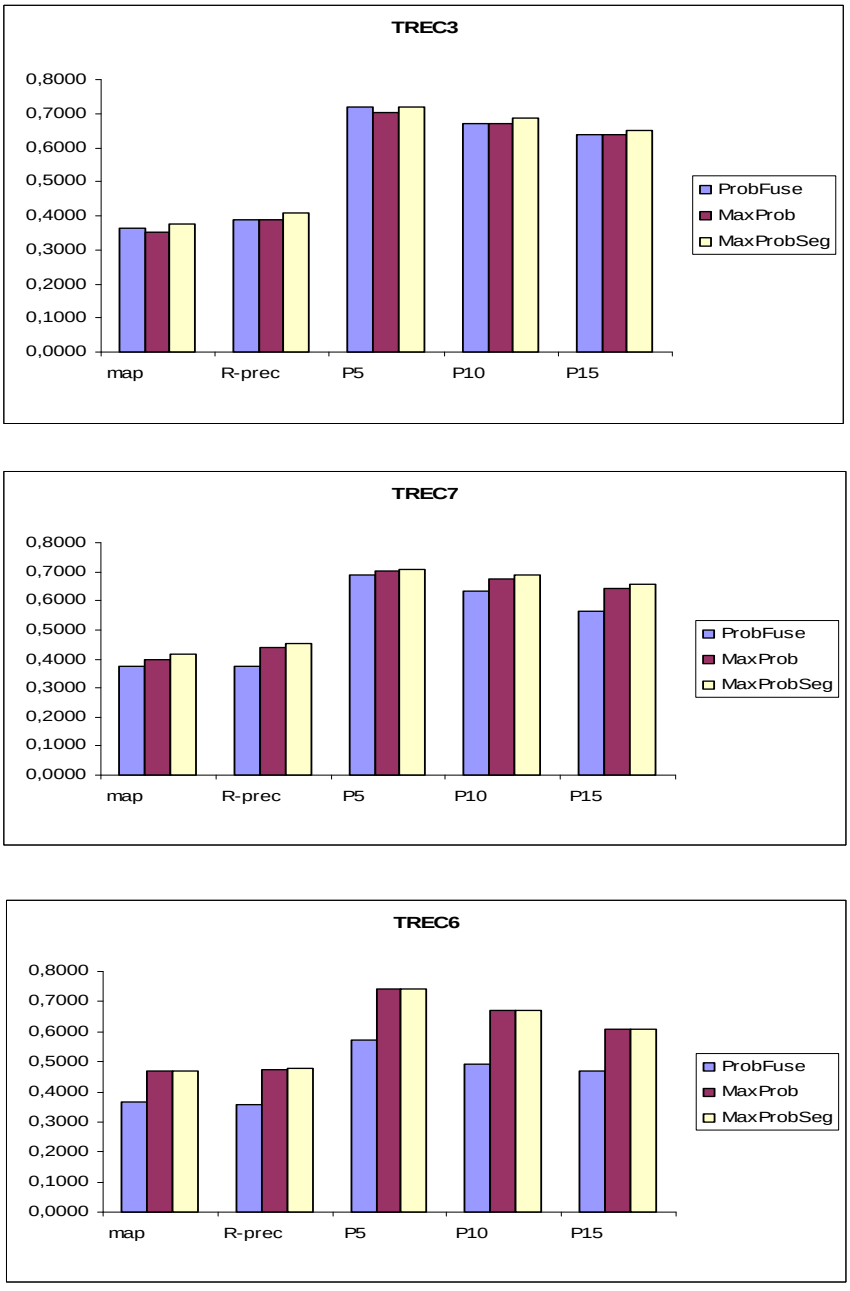

FIG. 6.6 – Comparaison des performances des algorithmes MaxProb (version best_all(10)), MaxProbSeg et ProbFuse pour la collection de TREC3, TREC6 et TREC7

D'après la figure 6.6, les résultats obtenus avec les collections de TREC3 ne se généralisent pas pour TREC6 et TREC7 par exemple. Sur cette figure, MaxProbSeg correspond à sa version best_all(10). Une des conclusions de la comparaison des algorithmes sur la collection de TREC3 indique que MaxProb obtenait des performances plus faibles que ProbFuse. En observant les graphiques de la figure 6.6, on constate que pour TREC6 et TREC7, les algorithmes MaxProb et MaxProbSeg permettent d'obtenir des performances supérieures à celles de ProbFuse. MaxProbSeg permet d'obtenir (avec la version best_all(10)) des performances meilleures que celles obtenues par MaxProb.

Dans le but de vérifier si les différences de performance entre les différents algorithmes que nous avons testés sont significatives, nous avons utilisé un test de significativité que nous présentons dans la section suivante.

6.4.4 Évaluation statistique des résultats

Les tests statistiques permettent de ne pas se limiter à une simple comparaison des performances des systèmes en pourcentage d'augmentation. Plusieurs tests d'hypothèse sont utilisés en RI ([Hul96]) comme par exemple le test de Wilcoxon ([Con80]), le T-test, etc. Ainsi , [Hul96] présente différents tests statistiques de la littérature qui permettent de déterminer si les différences dans les performances entre deux systèmes de RI ou deux méthodes sont significatives. Dans [SZ05], plusieurs tests statistiques sont étudiés et leurs résultats montrent par exemple que le T-test était un test adapté à la RI. Nous avons utilisé le T-test pour évaluer nos résultats.

Dans la logique des tests d'hypothèse comme le T-test, il est nécessaire de définir 2 hypothèses statistiques. La première hypothèse (nommée hypothèse nulle ou H0) définit l'hypothèse de départ. Dans notre cas, H0 suppose qu'*il n'y a pas de différence entre les performances des deux groupes de résultats* que nous analysons. La seconde hypothèse ou H1 suppose que *la différence entre les groupes de résultats est significative*. L'application du T-test sur des ensembles de données permet de calculer la *p-valeur* associée. La *p-valeur* correspond à la probabilité (ou au risque) de commettre une erreur en supposant l'hypothèse H0. En d'autres termes, la p-valeur permet de confirmer ou non l'hypothèse H0 en fonction d'un certain seuil. Dans notre cas, lorsque la p-valeur est supérieure à 0,05 (5%), l'hypothèse de départ est retenue. Dans le cas contraire, l'hypothèse H0 est rejetée au détriment de l'hypothèse H1.

Dans le tableau 6.19, la première ligne nous indique les performances moyennes obtenues respectivement par best_all(10) et ProbFuse, soit 0,4134 pour best_all(10), et 0,3637 pour ProbFuse. La p-valeur de 0,0129 invalide l'hypothèse H0 qui stipule que la différence entre les 2 algorithmes n'est pas significative. En regardant la dernière ligne du tableau, la faible variation qui existe entre la MAP moyenne obtenue avec MaxProb et celle obtenue avec Best_sys laisse penser que la différence entre les 2 algorithmes n'est pas significative. Rappelons que Best_sys correspond à la performance moyenne des meilleurs systèmes pour chacune des 3 classes de requêtes. La p-valeur nous indique cependant l'inverse. Dans le tableau, les valeurs en gras représentent les cas où les différences ne sont pas significatives. La lecture du tableau 6.19 nous montre qu'il

	Valeur moyenne	T-test	
	MAP	t	P-valeur
best_all(10), ProbFuse	0,4134 – 0,3637	3,0914	0,012900
best_all(10), MaxProb	0,4134 – 0,3516	6,0811	0,000183
best_all(10), Best_sys	0,4134 – 0,3896	1,6928	**0,124700**
ProbFuse, MaxProb	0,3637 – 0,3516	0,8379	**0,423800**
ProbFuse, Best_sys	0,3637 – 0,3896	-1,6212	**0,139400**
MaxProb, Best_sys	0,3516 – 0,3896	-2,9231	0,016940

TAB. 6.19 – Comparaison des algorithmes avec le T-test (MAP)

existe une différence significative entre la version best_all(10) de MaxProbSeg et les 2 algorithmes MaxProb et ProbFuse. La différence entre les performances en terme de MAP de la version best_all(10) et de best_sys n'est par contre pas significative.

Les mêmes interprétations peuvent être faites sur les tableaux 6.20, 6.21, 6.22, et 6.23.

	Valeur moyenne	T-test	
	R-précision	t	p-valeur
best_all(10), ProbFuse	0,4322 – 0,3891	2,8592	0,018800
best_all(10), MaxProb	0,4322 – 0,3889	5,7695	0,000270
best_all(10), Best_sys	0,4322 – 0,4209	0,8675	**0,408200**
ProbFuse, MaxProb	0,3891 – 0,3889	0,0201	**0,984400**
ProbFuse, Best_sys	0,3891 – 0,4209	-2,6219	0,027730
MaxProb, Best_sys	0,3889 – 0,4209	-2,9808	0,015430

TAB. 6.20 – Comparaison des algorithmes avec le T-test (R-précision)

	Valeur moyenne	T-test	
	P@5	t	p-valeur
best_all(10), ProbFuse	0,7679 – 0,7174	2,2281	**0,052860**
best_all(10), MaxProb	0,7679 – 0,7043	2,1876	**0,056480**
best_all(10), Best_sys	0,7679 – 0,7179	1,5151	**0,164100**
ProbFuse, MaxProb	0,7174 – 0,7043	0,6547	**0,529100**
ProbFuse, Best_sys	0,7174 – 0,7179	-0,0264	**0,979500**
MaxProb, Best_sys	0,7043 – 0,7179	-0,5706	**0,582200**

TAB. 6.21 – Comparaison des algorithmes avec le T-test (P@5)

	Valeur moyenne	T-test	
	P@10	t	p-valeur
best_all(10), ProbFuse	0,7367 – 0,6702	2,2421	**0,051670**
best_all(10), MaxProb	0,7367 – 0,6722	2,3615	0,042500
best_all(10), Best_sys	0,7367 – 0,6364	3,5055	0,006600
ProbFuse, MaxProb	0,6702 – 0,6722	-0,0752	**0,941700**
ProbFuse, Best_sys	0,6702 – 0,6364	1,3985	**0,195500**
MaxProb, Best_sys	0,6702 – 0,6364	1,8057	**0,104400**

TAB. 6.22 – Comparaison des algorithmes avec le T-test (P@10)

	Valeur moyenne	T-test	
	P@15	t	p-valeur
best_all(10), ProbFuse	0,7152 – 0,6370	2,5487	0,031260
best_all(10), MaxProb	0,7152 – 0,6367	4,3472	0,001858
best_all(10), Best_sys	0,7152 – 0,5905	4,2107	0,002271
ProbFuse, MaxProb	0,6370 – 0,6367	0,0105	**0,991900**
ProbFuse, Best_sys	0,6370 – 0,5905	1,3515	**0,209500**
MaxProb, Best_sys	0,6367 – 0,5905	2,275	0,048960

TAB. 6.23 – Comparaison des algorithmes avec le T-test (P@15)

6.5 Conclusion

Dans ce chapitre, nous avons proposé 2 algorithmes de fusion probabilistes MaxProb et MaxProbSeg. Dans ces algorithmes, un ensemble de probabilités de pertinence est associé à chaque segment dans les listes de documents restitués par les systèmes. Ces probabilités sont déterminées en fonction des résultats que les systèmes ont obtenus dans le passé en réponse à un ensemble de requêtes. Les résultats passés font référence aux résultats des systèmes lors de la phase d'apprentissage. Nous avons mesuré les performances des algorithmes à travers 5 mesures (la MAP, la R-précision, la P@5, P@10, et P@15).

Le premier algorithme MaxProb part du principe que les documents pertinents sont distribués différemment d'une liste de documents à une autre. Le principe de la segmentation que l'algorithme proposé par Lillis et ses collègues [LTP+06] nous a inspiré, permet de découper une liste de documents en plusieurs segments, afin d'étudier sur un ensemble de requêtes la répartition des documents pertinents. Le premier apport de MaxProb est lié à la sélection des systèmes intervenant lors de la fusion. En effet, le système qui obtient la meilleure probabilité de pertinence pour un segment donné est utilisé pour restituer les documents pertinents de ce segment lors de la phase de test. À travers les expérimentations qui ont été effectuées, MaxProb s'avère plus performant que la stratégie best_sys que nous avons présentée dans le chapitre 4. Les résultats montrent par exemple qu'une dizaine de systèmes sont sélectionnés sur un total de 42

systèmes pour répondre à une requête de test. Cependant, les résultats obtenus par MaxProb et best_sys montrent une différence assez faible pour les mesures de hautes précisions. L'explication est que les systèmes retrouvent en moyenne la même quantité de documents pertinents parmi les 15 premiers documents.

Un des inconvénients de MaxProb est que tous les documents d'un même segment ont la même probabilité de pertinence. Cependant, il est clair que la position des documents dans une liste témoigne de l'importance accordée à ce document. Pour pallier cette situation où l'agencement des documents dans les listes n'influence pas la qualité de la recherche, MaxProbSeg calcule pour chaque document situé dans un segment un score de pertinence. Ce score est calculé en fonction de la position du document considéré dans les listes restituées par N systèmes. Le choix des N systèmes est une particularité de l'algorithme. En effet, seuls les N systèmes qui ont obtenu les meilleures probabilités de pertinence moyennes lors de l'apprentissage sont sélectionnés. Les probabilités moyennes sont calculées sur l'ensemble des segments. Nous avons testé à travers des expérimentations, l'impact du nombre de systèmes sélectionnés sur les résultats de la fusion. Les résultats confirment que le choix d'un nombre maîtrisé de systèmes est plus bénéfique pour la fusion que la prise en compte de tous les systèmes. On aurait pu croire que lorsque l'on restreint la fusion aux 5 meilleurs systèmes, les résultats en terme de P@5 sont meilleurs que ceux obtenus avec plus de 5 systèmes. Cependant, les résultats montrent que la sélection des 10 meilleurs systèmes (best_all(10)) est plus performante que celle des 5 meilleurs systèmes. Les tests statistiques nous indiquent qu'il n'existe pas de différence significative entre les résultats des algorithmes MaxProb, MaxProbSeg, et ProbFuse lorsque la P@5 est considérée. Pour la MAP et la R-précision, la différence est significative entre les résultats de best_all(10) et les deux algorithmes ProbFuse et MaxProb. Les meilleurs résultats sont obtenus avec la sélection des 10 meilleurs systèmes (en utilisant leur probabilité de pertinence moyenne). La conclusion est qu'il n'y a pas de lien particulier entre les N systèmes sélectionnés et les N premiers résultats de chaque algorithme. Les résultats obtenus par MaxProbSeg sont meilleurs que ceux des algorithmes MaxProb et ProbFuse comme en témoignent les tests statistiques effectués.

Les résultats que nous avons obtenus montrent que la prise en compte du contexte linguistique de la requête permet d'améliorer les performances de la recherche. Les algorithmes que nous avons proposés restent toutefois applicables dans d'autres contextes où les requêtes ne seraient pas classées par exemple.

Plusieurs perspectives font suite à nos travaux. Une des perspectives que nous avons commencé à explorer consiste à appliquer une technique d'analyse canonique sur les données dont nous disposons (les CL des requêtes, les performances des systèmes, les collections de documents, etc). À travers cette étude, nous souhaitons analyser les relations qui existent entre toutes ces données afin de pouvoir proposer des mécanismes de fusion adaptative plus efficaces. Nous présentons, dans le chapitre suivant, les travaux en cours que nous avons entrepris concernant l'analyse canonique. Ce travail ouvre une nouvelle voie de recherche et nous a permis d'établir une collaboration avec des collègues

mathématiciens dans le cadre d'un projet nommé ACRIC (Analyse Canonique et RI Contextuelle).

Chapitre 7

Analyse Canonique et RI

7.1 Introduction

Les travaux présentés dans cette thèse ont pour objectif de prendre en compte certains éléments contextuels (contexte des requêtes, contexte des SRI), pour proposer un processus de recherche qui soit capable de s'adapter à ces différents contextes. Dans l'optique de proposer un processus fiable, nous abordons une des perspectives faisant suite à nos travaux. Il s'agit de mettre en évidence les relations qui existent entre ces différents contextes.

Nous disposons d'un grand nombre de données que nous exploitons dans le cadre de nos travaux. En effet, un premier jeu de données que nous utilisons est constitué des 13 CL que nous avons extraites des requêtes. Nous avons basé nos expérimentations sur la campagne d'évaluation TREC. Pour chaque requête, les performances des systèmes sont évaluées suivant un certain nombre de mesures comme la MAP, la R-précision, etc. Nous utilisons dans les données de 5 années consécutives de TREC (de TREC3 à TREC7), qui correspondent à 50 requêtes par année et à 27 mesures de performance. Nous souhaitons rajouter un autre jeu de données pour décrire les systèmes. Pour récapituler, nous disposons de 13 CL associées à 250 requêtes.

Nous utilisons pour arriver à nos fins une méthode statistique d'analyse multidimensionnelle – l'analyse canonique (AC). Notre objectif est d'arriver dans un premier temps à analyser les relations qui existent entre les CL et les performances des systèmes. Puis, nous comptons intégrer une troisième dimension dans l'analyse (description des systèmes). Nous présentons le principe de l' AC dans la section 7.2. Nous terminons ce chapitre en donnant quelques résultats d'expérimentation en cours (section 7.3).

7.2 L'analyse canonique

L'analyse canonique (AC) [Hot36] est une méthode de statistique descriptive multidimensionnelle dont le but est d'étudier les dépendances entre deux groupes de variables numériques, c'est-à-dire de déterminer les corrélations existant entre ces deux groupes.

L'étude des relations entre deux groupes de variables constitue la principale particularité de l' AC par rapport aux méthodes statistiques telles que l'ACP, la classification, ... L'analyse canonique est une méthode symétrique, c'est à dire que la méthode permet de permuter les groupes de variables sans modifier les résultats.

7.2.1 Méthodologie générale de l'AC

Comme nous l'avons précisé précédemment, l' AC recherche des combinaisons linéaires des 2 groupes de variables. Soient X et Y les 2 groupes de variables comme le montre la figure 7.1.

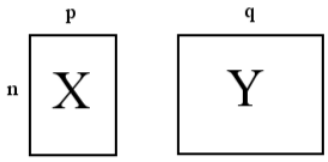

FIG. 7.1 – Groupes de variables à étudier avec l'AC

Dans la figure 7.1, X et Y sont représentés respectivement par deux matrices de dimension *nxp* et *nxq*. Le nombre de variables du groupe X est noté p (respectivement q pour le groupe Y) et le nombre d'individus correspond à n. La méthode de l' AC consiste à rechercher successivement des couples de variables canoniques tels que la corrélation canonique soit maximale. Une variable canonique représente un vecteur obtenu dans l'espace associé à chaque groupe de variables (X et Y), et correspond à la notion de facteurs dans l'analyse en composantes principales (ACP). Un couple de variables canoniques est composé des variables issues des 2 groupes à analyser. Chaque couple correspond à la combinaison linéaire des variables de son groupe. Le processus de sélection des couples de variables canoniques est répété, avec la contrainte que le nouveau couple de variables canoniques ne doit pas être corrélée avec les précédentes. De telles combinaisons linéaires, si elles existent, sont uniques. Comme en ACP, on peut tracer le cercle des corrélations sur le graphique des variables, ce qui en facilite l'interprétation (dont le principe est le même que pour le graphique des variables en ACP). La figure 7.2 nous donne un aperçu de la représentation graphique des variables. Notons qu'il est aussi possible d'établir une représentation graphique des individus pour éventuellement compléter l'interprétation des résultats qui sont obtenus.

Soit d le nombre de couples de variables obtenus. Cette valeur représente la dimension retenue pour la représentation graphique. Les couples (V^s, W^s), avec $s = (1, \ldots, d)$, sont les couples de variables canoniques retenus. On posera $\rho_s = \mathrm{Cor}(V^s, W^s)$ et on appellera *corrélations canoniques* les coefficients ρ_s qui sont, par construction, décroissants $(1 >= \rho_1 >= \rho_2 >= \ldots >= \rho_p)$.

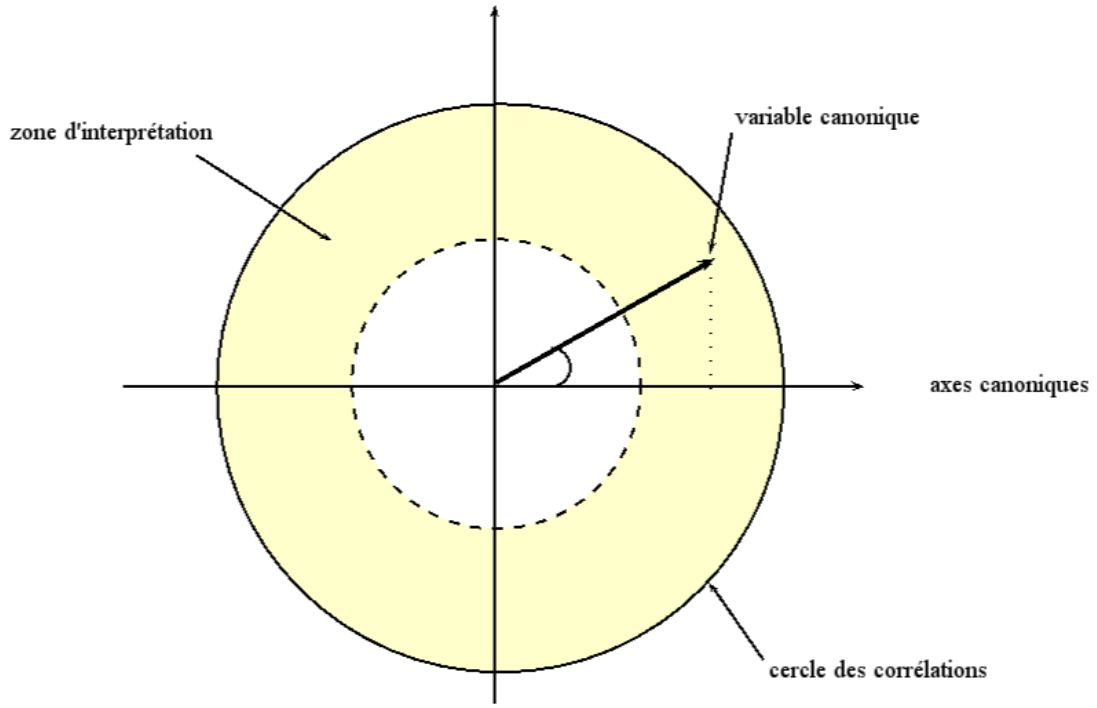

FIG. 7.2 – Graphiques des variables en AC

7.2.2 Aspects mathématiques

Cette section donne les éléments mathématiques de l'AC. Nous appelons $F = \mathbb{R}^n$ l'espace des variables. À chaque variable X^j est associé un vecteur unique x^j de F dont les coordonnées sont x_i^j $(i = 1, \ldots, n)$. De même, à chaque variable Y^k est associé un vecteur unique y^k de F, de coordonnées y_i^k. On peut ainsi définir dans F deux sous-espaces vectoriels : F_X, engendré par les vecteurs x^j $(j = 1, \ldots, p)$, en général de dimension p, et F_Y, engendré par les vecteurs y^k $(k = 1, \ldots, q)$, en général de dimension q.

En AC, la contrainte principale est que le nombre d'observations (individus) doit être supérieur au nombre de variables, soit n> p+q (cf. figure 7.1 pour les notations). Si le nombre de variables est trop important, on peut utiliser une technique de régularisation pour permettre l'inversion de la matrice dite (mal conditionnée, voire singulière) intervenant dans les projections qui sont au coeur de la méthode. Cette contrainte souligne la principale limite de l' AC. Dans le cas où la matrice est mal conditionnée, un algorithme de validation croisé est utilisé pour déterminer de manière optimale les paramètres de régularisation.

Pour reprendre formellement le principe de l' AC, un premier couple de variables (V^1, W^1), correspondant respectivement à une combinaison linéaire des variables X^j

(respectivement des variables Y^k), est choisi, tel que les variables V^1 et W^1 soient le plus corrélées possible. Le deuxième couple de variables (V^2, W^2) est choisi tel que V^2 (respectivement W^2) soit non corrélé à V^1 (respectivement W^1), tout en étant les plus corrélées entre elles, et ainsi de suite ....

Le passage d'un sous-espace vectoriel à un autre se fait par le biais de projecteurs. Les matrices des projecteurs sur les sous-espaces F_X et F_Y sont notées $\mathbf{P_X}$ et $\mathbf{P_Y}$. Les matrices sont déterminées à travers les formules 7.1 et 7.2.

$$\mathbf{P_X} = \mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'. \tag{7.1}$$

$$\mathbf{P_Y} = \mathbf{Y}(\mathbf{Y}'\mathbf{Y})^{-1}\mathbf{Y}'. \tag{7.2}$$

$\mathbf{X}'$ (respectivement $\mathbf{Y}'$) désigne la matrice transposée de $\mathbf{X}$ (respectivement $\mathbf{Y}$). On peut alors montrer la propriété ci-dessous.

Propriété 1 *Les vecteurs V^s sont les vecteurs propres normés de la matrice $\mathbf{P_X P_Y}$ respectivement associés aux valeurs propres λ_s rangées par ordre décroissant (ces valeurs propres sont comprises entre 1 et 0). De même, les vecteurs W^s sont les vecteurs propres normés de la matrice $\mathbf{P_Y P_X}$ respectivement associés aux mêmes valeurs propres λ_s.*

Cette propriété permet l'utilisation d'un algorithme standard de recherche des vecteurs propres d'une matrice pour déterminer les variables canoniques V^s et W^s.

7.3 Quelques expérimentations en cours

Les résultats que nous présentons dans cette section sont issus des analyses préliminaires que nous avons effectuée dans le cadre du projet ACRIC. Nous cherchons à travers cette étude faire analyser les corrélations qui existent entre les différentes mesures d'évaluation utilisée dans la campagne d'évaluation TREC (grâce au programme *trec_ eval*) afin de déterminer les principales mesures à utiliser.

Les 27 mesures de départ que nous utilisons sont obtenues par l'intermédiaire de trec_eval. Leur description est donnée dans le tableau 7.1

7.3.1 Analyse de la corrélation entre les mesures de performance

Les mesures que nous avons introduites dans la section précédente sont souvent utilisées en RI. Cependant, certaines mesures sont corrélées entre elles et ne doivent pas toutes être utilisées. Nous donnons dans la figure 7.3 une image de la matrice de corrélation entre les 27 mesures.

Dans la figure 7.3, la bande de couleur représente le degré de corrélation entre les mesures. L'intensité de la couleur indique une forte corrélation (positive ou négative) des mesures. Dans cette figure, on peut remarquer les corrélation en regardant les blocs

Mesure	Description
map	Moyenne des précisions moyennes
R-prec	précision lorsque R documents pertinents sont retrouvés
bpref	Préférence binaire des R premiers documents jugés non pertinents
recip_rank	Rang des premiers documents
ircl_prn.0.00	Rappel ou précision interpolé à un taux de rappel égal à 0
ircl_prn.0.10	Rappel ou précision interpolé à un taux de rappel égal à 10%
ircl_prn.0.20	Rappel ou précision interpolé à un taux de rappel égal à 20%
ircl_prn.0.30	Rappel ou précision interpolé à un taux de rappel égal à 30%
ircl_prn.0.40	Rappel ou précision interpolé à un taux de rappel égal à 40%
ircl_prn.0.50	Rappel ou précision interpolé à un taux de rappel égal à 50%
ircl_prn.0.60	Rappel ou précision interpolé à un taux de rappel égal à 60%
ircl_prn.0.70	Rappel ou précision interpolé à un taux de rappel égal à 70%
ircl_prn.0.80	Rappel ou précision interpolé à un taux de rappel égal à 80%
ircl_prn.0.90	Rappel ou précision interpolé à un taux de rappel égal à 90%
ircl_prn.1.00	Rappel ou précision interpolé à un taux de rappel égal à 100%
P@5	Précision lorsque 5 documents ont été retrouvés
P@10	Précision lorsque 10 documents ont été retrouvés
P@15	Précision lorsque 15 documents ont été retrouvés
P@20	Précision lorsque 20 documents ont été retrouvés
P@30	Précision lorsque 30 documents ont été retrouvés
P@100	Précision lorsque 100 documents ont été retrouvés
P@200	Précision lorsque 200 documents ont été retrouvés
P@500	Précision lorsque 500 documents ont été retrouvés
P@1000	Précision lorsque 1000 documents ont été retrouvés

Tab. 7.1 – Description des 27 mesures utilisées

de couleur. Ainsi par exemple, le bloc P@5, P@10, P@15, P@20 et P@30 est très corrélé. On remarque aussi une corrélation entre la F-mesure et la précision par exemple. En appliquant une ACP sur les mesures de performance, on obtient le résultat présenté dans la figure 7.4(a) sur les 2 axes principaux.

On remarque que conformément à la matrice de corrélation, les mesures P@5, P@10, P@15, P@20, et P@30 sont corrélées. La même observation peut être faite dans la figure 7.4(b).

L'analyse des corrélations entre les mesures permet de réduire le nombre de mesures qui seront utilisées, tout en gardant celles qui sont les plus représentatives des blocs de mesures corrélées. On pourra par exemple choisir la P@5 ou la P@10 pour représenter le bloc P@5, P@10, P@15, P@20, et P@30. Le même type de substitution peut être fait sur les autres mesures.

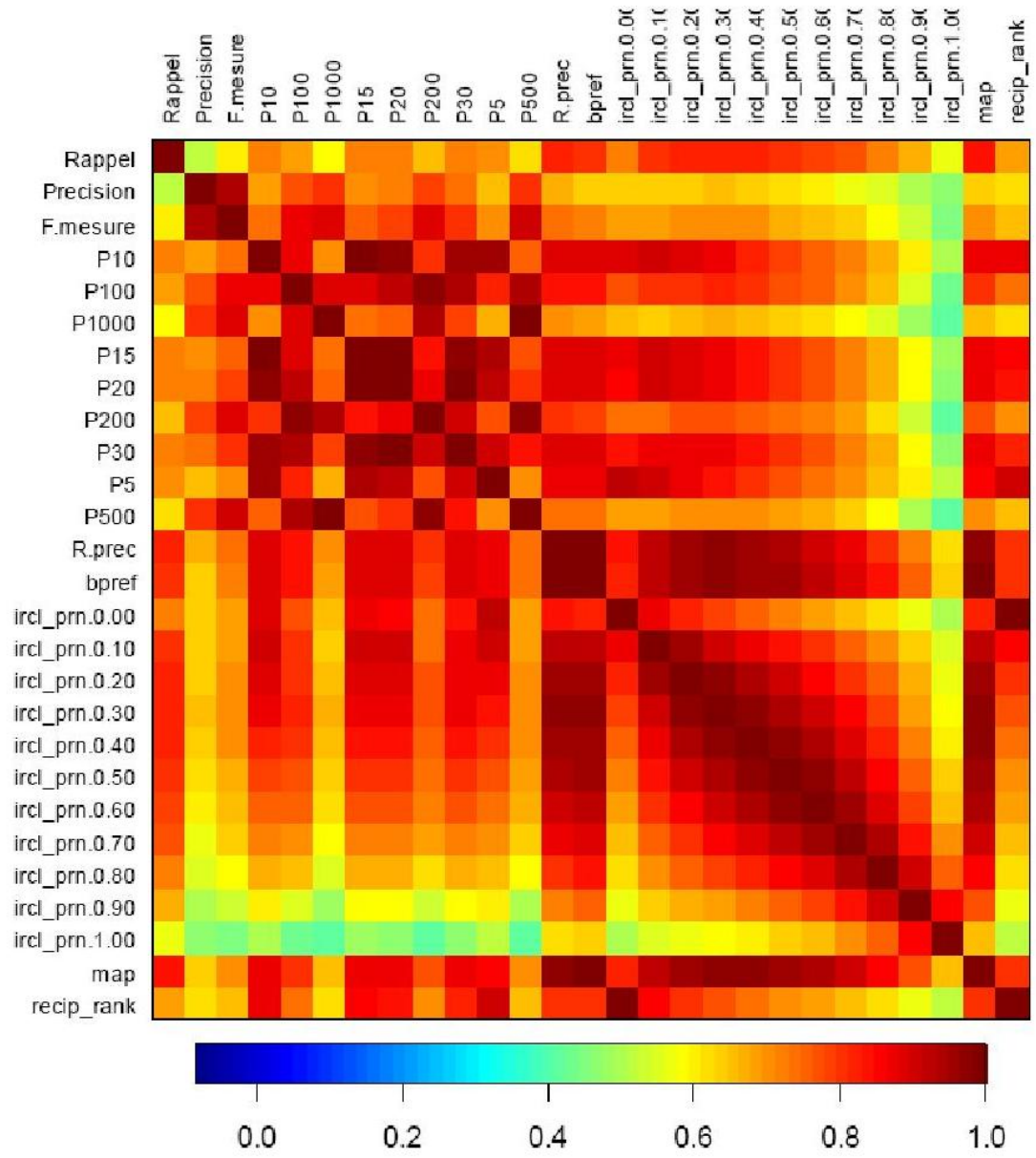

FIG. 7.3 – Matrice de corrélation entre les différentes mesures

7.3.2 Étude des relations entre les CL et les performances des systèmes

Nous présentons quelques résultats préliminaires que nous avons obtenus lors de l'analyse des relations entre les CL et les performances des systèmes. Les performances des systèmes sont mesurées en terme de rappel. Le résultat de l' AC est illustré dans la

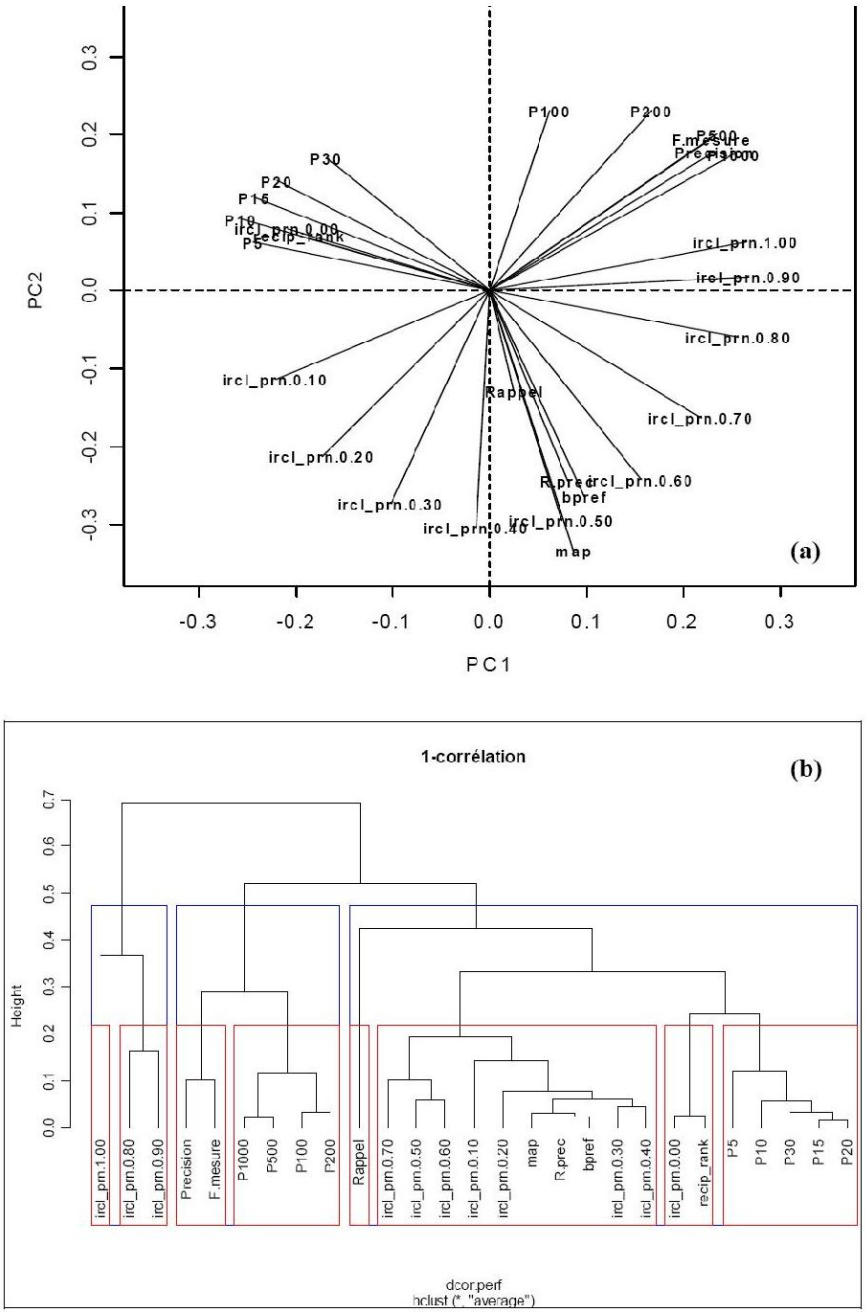

FIG. 7.4 – ACP et CAH appliquées sur les mesures de performances

figure 7.5.

Le cercle des corrélations nous indique un lien entre le rappel du système umd98a2

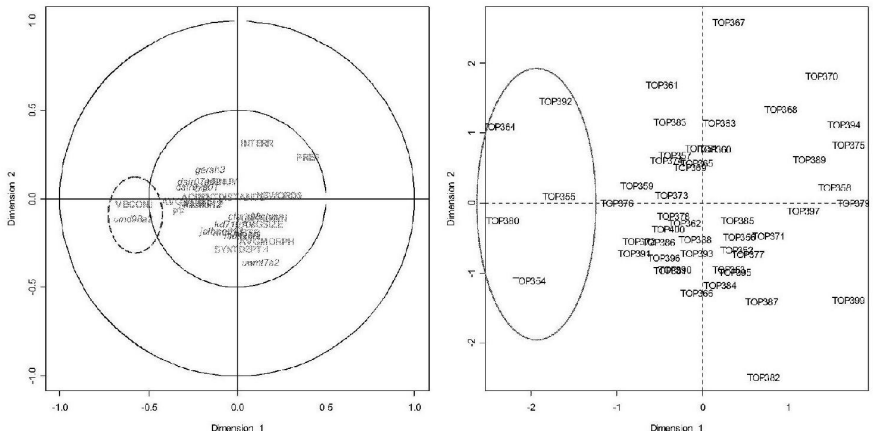

FIG. 7.5 – AC appliquée aux performances des systèmes et aux CL des requêtes

et la CL VBCONJ. Le système de l'université de Maryland (umd98a2) utilise le titre, la description et la partie narrative des requêtes TREC. L'association entre umd98a2 et VBCONJ signifie que le système obtient des mesures de rappel élevées et que les requêtes pour lesquelles ce taux est élevé contiennent un grand nombre de verbes conjugués. Dans la figure 7.5, on remarque que la répartition des individus peut être rapprochée du cercle des corrélations. Ainsi, les requêtes qui sont entourées sur la figure peuvent être associées (d'après leur position) à la corrélation umd98a2/VBCONJ.

La figure 7.6 vérifie que umd98a2 possède un nombre élevé de verbes conjugués pour les requêtes considérées.

7.4 Conclusion

Les expérimentations sur l' AC ne sont encore qu'à la phase initiale du processus. Toutefois, nous pouvons déjà avoir une idée du type d'analyses que nous pouvons effectuer sur nos données. Grâce à l' AC, nous comptons effectuer des analyses plus complètes sur nos données et comprendre par exemple l'impact des caractéristiques des systèmes sur les performances qu'ils obtiennent.

Nos travaux s'inscrivent dans le cadre de l'étude des variabilités qui interviennent dans la RI. Les hypothèses de variabilités de la RI sont nombreuses, et les solutions envisagées sont toutes aussi nombreuses. Le couplage AC/RI est une nouvelle orientation en RI que nous souhaitons prendre, car nous pensons que la compréhension et l'analyse des différentes relations qui existent entre les données que nous manipulons, justifient le choix de méthodes plus efficaces et adaptées à la recherche. La technique de l'AC est une technique statistique parmi tant d'autres qui permet une exploration multidimensionnelle des données. Nous pensons que les apports de cette technique seront bénéfiques pour la recherche dans un contexte adaptatif.

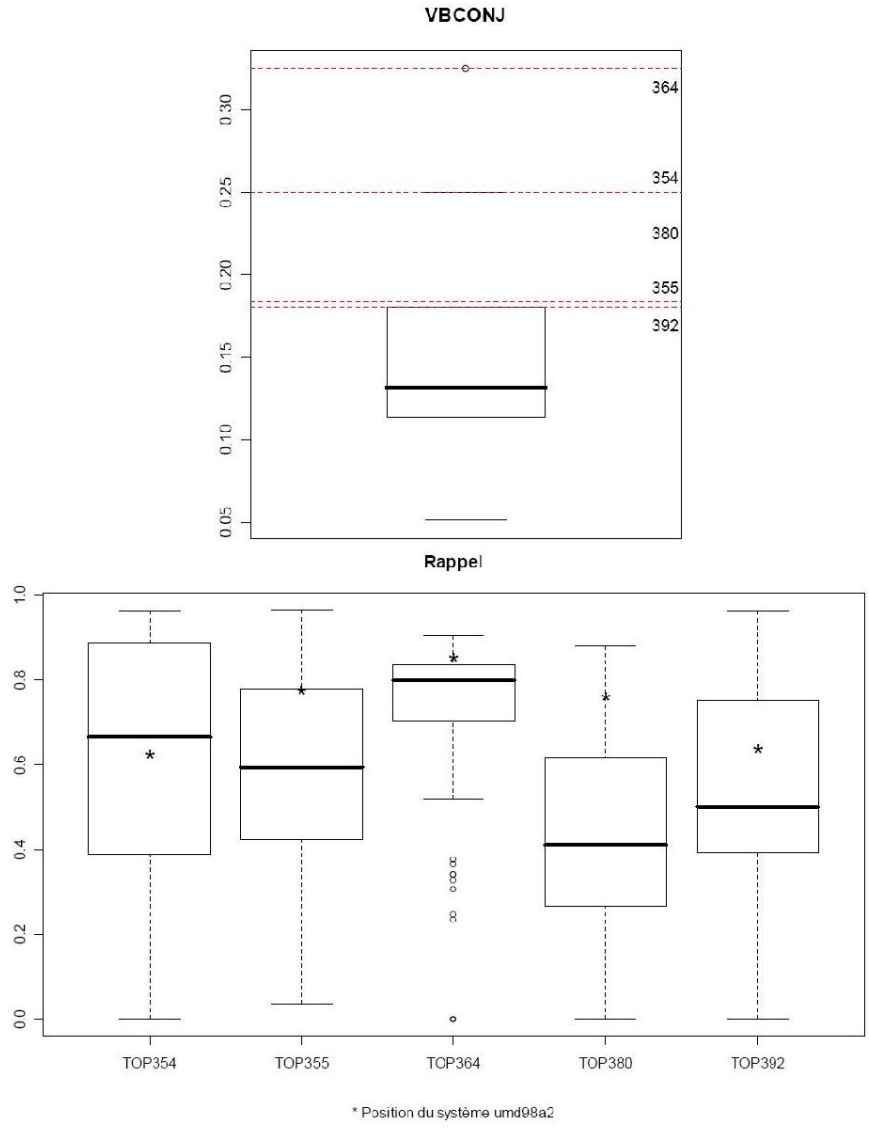

FIG. 7.6 – AC appliquée aux performances des systèmes et aux CL des requêtes

Chapitre 8

Conclusion générale

Synthèse

Les travaux que nous avons présentés s'inscrivent dans le cadre de la RI textuelle non structurée.

Des SRI ont pour objectif de faciliter la recherche d'information d'un utilisateur. Un certain nombre de mécanismes sont nécessaires à la mise en place d'une recherche d'information.

Les SRI mettent en place un mécanisme d' indexation qui a pour but de leur permettre d'avoir une représentation synthétique du contenu de la collection de documents à laquelle ils accèdent. À l'issue de cette indexation, les SRI disposent des descripteurs des documents. L'utilisateur soumet alors son besoin d'information sous forme de requête au SRI . Grâce au modèle de recherche implémenté par le SRI , une mise en correspondance est effectuée entre la requête de l'utilisateur et la collection de documents. Le SRI restitue à l'utilisateur une liste de documents supposés pertinents.

Un certain nombre de variabilités existent dans le processus de RI. Ces variabilités concernent aussi bien l'expression de la requête, les résultats obtenus par les systèmes, et les collections de documents utilisés. Nous proposons dans nos travaux de prendre en compte ces variabilités afin de proposer des techniques de recherche pouvant être intégrées dans un processus de recherche s'adaptant aux différents contextes. Nos propositions se découpent en 4 grandes parties que nous avons validées à travers une série d'expérimentations. Dans toutes nos propositions, nous disposons des résultats obtenus par plusieurs systèmes lors de leur participation aux campagnes TREC.

La première partie de notre contribution concerne la variabilité des requêtes en RI [KMDB07b]. Un des objectifs de nos travaux est de tenir compte du contexte de la requête (exprimé à travers une analyse linguistique des requêtes) pour proposer la meilleure stratégie de recherche à adopter. Les requêtes ne sont plus considérées individuellement, mais plutôt comme faisant partie d'un groupe de requêtes possédant des CL similaires. Notre hypothèse de travail est donnée ci-dessous :

Hypothèse : *Si pour chaque requête il est possible de l'associer à une catégorie de requêtes en fonction de ses CL, et si pour chaque SRI il est possible de déterminer la(les) catégorie(s) de requête(s) pour lesquelles il est le plus performant, alors le processus adaptatif que nous proposons devrait être capable de choisir en fonction de la catégorie de la requête les meilleures stratégies (ou techniques de recherche) à utiliser.*

Nous avons utilisé un ensemble de 13 CL automatiquement extraites des requêtes. Ces CL ont été initialement proposées dans [MT05]. Les requêtes que nous avons utilisées proviennent des campagnes TREC. Nous disposons de 200 requêtes sur lesquelles nous appliquons des techniques statistiques de classification de données, après avoir extrait pour chaque requête 13 CL. Trois classes de requêtes sont déterminées. Nous utilisons aussi 5 mesures de performance issues de TREC (MAP, R-précision, P@5, P@10, et P@15). À travers une série d'expérimentations, nous vérifions si le principe de la classification linguistique des requêtes permet d'obtenir des résultats intéressants. Lors de la phase d'entraînement, nous proposons d'utiliser le système obtenant la meilleure performance sur une classe de requête, et pour une mesure donnée. Ce système peut être différent d'une classe à l'autre et d'une mesure à l'autre. Les résultats obtenus par le système recommandé par notre méthode montrent une amélioration (par rapport aux performances du meilleur système) de 9,22% pour la MAP, 6,36% pour la R-précision, 4,66% pour la P@5, 4,4% pour la P@10.

Dans la deuxième partie de notre contribution, nous nous sommes intéressé à la variabilité des résultats des systèmes. La de données [FS94] est une technique utilisée en RI permettant de combiner les résultats de plusieurs systèmes afin d'améliorer les résultats de la recherche. Nous proposons une technique de fusion facile à mettre en œuvre, et nous l'appliquons aux collections de la tâche "détection de passages" de TREC . Dans ces collections, les jugements de pertinence sont binaires et on ne dispose pas de score de pertinence pour les documents. Notre méthode de fusion est basée sur deux opérateurs de fusion à savoir l' union et l' intersection. Pendant nos expérimentations, nous avons analysé deux types de fusion.

Le premier type de fusion, ou fusion systématique, réalise la fusion par union ou par intersection des listes restituées par les systèmes, sans tenir compte de critères supplémentaires. Les meilleurs systèmes sont déterminés à partir de leur performance passée pour une mesure donnée. Deux stratégies consistant d'une part (stratégie1) à sélectionner les meilleurs systèmes sur la base de leur performance en termes de F-mesure, et d'autre part (stratégie2) à sélectionner pour chaque mesure les meilleurs systèmes, ont été testées. Les deux stratégies que nous proposons permettent d'augmenter la valeur du rappel et de la précision, aussi bien pour la fusion par union que par intersection. Ainsi, nous obtenons des améliorations allant de 0,9% pour la précision en 2003 avec la stratégie1, à 131,8% pour le rappel en 2002 avec la stratégie2. Une analyse plus ciblée nous indique que la stratégie2 est meilleure que la stratégie1, quels que soient le type de fusion et la mesure. Les résultats montrent par exemple en 2002 que la stratégie2 obtient une amélioration de 24,2% pour le rappel et une amélioration de 6,66% en 2003,

par rapport à la stratégie1. Pour la fusion par intersection la stratégie2 permet d'obtenir en moyenne 1,17% d'amélioration par rapport à la stratégie1 pour la précision en 2002. En 2003, la mesure de l'amélioration nous indique 18,57% pour la précision lorsque la stratégie2 est utilisée à la place de la stratégie1. L'issue de la fusion par union et par intersection est connue. La fusion par union devrait permettre d'augmenter le rappel et la fusion par intersection devrait permettre d'améliorer la précision. De plus, il est bien connu que'appliquer une méthode qui augmente le rappel dégrade généralement la précision et inversement. Notre contribution à travers nos expérimentations consiste à donner une indication quantitative sur les améliorations obtenues à l'issue de la fusion. Cependant, rappelons que les stratégies que nous avons proposées nécessitent la connaissance préalable des performances des systèmes, donc du nombre de documents pertinents dans la collection utilisée ainsi que des scores des systèmes. Ces données sont disponibles dans le cadre de la campagne TREC sur laquelle nous nous sommes basé pour nos expérimentations. Nous avons également mesuré l'impact de la fusion par union et par intersection, lorsqu'aucune connaissance *a priori* sur les systèmes n'est disponible. Cette méthode de fusion, simple à implémenter, est applicable dans des conditions d'exploitation réelles. Les résultats montrent que les performances initiales des systèmes subissent une augmentation de 49,82% pour le rappel après application de la fusion par union en 2002 (48,74% en 2003). Pour la précision, l'amélioration des performances initiales est de l'ordre de 31,36% en 2002 avec la fusion par intersection (14,58% en 2003).

Le deuxième type de fusion analyse l'impact de la typologie des requêtes, ainsi que le chevauchement des listes de documents restitués lors de la fusion avec la stratégie2. Lors de la précédente expérimentation, les requêtes jouaient toutes le même rôle. Nous avons étudié une technique de combinaison moins systématique en analysant les requêtes en fonction de leur typologie (*faciles*, *intermédiaires* et *difficiles*), ainsi que le taux de chevauchement des documents pertinents restitués par les systèmes. À travers cette étude, nous souhaitons prendre en compte d'autres critères afin de mieux affiner la fusion. La typologie des requêtes est définie en fonction de leur caractère *facile*, *difficile*, ou *intermédiaire*. Une requête difficile est une requête pour laquelle la précision moyenne des systèmes est faible [DHMO04]. Les requêtes faciles correspondent aux requêtes pour lesquelles la moyenne des précisions est élevée. Enfin, les requêtes dites intermédiaires sont celles pour lesquelles la moyenne des précisions est la plus proche de la moyenne des précisions moyennes. Une des conclusions que l'on peut tirer de nos expérimentations est que, contrairement à ce que l'on pourrait penser, il est plus facile d'améliorer les résultats des requêtes faciles que ceux des requêtes difficiles sous certaines conditions. Par exemple, l'amélioration du rappel lors de la fusion par union des requêtes faciles est de l'ordre de 53,90% en 2002 et de 25,54% en 2003. L'augmentation maximale du rappel est de 53,90% pour les requêtes faciles, et de 97,12% pour les requêtes difficiles avec la fusion par union. La fusion par intersection permet d'améliorer les performances moyennes des systèmes à hauteur de 11,48% pour les requêtes faciles, et jusqu'à 15,60% pour les requêtes difficiles.

L'analyse du taux de chevauchement complète celle effectuée sur la typologie des requêtes. Le taux de chevauchement dans les listes restituées par les systèmes permet de mesurer à quel degré les systèmes s'accordent sur les documents qu'ils jugent pertinents. Les résultats que nous avons obtenus par rapport au taux de chevauchement montrent qu'en moyenne le taux de chevauchement est plus important pour les requêtes faciles que pour les requêtes difficiles en 2002. Par exemple, 17,4% des systèmes fusionnés obtiennent un taux de chevauchement compris entre 0,8 et 0,9 pour les requêtes faciles en 2002 (et 2,6% pour les requêtes difficiles), et 49,2% des systèmes obtiennent un taux de chevauchement de 1 (17,4% pour les requêtes difficiles). En 2003, 24,4% des sytèmes fusionnés obtiennent un taux de chevauchement de 1 pour les requêtes difficiles, et 10,2% des systèmes obtiennent un taux de chevauchement de 1 pour les requêtes faciles.

S'il est vrai que le taux de chevauchement semble être lié à la collection utilisée, le taux de chevauchement peut être vu comme un indicateur sur la difficulté de la requête. Cela permet en situation d'exploitation réelle d'avoir un moyen pour catégoriser les requêtes.

Les conclusions que nous tirons à l'issue de nos expérimentations nécessitent cependant d'être validées sur plusieurs années. En effet, nous avons utilisé 2 années dans ces expérimentations, et les résultats auxquels nous aboutissons constituent une piste intéressante qui devra être complétée. Il en est de même pour les expérimentations portant sur la typologie des requêtes, ainsi que sur le taux de chevauchement. Notre échantillon de requêtes est constitué de 12 requêtes et pourra être étendu afin de pouvoir généraliser les résultats. Nous songeons notamment à utiliser un sous-ensemble de SRI ayant participé à plusieurs années de TREC comme perspective à court terme de nos travaux, afin de compléter nos analyses et valider nos hypothèses.

Le troisième volet de notre contribution consiste à établir un rapprochement entre la classification linguistique des requêtes et les techniques de fusion abordés dans la deuxième partie de notre contribution. L'intuition que nous avions était qu'en établissant le couplage classification linguistique/ fusion, nous serions capables de proposer des stratégies de recherche à même de s'adapter en fonction du contexte linguistique de la requête. Nous avons proposé dans cette partie deux méthodes de fusion probabilistes (MaxProb et MaxProbSeg) permettant de mettre en place une phase d'apprentissage pendant laquelle la meilleure stratégie à utiliser en fonction des CL des requêtes est déterminée. Cette stratégie est ensuite appliquée sur les données lors de la phase de test. Nous avons utilisé dans nos expérimentations les collections de données issues de TREC . Les mesures que nous avons utilisées sont la MAP, la R-précision, et les mesures de haute précision (P@5, P@10, et P@15).

L'algorithme de base (ProbFuse) présenté dans [LTP+06] combine les documents en fonction de leur position dans les listes de documents restitués par les différents SRI . L'algorithme ProbFuse utilise la notion de segment pour déterminer la probabilité de pertinence d'un système. Le découpage d'une liste de documents restitués par un système donne un ensemble de segments. Un segment contient un nombre fixé de

documents. Dans nos expérimentations, ce nombre est fixé à 25 comme le préconise [LTP+06]. Les deux algorithmes que nous proposons sont des adaptations de l'algorithme ProbFuse.

La première adaptation que nous proposons (MaxProb) diffère de ProbFuse essentiellement lors de la phase de test. ProbFuse attribue à tous les documents un score à partir d'une combinaison des scores attribués par les autres systèmes. MaxProb quant à lui se base sur les performances passées des systèmes pour sélectionner par segment le système qui sera utilisé. La liste finale est alors constituée de documents issus de différents segments et de différents systèmes. MaxProbSeg réalise la fusion en choisissant les N meilleurs systèmes ayant obtenu la meilleure probabilité de pertinence. Nos résultats montrent que MaxProbSeg permet d'obtenir de meilleurs résultats que ProbFuse et MaxProb, indépendamment de la mesure choisie, lorsque le nombre de systèmes utilisés est égal à 10 (nous appelons cette version de MaxProbSeg best_all(10)). Nous obtenons par exemple une amélioration de la MAP de 13,67% de best_all(10) par rapport aux résultats de ProbFuse, et de 17,6% par rapport à MaxProb. Les amélioration de la P@15 (best_all(10)) sont de l'ordre de 12,28% par rapport à ProbFuse.

Grâce à des tests de significativité, nous vérifions que les différences de performance obtenues sont statistiquement significatives. Nous avons utilisé le T-test et les résultats nous indiquent qu'il n'existe pas de différence significative entre les résultats des algorithmes MaxProb, MaxProbSeg, et ProbFuse lorsque la P@5 est considérée. Par contre, pour la MAP et la R-précision, la différence est significative entre les résultats de best_all(10) et les deux algorithmes ProbFuse et MaxProb. On obtient une valeur de P égale à 0,012900 pour le test best_all(10)/ProbFuse, et une valeur de P égale à 0,000183 pour le test best_all(10)/MaxProb lorsque la MAP est considérée. Dans le cas de la P@15, les résultats du T-test indiquent une valeur de P égale à 0,031260 pour le test best_all(10)/ProbFuse, et 0,001858 pour le test best_all(10)/MaxProb.

Les résultats que nous avons obtenus sont encourageants et montrent que la prise en compte des spécificités locales des requêtes (CL, niveau de difficulté, performances des systèmes, taux de chevauchement, etc), des systèmes, et des documents est un moyen d'amélioration des résultats à travers la mise en place de processus s'adaptant à ces différents contextes. Les différentes méthodes que nous avons proposées orientent nos travaux vers la mise en place d'un processus de recherche d'information adaptatif. Les travaux que nous avons menés dans cette thèse (notamment ceux présentés dans le chapitre 4, [KMDB07b], [KMDB07a]) nous ont permis en outre d'établir une collaboration ((depuis 2006)) avec des collègues mathématiciens à travers la mise en place d'un projet nommé ACRIC (Analyse Canonique AC et Recherche d'Information Contextuelle). Ce projet offre un formidable cadre de rapprochement entre des informaticiens et des mathématiciens autour de problématiques communes. En l'occurrence, les collègues avec lesquels nous collaborons travaillent sur l'analyse canonique (cf. chapitre 7), et son application en RI ouvre une nouvelle orientation de recherche que nous souhaitons prendre.

Plusieurs perspectives sont envisageables à nos travaux, et certaines s'inscrivent dans le cadre du projet ACRIC. Nous présentons les cinq perspectives que nous envisageons mettre en œuvre à court et moyen termes.

- Une première perspective que nous souhaitons aborder concerne les CL des requêtes. Cet aspect est un aspect important de nos travaux et nous souhaitons mener une analyse plus approfondie des CL afin de proposer les "meilleures CL" à utiliser. Nous avons utilisé dans nos travaux 13 CL définies dans [MT05], et nous souhaitons faire ressortir quelques traits linguistiques significatifs ainsi qu'une méthode d'extraction plus fine. Nous comptons à cet effet établir une collaboration avec des collègues linguistes de l'université du Mirail à Toulouse. De plus, nous avons choisi pour nos expérimentations de regrouper nos requêtes en 3 classes. Le choix du nombre optimal de classes à retenir à l'issue de la classification a sans doute un impact sur les résultats qui seront obtenus. Toutefois, une étude complémentaire devrait permettre de déterminer le nombre de classes à utiliser, et de mesurer l'impact du nombre de classes sur les résultats.

- Une autre perspective porte sur le choix des indicateurs de performance des systèmes à utiliser dans le cadre d'un processus adaptatif. En effet, l'évaluation des SRI est au coeur des problématiques de la RI . Nous souhaitons d'une part analyser les corrélations existant entre les 27 mesures proposées par TREC afin de retenir parmi elles les meilleures à utiliser. Cette perspective entre dans le cadre du projet ACRIC.

- Une troisième perspective envisageable relève directement de l'application de l'analyse canonique (AC) en RI . Dans le cadre du projet ACRIC, nous souhaitons utiliser le potentiel de l'AC afin d'explorer les relations qui existent entre les CL des requêtes et les performances des systèmes. Le groupe IMT avec lequel nous collaborons dans le cadre du projet ACRIC travaille sur l'approfondissement de l'AC dans le cadre de leur mise en œuvre sur des données issues de la génomique. En particulier, dans le cadre de sa thèse, Ignacio González a développé une extension régularisée de l'AC pour pouvoir traiter des données comportant un nombre important de variables [GDMB07, GDM$^+$07].

- La quatrième perspective à nos travaux concerne l'application de l'AC à deux groupes de variables. Cette technique est directement transposable en RI . Dans le cadre de la RI adaptative, nous pensons que l'étude des performances locales des systèmes peut permettre de proposer de meilleures stratégies de fusion à utiliser. Nous pensons notamment que les caractéristiques des systèmes (modèles théoriques utilisés par exemple) constituent un niveau supplémentaire, détail qu'il est bon de prendre en compte. Nous souhaitons donc appliquer l'AC à plus de deux groupes variables. Cela nous permettra de contribuer au développement d'une AC généralisée. Cette perspective s'inscrit dans le cadre du projet ACRIC, et la nature complexe des données que nous souhaitons manipuler (un ensemble de requêtes, des CL associées à chaque requête, pour chaque requête un ensemble d'indica-

teur de performance des systèmes, des caractéristiques de chaque système utilisé, etc) constitue une nouvelle problématique qui intéresse fortement nos collègues mathématiciens.

– Pour finir, nous souhaitons mettre en place un prototype de RI adaptative, prenant en compte les profils linguistiques des requêtes pour la classification, et les techniques de fusion que nous avons proposées. Ce prototype devrait aussi intégrer les nouvelles orientations que nous avons abordées en perspective.

Bibliographie

[ABC+96] J. Allan, L. Ballesteros, J. P. Callan, W. B. Croft, and Z. Lu. Arecent experiments with inquery. *In Harman (1996). NIST Special Publication 500-236*, 1996.

[ACBJ02] A. Abdur Chowdhury, S. Beitzel, and E. Jensen. Analysis of combining multiple query representations with varying lengths in a single engine. *Proceedings of the International Conference on Information Technology : Coding and Computing*, page 236, 2002.

[ACR04] G. Amati, C. Carpineto, and G. Romano. Query difficulty, robustness and selective application of query expansion. *In Proceedings of the 25th European Conference on Information Retrieval (ECIR 2004).*, pages 127–137, 2004.

[AG92] J. Aitchison and A. Gilchrist. Construire un thésaurus. *ADBS.*, 1992.

[AGBS00] N. Aussenac-Gilles, B. Biébow, and N. Szulman. Revisiting ontology design : a method based on corpus analysis. *Knowledge engineering and knowledge management : methods, models and tools, Proc. of the 12th International Conference on Knowledge Engineering and Knowledge Management. Juan-Les-Pins (F). Oct 2000. R Dieng and O. Corby (Eds). Lecture Notes in Artificial Intelligence Vol 1937. Berlin : Springer Verlag.*, pages 172–188, 2000.

[AGCS06a] N. Aussenac-Gilles, A. Condamines, and F. Sèdes. Evolution et maintenance des ressources termino-ontologiques : une question à approfondir. *Information - Interaction - Intelligence, Cépaduès Editions, Numéro spécial Ressources termino-ontologiques, Vol. Hors-série*, pages 7–14, 2006.

[AGCS06b] N. Aussenac-Gilles, A. Condamines, and F. Sèdes. Ressources termino-ontologiques. *Information - Interaction - Intelligence, Cépaduès Editions, Vol. Hors-série*, 2006.

[AGM04] N. Aussenac-Gilles and J. Mothe. Ontologies as background knowledge to explore document collections. *In Actes de la Conférence sur la Recherche d'Information Assistée par Ordinateur (RIAO).*, pages 129–142, 2004.

[ALN03] C. Alvarez, P. Langlais, and J. Y. Nie. Word pairs in language modeling for information retrieval. *Rapport interne, RALI.*, 2003.

[AM01] A.J Aslam and M. Montague. Models for metasearch . *24th annual international ACM SIGIR conference on Research and development in information retrieval. ACM press*, pages 276–284, 2001.

[APC01] J. Anderson and J. Pérez-Carballo. The nature of indexing : How humans and machines analyze messages and texts for retrieval : Part ii : Machine indexing, and the allocation of human versus machine effort. *Information Processing and Management, (37)*, pages 255–277, 2001.

[AvdWKvB00] A. Arampatzis, T. van der Weide, C.H. Koster, and P. van Bommel. Linguistically motivated information retrieval. *In A. Kent, editor, Encyclopedia of Library and Information Science.*, pages 201–222, 2000.

[Bat86] M. Bates. Subject access in online catalogs : A design model. *Journal of the American Society for Information Science, 11.*, pages 357–376, 1986.

[Bau97] C. Baumgarten. A probabilistic model for information retrieval. *Belkin et al., SIGIR97*, pages 258–266, 1997.

[Baz05] M. Baziz. Indexation conceptuelle guidée par ontologie pour la recherche d'information. *Thèse de doctorat, Université Paul Sabatier, Toulouse, France.*, 2005.

[BB05] A. Baccini and P. Besses. Data mining i exploration statistique. *Publication du Laboratoire de Statistique et Probabilité de Toulouse*, 2005.

[BC92] J. Belkin and W. Croft. Information filtering and information retrieval : two sides of the same coin ? *Communications of the ACM, 35(12)*, 1992.

[BCCC93] N.J. Belkin, C. Cool, W.B. Croft, and J.P. Callan. The effect of multiple query representations on information retrieval system performance. *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 339–346, 1993.

[BEB99] P. Bellot and M. El-Beze. Query length, number of classes and routes through clusters : Experiments with a clustering method for information retrieval. *In Proceedings of IEEE ICSC'99, Hong-Kong. In Lecture Notes in Computer Science (LNCS 1749 Internet Applications), L. Chi-Wong Hui and D. L. Lee (Ed.). Springer-Verlag*, pages 196–205, 1999.

[Bel00] P. Bellot. Méthodes de classification et de segmentation locales non supervisées pour la recherche documentaire. *Thèse de doctorat à l'Université d'Avignon et des Pays de Vaucluse.*, 2000.

[BH04] C. Buckley and D. Harman. Reliable information access. *Final report, 27th International ACM SIGIR Conference on Research and Development in Information Retrieval. Sheffield : ACM Press*, pages 528–529, 2004.

[BJC+03] S.M. Beitzel, E.C. Jensen, A. Chowdhury, O. Frieder, D. Grossman, and N. Goharian. Disproving the fusion hypothesis : An analysis of data

fusion via effective information retrieval strategies. *In ACM SAC'03*, pages 823–827, 2003.

[BJC+04] S.M. Beitzel, E.C. Jensen, A. Chowdhury, D. Grossman, O. Frieder, and Goharian N. Fusion of effective retrieval strategies in the same information retrieval system. *J. Am Soc. Inf. Sci. Technol., 55(10)*, pages 859–868, 2004.

[Bla90] D.C. Blair. Language and representation in information retrieval. *Elsevier science publishers, second edition*, 1990.

[BOB82] N. J. Belkin, R. Oddy, and H. Brooks. Ask for information retrieval : Part i background and theory. *Journal of Documentation, 38(2)*, pages 61–71, 1982.

[Buc85] C. Buckley. Implementation of the smart information retrieval system. *Technical Report, Ithaca, New York : Computer Science Department, Cornell University.*, pages 85–686, 1985.

[Buc04] C. Buckley. Why current ir engines fail. *In Proceedings of the 27th annual International ACM SIGIR conference on Research and development in information retrieval. ACM Press*, pages 584–585, 2004.

[BW00] C. Buckley and J. Waltz. Smart in trec 8. *In The Eighth Text Retrieval Conference (TREC-8). Ed. E.M. Voorhees and D.K. Harman. Gaithersburg, MD : NIST*, 2000.

[BYH07] R. Baeza-Yate and C. Hurtado. Improving search engines by query clustering. *Journal Of The American Society for Information Science and Technology, 58(12)*, pages 1793–1804, 2007.

[BYRN99] R. Baeza-Yates and B. Ribeiro-Neto. Modern information retrieval. *ACM (Association for Computing Machinery)*, 1999.

[CDHK01] J. Carlberger, H. Dalianis, M. Hassel, and 0. Knutsson. Improving precision in information retrieval for swedish using stemming. *In Proceedings of the 13th Nordic Conference on Computational Linguistics (NODALIDA).*, 2001.

[CL92] H. Chen and K.J. Lynch. Automatic construction of networks of concepts characterizing documents databases. *IEEE Expert Intelligent Systems and their Applications.*, pages 887–902, 1992.

[Cla03] V. Claveau. Acquisition automatique de lexiques sémantiques pour la recherche d'information. *Thèse de doctorat, Université de Rennes 1, France*, 2003.

[CLC95] J.P. Callan, Z. Lu, and W.B. Croft. Searching distributed collections with inference networks. *In Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval.*, pages 21–28, 1995.

[Cle67] C.W. Cleverdon. The cranfield tests on index language devices. *Aslib Proceedings 19(6)*, pages 173–193, 1967.

[Con80] W.J. Conover. Practical nonparametric statistics. *2nd edition, John Wiley and Sons, New York.*, 1980.

[Cro80] W. B. Croft. A model of cluster searching based on classification. *Information Systems, Vol. 5*, pages 189–195, 1980.

[Cro00] W.B. Croft. Combining approaches to information retrieval. *In W. B. Croft, editor, Advances in Information Retrieval. Kluwer Academic Publishers.*, pages 1–36, 2000.

[CS04] V. Claveau and P. Sébillot. Extension de requêtes par lien sémantique nom-verbe acquis sur corpus. *In proceedings TALN'04*, 2004.

[CTZC02] S. Cronen-Townsend, Y. Zhou, and W.B. Croft. Predicting query performance. *Proceedings of the 25th annual international ACM-SIGIR conference on research and development in information retrieval, Tampere*, pages 299–306, 2002.

[CYDP06] D. Carmel, E. YomTov, A. Darlow, and D. Pelleg. What makes a query difficult ?. *SIGIR'06.*, pages 390–397, 2006.

[Dai96] B. Daille. Study and implementation of combined techniques for automatic extraction of terminology. *The balancing act combining symbolic and statistical approaches to language, MIT Press, Cambridge, Massachusetts*, pages 49–66, 1996.

[Dai02] B. Daille. Découvertes linguistiques en corpus. *Mémoire d'habilitation à diriger des recherches. Université de Nantes, France*, 2002.

[Dam95] M. Damashek. Gauging similarity with n-grams : Language-independent categorization of text. *Science, Volume 267*, pages 843–848, 1995.

[DDL+90] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *In Journal of the American Society of Information Science, Vol. 41 :6*, pages 391–407, 1990.

[DFS02] B. Daille, C. Fabre, and P. Sébillot. Applications of computational morphology. *In P. Boucher, editor, Many Morphologies. Cascadilla Press.*, pages 210–234, 2002.

[DHMO04] T. Dkaki, G. Hubert, J. Mothe, and E. Orain. Recherche de la nouveauté dans les textes : une tâche difficile. *VSST, Veille Stratégique Scientifique et Technologique*, pages 355–368, 2004.

[dLBEBM98] C. de Loupy, P. Bellot, M. El-Bèze, and P.F. Marteau. Query expansion and classification of retrieved documents. *Actes de Seventh Text Retrieval Conference (TREC-7).*, pages 443–450, 1998.

[Fag87] J. Fagan. Experiments in automatic phrase indexing for document retrieval : A comparison of syntactic and non-syntactic methods. *Thèse de doctorat, Université de Cornell, New-York, États-Unis.*, 1987.

[FB01] C. Fabre and D. Bourigeault. Linguistic clues for corpus-based acquisition of lexical dependencies. *in Proceeding of Corpus Linguistics, Lancaster*, 2001.

[FDD+88] G.W. Furnas, S. Deerwester, S.T. Dumais, Harshman R.A. and Streeter L.A. Landauer, T.K. and, and K.E. Lochbaum. Information retrieval using a singular value decomposition model of latent semantic structure. *Proceedings of the 11th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 465–480, 1988.

[Fox92] C. Fox. Lexical analysis and stoplists. *Information Retrieval : Data Structure and Algorithms*, pages 102–130, 1992.

[FS94] E.A. Fox and J.A. Shaw. Combination of multiple searches. *Proceedings of the 2nd Text Retrieval Conference (TREC-2), NIST special publication*, pages 243–252, 1994.

[Fur92] N. Furh. Probabilistic models in information retrieval. *The Computer Journal,35(3)*, pages 233–245, 1992.

[FV07] M. Farah and D. Vanderpooten. An outranking approach for rank aggregation in information retrieval. *SIGIR 2007 proceeedins.*, pages 591–598, 2007.

[Gau06] E. Gaussier. Quelle langue pour la ri ? *EARIA.*, 2006.

[GDM+07] I. Gonzalez, S. Déjean, P. Martin, A. Baccini, O. Gonçalves, and P. Besse. Integrating gene expression data through regularized canonical correlation analysis. *Biostatistics*, 2007.

[GDMB07] I. Gonzalez, S. Déjean, P. Martin, and A. Baccini. Cca : an r package to extend canonical correlation analysis. *Journal of Statistical Software*, 2007.

[GGHR00] E. Gaussier, G. Grefenstette, D. Hull, and R. Roux. Recherche d'information en français et traitement automatique des langues. *Traitement automatique des langues, vol 41*, pages 473–493, 2000.

[GGS97] E. Gaussier, G. Grefenstette, and M. B. Schulze. Traitement du langage naturel et recherche d'informations : quelques expériences sur le français. *In Actes des premières journées scientifiques et techniques du réseau francophone de l'ingénierie de la langue de l'AUPELF-UREF.*, pages 9–14, 1997.

[GMV99] N. Guarino, C. Masolo, and G. Vetere. Ontoseek : Using large linguistic ontologies for accessing on-line yellow pages and product catalogs. *National Research Council, LADSEBCNR : Padavo, Italy.*, 1999.

[Gre92] G. Grefenstette. Use of syntactic context to produce term association lists. *In Proceedings of the 15th Annual International ACM SIGIR Conference on Research and development in Information Retrieval.*, pages 89–97, 1992.

[GT94] G. Grefenstette and P. Tapanainen. What is a word, what is a sentence ? problems of tokenization. *in The 3rd International Conference on Computational Lexicography, Budapest, (1994).*, pages 79–87, 1994.

[Gua97] N. Guarino. Semantic matching : Formal ontological distinctions for information organization, extraction, and integration. *SCIE*, pages 139–170, 1997.

[GWR99] S. Gauch, J. Wang, and S.M. Rachakonda. A corpus analysis approach for automatic query expansion and its extension to multiple databases. *ACM Transactions on Information Systems*, pages 250–269, 1999.

[Had02] H. Haddad. Extraction et impact des connaissances sur les performances des sustèmes de recherche d'information. *PhD thesis, Université Joseph Fourier, Grenoble, France*, 2002.

[Har91] D. Harman. How effective is suffixing ? *Journal of the American Society for Information Science.*, pages 7–15, 1991.

[HCHM06] N. Hernandez, C. Chrisment, C. Hubert, and J. Mothe. Mise à jour d'une ontologie de domaine à partir de l'analyse de nouveaux documents du domaine pour l'indexation de documents. *Information - Interaction - Intelligence, Cépaduès Editions, Numéro spécial Textes et ressources terminologiques et/ou ontologiques : évolution et maintenance, Vol. Hors-série*, pages 53–83, 2006.

[HCM06] N. Hernadez, G. Chrisment, C. Hubert, and J. Mothe. Mise à jour d'une ontologie de domaine à partir de l'analyse de nouveaux documents du domaine pour l'indexation de documents. *Information - Interaction - Intelligence, Cépaduès Editions, Numéro spécial Textes et ressources terminologiques et/ou ontologiques : évolution et maintenance, Vol. Hors-série*, pages 53–83, 2006.

[HEH+95] W.R. Hersch, D.L. Elliot, D.H. Hickam, S.L. Wolf, A. Molnar, and C. Lechtenstien. Towards new measures of information retrieval evaluation. *Proceedings of the 18th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 164–170, 1995.

[HM04] N. Hernandez and J. Mothe. Ontologies pour l'aide à l'exploration d'une collection de documents. *Dans : VSST, Veille Stratégique Scientifique et Technologique, Toulouse, 23/10/04-25/10/04, IRIT*, pages 405–416, 2004.

[HMCE07] N. Hernadez, J. Mothe, C. Chrisment, and D. Egret. Modeling context through domain ontologies. *Journal of Information Retrieval, Springer, Numéro spécial Contextual Information Retrieval Systems, Vol. 10 N. 2*, pages 143–172, 2007.

[HMK07] G. Hubert, J. Mothe, and D. Kompaore. Réinjection de pertinence pour la fusion de systèmes. *Colloque Veille Stratégique Scientifique et Technologique (VSST 2007)*, 2007.

[HO03a] B. He and I. Ounis. A query-based model selection approach for the poorly-performing queries. *In Proceedings of TREC 2003.*, pages 636–645, 2003.

[HO03b] B. HE and I. Ounis. A study of parameter tuning for term frequency normalization. *In Proceedings of the Twelveth ACM CIKM International Conference on Information and Knowledge Management*, pages 10–16, 2003.

[HO04a] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. *In A. Apostolico and M. Melucci, editors, String Processing and Information Retrieval, 11th International Conference, SPIRE 2004, volume 3246 of Lecture Notes in Computer Science,*, 2004.

[HO04b] B. He and I. Ounis. A query-based pre-retrieval model selection approach to information retrieval. *In Proceedings of RIAO 2004.*, 2004.

[Hot36] H. Hotelling. Relations between two sets of variates. *Biometrika*, 1936.

[HP96] M.A. Hearst and J.O. Pedersen. Re-examining the cluster hypothesis : Scatter/gather on retrieval results. *In SIGIR 1996.*, pages 76–84, 1996.

[Hud94] M. Hudon. Le thésaurus : construction, élaboration, gestion. *ASTED.*, 1994.

[Hul96] D. Hull. Stemming algorithms. a case study for detailed evaluation. *Journal of the American Society for Information Science, 47(1).*, 1996.

[JG01] S. Jones and W. Gordon. Paynter : Human evaluation of kea, an automatic keyphrasing system. *JCDL.*, pages 148–156, 2001.

[JM00] D. Jurafsky and J. H. Martin. Speech and language processing. *Prentice-Hall.*, 2000.

[JvR71] N. Jardine and C.J. van Rijsbergen. The use of hierarchical clustering in information retrieval. *Information Storage and Retrieval, 7.*, pages 217–240, 1971.

[JZ00] C. Jacquemin and P. Zweigenbaum. Traitement automatique des langues pour l'accès au contenu des documents. *Jacques Le Maître, Jean Charlet et Catherine Garbay, éditeurs, Le document en sciences du traitement de l'information*, pages 71–109, 2000.

[Kar99] J.. Karlgren. Stylistic experiments in information retrieval. *In Natural Language Information Retrieval, Kluwer.*, 1999.

[Kel88] J.S. Kelly. Social choice theory : An introduction. *Springer-Verlag.*, 1988.

[KM07] D. Kompaore and J. Mothe. Query clustering and ir system detection. experiments on trec data. *ACM International Workshop for Ph.D. Students in Information and Knowledge Management (ACM PIKM 2007)*, 2007.

[KMDB07a] D. Kompaore, J. Mothe, S. Dejean, and A. Baccini. Prédiction du sri à utiliser en fonction des critères linguistiques de la requête. *Conférence francophone en Recherche d'Information et Applications (CORIA 2007)*, pages 239–254, 2007.

[KMDB07b] D. Kompaore, J. Mothe, S. Dejean, and A. Baccini. Probabilistic fusion and categorization of queries based on linguistic features. impact on high precision measures. *Large-Scale Semantic Access to Content (Text, Image, Video and Sound) (RIAO 2007)*, 2007.

[KML06] D. Kompaore, J. Mothe, and E. Lemoing. Fusion de systèmes pour la recherche de passages dans les textes. *Conférence francophone en Recherche d'Information et Applications (CORIA 2006)*, pages 295–300, 2006.

[Kro93] R. Krovetz. Viewing morphology as an inference process. *In Korfhage et al.*, pages 191–203, 1993.

[KSYCKS01] L. Kyung-Soon, P. Young-Chan, and C. Key-Sun. Re-ranking model based on document clusters. *Information Processing and Management 37.*, pages 1–14, 2001.

[Lal96] M. Lalmas. Theories of information and uncertainty for the modelling of information retrieval : an application of situation theory and dempster-shafer's theory of evidence. *PhD thesis, Department of Computing Science, University of Glasgow, Scotland*, 1996.

[Lee95] J.H. Lee. Combining multiple evidence from different properties of weighting schemes. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 180–188, 1995.

[Lee97] J. Lee. Analysis of multiple evidence combination. *22th International ACM SIGIR Conference on Research and Development in Information Retrieval. New York : ACM Press*, pages 267–276, 1997.

[Lef00] P. Lefevre. La recherche d'information. du texte intégral au thésaurus. *Hermes science*, 2000.

[Lid98] E.D. Liddy. Enhanced text retrieval using natural language processing. *Bulletin of the American Society for Information Science, Vol. 24, No. 4*, 1998.

[LLYM04] S. Liu, F. Liu, C. Yu, and W. Meng. An effective approach to document retrieval via utilizing wordnet and recognizing phrases. *In Proceedings of the 27th Annual international Conference on Research and Development in information Retrieval, SIGIR'04, ACM Press, New York, NY*, pages 266–272, 2004.

[LMP06] L. Lebart, A. Morineau, and M. Piron. Statistique exploratoire multidimensionnelle : Visualisations et inférences en fouille de données. *4e Edition, Dunod, Paris.*, 2006.

[Lov68] J.B. Lovins. Development of a stemming algorithm. *Mechanical Translation and Computationnal Linguistics 11(1)*, pages 22–37, 1968.

[LTP+05] D. Lillis, F. Toolan, L. Peng, A. Mur, R. Collier, and J. Dunnion. Probability-based fusion of information retrieval result sets. *In Proceedings of the 16th Irish Conference on Artificial Intelligence and Cognitive Science (AICS 2005)*, pages 147–156, 2005.

[LTP+06] D. Lillis, F. Toolan, L. Peng, R. Collier, and J. Dunnion. Probability-based fusion of information retrieval result sets. *in Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development on Information Retrieval, Seattle 2006.*, pages 139–146, 2006.

[MA01] M. Montague and J.A. Aslam. Models for metasearch. *SIGIR 2001.*, pages 9–12, 2001.

[Mac67] J.B. MacQueen. Some methods for classification and analysis of multivariate observations. *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press*, pages 281–297, 1967.

[MBSC97] M. Mitra, C. Buckley, A. Singhal, and C. Cardie. An analysis of statistical and syntactic phrases. *In Devroye, L. and Chrisment, C., editors, Proceedings of RIAO'97*, pages 200–214, 1997.

[MdG91] J. Maniez and E. de Grolier. A decade of research in classification. 1991.

[Mih04] R. Mihalcea. Co-training and selftraining for word sense disambiguation. *In Proceedings of the Conference on Natural Language Learning (CoNLL 2004).*, 2004.

[MK60] M. Maron and J. Kuhns. On relevance, probabilistic indexing and information retrieval. *Journal of the Association for Computing Machinery,(7)*, pages 216–244, 1960.

[MKB89] K.V. Mardia, J.T. Kent, and J.M. Bibby. Multivariate analysis. *Academic Press.7th Printing.*, 1989.

[MM00] D. Moldovan and R. Mihalcea. Using wordnet and lexical operators to improve internet searches. *In IEEE Internet Computing.*, pages 34–43, 2000.

[Mor06] F. Moreau. Revisiter le couplage traitement automatique des langues et recherche d'information. *Thèse de doctorat, université de Rennes 1, France*, 2006.

[Mou88] H. Moulin. Axioms of cooperative decision making. *Cambridge University Press.*, 1988.

[MT05] J. Mothe and L. Tanguy. Linguistic features to predict query difficulty-a case study on previous trec campaigns. *SIGIR workshop on Predicting Query Difficulty - Methods and Applications.*, 2005.

[MT06] J. Mothe and L. Tanguy. Rapport de fin de projet, projet ariel. *Rapport de contrat, Rapport final ARIEL, TCAN*, 2006.

[MT07] J. Mothe and L. Tanguy. Unités d'indexation et taille des requêtes pour la recherche d'information en français. *Congrès Informatique des Organisations et Systèmes d'Information et de Décision (INFORSID 2007)*, pages 225–241, 2007.

[MWH02] T. Mandl and C. Womser-Hacker. Linguistic and statistical analysis for the clef topics. *Spinger Verlag*, pages 505–511, 2002.

[Nam00] F. Namer. Flemm : un analyseur dérivationel du français à base de règles. *raitement automatique des langues, 41(2)*, pages 523–547, 2000.

[OMK91] Y. Ogawa, T. Morita, and K. Kobayashi. A fuzzy document retrieval system using the keyword connection matrix and a learning method. *Fuzzy sets and systems.*, pages 163–179, 1991.

[OP97] I. Ounis and M. Pasca. An extended inverted file approach for information retrieval. *In International Database Engineering and Application Symposium (IDEAS'97). IEEE Computer Society Press*, pages 397–402, 1997.

[P.96] Harter S. P. Variations in relevance assessments and the measurement on retrieval experiments. *Journal of the American Society for Information Science, 29 (4)*, pages 411–414, 1996.

[PB97] T. Pedersen and A. Bruce. A new supervised learning algorithm for word sense disambiguation. *In Proceedings of the 14th National Conference on Artificial Intelligence, AAAI, Providence, États-Unis.*, 1997.

[Pea88] J. Pearl. Probabilistic reasoning in intelligent systems : Networks of plausible inference. *Morgan Kaufmann Publishers*, 1988.

[Pet00] C. Peters. Cross-language information retrieval and evaluation. *Workshop of Cross-Language Evaluation Forum, CLEF 2000, Lisbon, Portugal*, pages 21–22, 2000.

[Pic05] A. Picton. Utilisation raisonnée de connaissances sémantiques pour la recherche d'information. le cas de l'expansion de requêtes par voisins distributionnels. *Rapport DEA, Toulouse*, 2005.

[PKJ99] A. Pirkola, H. Keskustalo, and K. Jarvelin. The effects of conjunction, facet structure and dictionary combinations in concept-based cross-language retrieval. *Information Retrieval, 1(3)*, pages 217–250, 1999.

[Por80] M.F. Porter. An algorithm for suffix stripping. pages 130–137, 1980.

[QF95] Y. Qiu and H.P. Frei. Improving the retrieval effectiveness by a similarity thesaurus. *Rapport interne 225, Departement of Computer science, ETH Zürich.*, 1995.

[RFN99] F. Ren, L. Fan, and J. Nie. Aak approach : How to acquire knowledge in an actual application system. *IASTED International Conference on Artificial Intelligence and Soft Computing, Honolulu*, pages 136–140, 1999.

[Rik82] W.H. Riker. Liberalism against populism. *Waveland Press.*, 1982.

[Ril95] E. Riloff. Little words can make a big difference for text classification. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 130–136, 1995.

[RNM96] B. A. Ribeiro-Neto and R. Muntz. A belief network model for ir. *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 253–260, 1996.

[RSJ76] S. E. Robertson and K. Spark-Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science.*, pages 129–146, 1976.

[RW94] S.E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. *In Proceedings of SIGIR 1994.*, pages 232–241, 1994.

[RWHB+95] S.E. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau. Okapi at trec-3. *In D. K. Harman, editor, The Third Text REtrieval Conference (TREC-3) NIST.*, pages 129–146, 1995.

[Sal68] G. Salton. Automatic information organisation and retrieval. *McGraw-Hill, New York. opz.*, 1968.

[Sal71a] G. Salton. A comparison between manual and automatic indexing methods. *Journal of the American Documentation, 20(1).*, page 6171, 1971.

[Sal71b] G. Salton. The smart retrieval system - experiments in automatic document processing. *Prentice Hall Inc. Englewood Cliffs, NJ*, 1971.

[Sar95] T. Saracevic. Evaluation of evaluation in information retrieval. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 137–146, 1995.

[SBH97] W. M. Shaw, R. Burgin, and P. Howell. Performance standards and evaluations in ir test collections : Cluster-based retrieval models. *Information Processing and Management*, pages 1–14, 1997.

[SFW83] G. Salton, E.A Fox, and H. Wu. Extended boolen information retrieval. *communication of the ACM.*, pages 1022–1036, 1983.

[Sha48] C. Shannon. A mathematical theory of communication. *Bell System Technical Journal, (27)*, pages 623–656, 1948.

[SJ72] K. Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation, (28)*, pages 11–21, 1972.

[SJ92] K. Sparck-Jones. Assumptions and issues in text-based retrieval. *Text-Based Intelligent Systems - Current Research and Practice in Information Exctraction and Retrieval. Lawrence Erlbaum Assc.*, 1992.

[SJ95] K. Sparck-Jones. Further reflexion on trec. *Information Processing and Management, vol 36, N.2000*, pages 37–85, 1995.

[SJ97] K. Sparck-Jones. Natural language and the information layer. *2007 Athena Lecturer Award, SIGIR'07*, pages 3–6, 1997.

[SL68] G. Salton and M. E. Lesk. Computer evaluation of indexing and text processing. *Journal of the ACM*, pages 8–36, 1968.

[SLWPC99] T. Strzalkowski, F. Lin, J. Wang, and J. Perez-Carballo. Evaluating natural language processing techniques in information retrieval. *In T. Strzalkowski,editor, Natural Language Information Retrieval. Kluwer Academic Publishers*, pages 113–145, 1999.

[SM83] G. Salton and M.J. McGill. Introduction to modern information retrieval. *McGraw Hill Publishing Company, New York*, 1983.

[SM86] G. Salton and J. McGill. Introduction to modern information retrieval. *McGraw-Hill, Inc., New York, NY.*, 1986.

[SNC01] I. Soboroff, C. Nicholas, and P. Cahan. Ranking retrieval systems without relevance judgements. *In Proceedings of 24th Annual International ACM SIGIR Conference.*, pages 66–73, 2001.

[SSMB96] A. Singhal, G. Salton, M. Mitra, and C. Buckley. Document length normalization. *In Information Processing and Management, 32(5).*, pages 619–633, 1996.

[SZ05] M. Sanderson and J. Zobel. Information retrieval system evaluation : effort, sensitivity, and reliability. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval.*, pages 162–169, 2005.

[TC90] H. Turtle and B. Croft. Inférence networks for document retrieval. *Proceedings of the 13th Annual Int. ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1–24, 1990.

[TC91] H. Turtle and W.B. Croft. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems, Vol. 9, No. 3*, 1991.

[THH00] J. Takenobu, O. Hironori, and T. Hozumi. Effectiveness of complex index terms in information retrieval. *In proceedings of the 6th International Conference Recherche d'Information Assistée par Ordinateur.*, 2000.

[TM06] M. Tran and D. Maurel. Prolexbase. un dictionnaire relationnel multilingue de noms propres. *TAL Vol 47 Ũ nř3*, pages 115–139, 2006.

[Tuk77] J. W. Tukey. Exploratory data analysis. *EDA, Reading, MA, (Addison-Wesley).*, 1977.

[TVGJL95] G. Towell, E.M. Voorhees, N.K. Gupta, and B. Johnson-Laird. Learning collection fusion strategies for information retrieval. *In Armand Prieditis and Stuart Russell, editors, Proceedings of the Twelfth Annual Machine Learning Conference, Lake Tahoe*, 1995.

[TVVR02] A. Tombros, R. Villa, and C.J. Van Rijsbergen. The effectiveness of query-specific hierarchic clustering in information retrieval. *Information Processing and Management, 38.*, pages 559–582, 2002.

[VC98] C.C. Vogt and G.W. Cottrell. Predicting the performance of linearly combined ir systems. *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval.*, pages 190–196, 1998.

[VGJL94] E.M. Voorhees, N.K. Gupta, and B. Johnson-Laird. The collection fusion problem. *3rd Annual Text Retrieval Conférence (TREC-3), NIST*, 1994.

[VGJL95] E.M. Voorhees, N.K. Gupta, and B. Johnson-Laird. Learning collection fusion strategies. *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 172–179, 1995.

[VH00] E.M. Voorhees and D. Harman. Overview of the sixth text retrieval conference (trec-6). *Information Processing and Management*, pages 3–35, 2000.

[Voo85] E. M. Voorhees. The effectivenss and efficiency of agglomerative hierarchic clustering in document retrieval. *Ph.D. Thesis, Technical Report TR 85-705 of the Department of Computing Science, Cornell University.*, 1985.

[Voo94] E. M. Voorhees. Query expansion using lexical-semantic relations. *In Croft and vanRijsbergen*, pages 61–69, 1994.

[VPOAR03] H. B. Vassilis Plachouras, I. Ounis, G. Amati, and C. J. Rijsbergen. University of glasgow at the web track : Dynamic application of hyperlink analysis using the query scope. *In Proceedings of the Twelth Text REtrieval Conference (TREC 2003), Gaithersburg, MD*, 2003.

[VR79] C.J. Van-Rijsbergen. Information retrieval. *Butterworth*, 1979.

[vRC75] C.J. van Rijsbergen and W. B. Croft. Document clustering : An evaluation of some experiments with the cranfield 1400 collection. *Information Processing and Management, 11*, pages 171–182, 1975.

[VS86] G. Van Slype. Les languages d'indexation. *Editions d'organisation.*, 1986.

[WA98] W. A. Woods and J. Ambroziak. Natural language technology in precision content retrieval. *In Proceedings of the International Conference on Natural Language Processing and Industrial Applications, (NLP+IA), Moncton, Canada*, 1998.

[War63] J.H. Ward. Hierarchical grouping to optimize an objective function. *Journal of American Statistical Association, vol. 58, nř 301.*, pages 235–244, 1963.

[WH91] R. Wilkinson and P. Hingston. Using the cosine measure in a neural network for document retrieval. *Proceedings of the ACM SIGIR Conference*

on Research and Development in Information Retrieval, pages 202–210, 1991.

[Wil88] P. Willett. Recent trends in hierarchic document clustering : a critical review. *Information Processing and Management, vol. 24, no 5.*, pages 577–597, 1988.

[WNZ02] J. Wen, J. Y. Nie, and H. Zhang. Query clustering using user logs. *ACM Transactions on Information Systems (TOIS)*, pages 59–81, 2002.

[WZW85] S.K.M. Wong, W. Ziarko, and P.C.N. Wong. Generalized vector space model in information retrieval. *Proceedings of the 8th ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 18–25, 1985.

[XC00] J. Xu and W.B Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems.*, pages 79–112, 2000.

[YR98] R.R. Yager and A. Rybalov. On the fusion of documents from multiple collection information retrieval systems. *Journal of the American Society for Information Science.*, pages 1177–1184, 1998.

[ZGZW01] J. Zhang, J. Gao, M. Zhou, and J. Wang. Improving the effectiveness of information retrieval with clustering and fusion. *Computational Linguistics and Chinese Language Processing.*, pages 109–125, 2001.

[Zip49] G. Zipf. Human behaviour and the principle of least effort. *Addison Wesley*, 1949.

Annexe A

Annexes

Caractéristiques des collections de TREC

Nous présentons les différents éléments de TREC que nous avons utilisés dans nos travaux. Nous abordons notamment les collections de documents, les requêtes et les jugements de pertinence tels qu'ils sont présentés dans TREC. Nous indiquons aussi le fonctionnement du programme trec_eval et nous donnons les mesures associées à ce programme.

Les documents

Les documents de TREC sont fournis sur des CD contenant approximativement 1Go de texte. Les documents sont formatés selon le format SGML pour faciliter leur exploitation.Un traitement de suppression des balises est au préalable effectué. Les documents sont issus d'articles de journaux et sont décomposés en un certain nombre de champs (identifiant, date, titre...).

Exemple de document extrait du Financial Times

<DOC>
<DOCNO>FT911-3</DOCNO>
<PROFILE>AN-BEOA7AAIFT</PROFILE>
<DATE>910514
</DATE>
<HEADLINE>
FT 14 MAY 91 / International Company News : Contigas plans DM900m east German project
</HEADLINE>
<BYLINE>
By DAVID GOODHART
</BYLINE>
<DATELINE>
BONN
</DATELINE>
<TEXT>
CONTIGAS, the German gas group 81 per cent owned by the utility Bayernwerk, said yesterday that it intends to invest DM900m (Dollars 522m) in the next four years to build a new gas distribution system in the east German state of Thuringia...
</TEXT>
</DOC>

Les requêtes

Une requête se compose est appelée topic dans TREC. Le topic décrit le besoin d'information de l'utilisateur et est composé de plusieurs champs. Le champ <top> délimite un topic dont le numéro est spécifié par le champ <num>. Le champ <title> donne le titre du topic. La description du sujet (en2 ou 3 phrases) est délimitée par le champ <desc>. Le champ <narr> précise les documents qui sont attendus en réponse. Des exemples de requêtes de TREC sont donnés ci -dessous.

Les 200 topics utilisés dans les tâches adhoc de TREC3, TREC5, TREC6, et TREC7 sont numérotés de 151 à 400 (respectivement 151-200, 251-300, 301-350, 351-400).

Nous donnons dans le tableau A.2, une description des requêtes utilisées dans TREC6.

Jugements de pertinence

Les jugements de pertinence sont une élément important des collections de test. Pour chaque topic, une liste de documents pertinents est établie. La technique utilisée pour composer la liste des jugements de pertinence est la "pooling méthod". L'efficacité de la méthode dépend donc du nombre de participants à lŠévaluation et surtout du nombre de

méthodes différentes utilisées. DŠautre part (et peut-être principalement), cela donne un aspect statique aux résultats. Un système peut rapporter, dans les premiers documents, des textes pertinents mais qui nŠont été rapportés par aucun autre système. Le schéma de la pooling method est donné ci-dessous

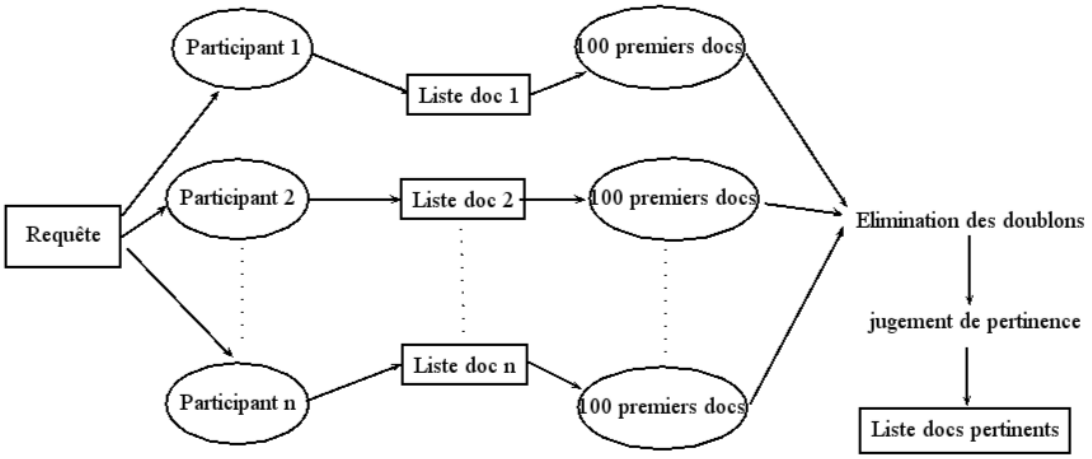

FIG. A.1 – Représentation schématique de la "pooling method"

Exemple d'évaluation de phrases pertinentes pour la tâche Nouveauté de TREC

La liste ci-dessous donne les résultats de l'évaluation du "run" du système ISIALL03 pour la sous tâche détection des passages de TREC Novelty 2003.

```
Evaluation of relevant sentences for run novelty2003_tache1/inputs/input.ISIALL03
      Judgment  System  Number
Topic   count   count   matches  Precision  Recall    F
 N1     184       879    184       0.21      1.00    0.346
 N2      78       500     78       0.16      1.00    0.270
 N3     596       931    595       0.64      1.00    0.779
 N4     438       926    436       0.47      1.00    0.639
 N5     259      1662    259       0.16      1.00    0.27
 N6     317       423    316       0.75      1.00    0.854
 N7      95       306     95       0.31      1.00    0.474
 N8     158       659    158       0.24      1.00    0.387
 N9     160       637    160       0.25      1.00    0.402
 N10   191       257    191       0.74      1.00    0.853
 N11   148       393    148       0.38      1.00    0.547
 N12   729      1043    728       0.70      1.00    0.822
 N13   205       936    205       0.22      1.00    0.359
```

```
N14    93      1129     93       0.08      1.00    0.152
N15   522       649    522       0.80      1.00    0.892
N16   179       500    179       0.36      1.00    0.527
N17   792      1106    792       0.72      1.00    0.835
N18   537      1237    537       0.43      1.00    0.605
N19    62       866     62       0.07      1.00    0.134
N20    69       886     69       0.08      1.00    0.145
N21   340       932    340       0.36      1.00    0.535
N22    84       841     84       0.10      1.00    0.182
N23   346       896    346       0.39      1.00    0.557
N24   160       968    160       0.17      1.00    0.284
N25    19       701     19       0.03      1.00    0.053
N26   661       911    661       0.73      1.00    0.841
N27   730       962    730       0.76      1.00    0.863
N28   371       977    370       0.38      1.00    0.549
N29    65       861     65       0.08      1.00    0.140
N30   445       900    445       0.49      1.00    0.662
N31   985      1219    984       0.81      1.00    0.893
N32   216      1077    216       0.20      1.00    0.334
N33   526       680    526       0.77      1.00    0.872
N34   475      1029    475       0.46      1.00    0.632
N35   221       399    221       0.55      1.00    0.713
N36   167       353    166       0.47      0.99    0.638
N37    76       547     76       0.14      1.00    0.244
N38   140      1126    140       0.12      1.00    0.221
N39   211       590    211       0.36      1.00    0.527
N40   307       533    307       0.58      1.00    0.731
N41   535       672    535       0.80      1.00    0.886
N42   400      1117    398       0.36      0.99    0.525
N43   122       224    122       0.54      1.00    0.705
N44   432       585    432       0.74      1.00    0.850
N45   262      1053    261       0.25      1.00    0.397
N46   322       549    322       0.59      1.00    0.739
N47   420       601    420       0.70      1.00    0.823
N48    98      1075     98       0.09      1.00    0.167
N49   209       684    209       0.31      1.00    0.468
N50   400       810    400       0.49      1.00    0.661
Averages over 50 topics:
Average precision: 0.41
Average recall:    1.00
Average F:         0.540
```

Exemple d'évaluation de inq102

Nous donnons dans la liste suivante les résultats de l'évaluation du run du système inq102 de TREC 3 par le programme trec_eval. L'extrait présenté concerne la requête 151.

```
num_ret         151 1000
num_rel         151 85
num_rel_ret     151 84
map             151 0.6259
R-prec          151 0.6235
bpref           151 0.5978
recip_rank      151 1.0000
ircl_prn.0.00   151 1.0000
ircl_prn.0.10   151 0.9091
ircl_prn.0.20   151 0.7857
ircl_prn.0.30   151 0.7297
ircl_prn.0.40   151 0.6615
ircl_prn.0.50   151 0.6615
ircl_prn.0.60   151 0.6341
ircl_prn.0.70   151 0.5556
ircl_prn.0.80   151 0.4789
ircl_prn.0.90   151 0.3565
ircl_prn.1.00   151 0.0000
P5              151 1.0000
P10             151 0.9000
P15             151 0.8000
P20             151 0.7500
P30             151 0.7333
P100            151 0.5900
P200            151 0.3750
P500            151 0.1640
P1000           151 0.0840
```

Performances initiales des meilleurs systèmes de TREC

Comparaison avec d'autres techniques de fusion

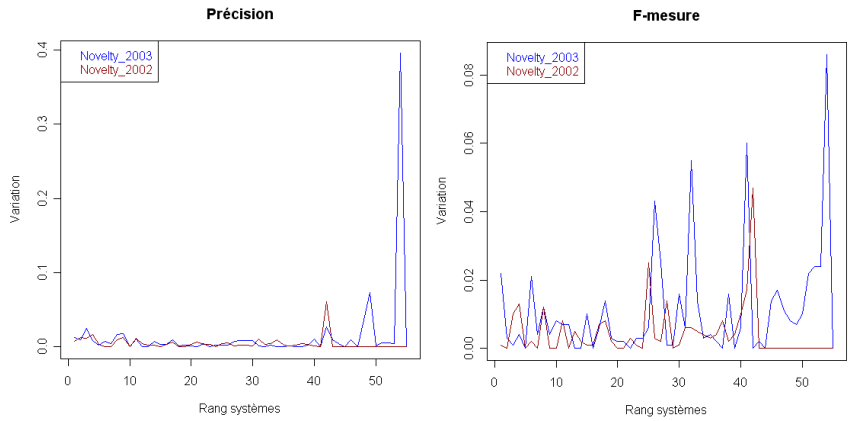

FIG. A.2 – Variation entre la précision et la F-mesure de systèmes de rangs consécutifs

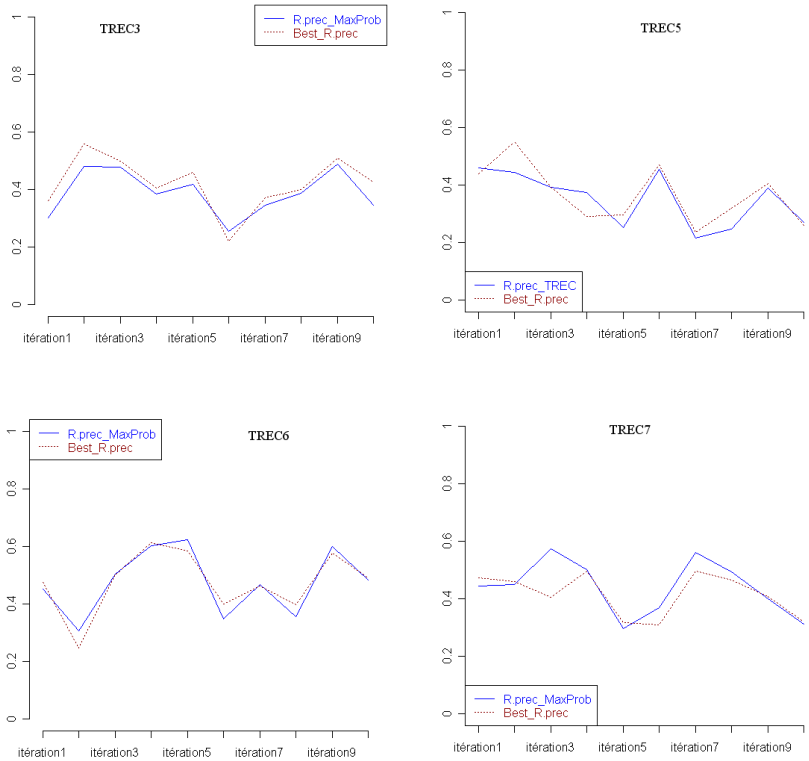

FIG. A.3 – Comparaison de la R-précision

<top><num> Number : 301
<title> International Organized Crime
<desc> Description :
Identify organizations that participate in international criminal activity, the activity, and, if possible, collaborating organizations and the countries involved.
<narr> Narrative :
A relevant document must as a minimum identify the organization and the type of illegal activity (e.g., Columbian cartel exporting cocaine). Vague references to international drug trade without identification of the organization(s) involved would not be relevant.
</top><top>
<num> Number : 302
<title> Poliomyelitis and Post-Polio
<desc> Description :
Is the disease of Poliomyelitis (polio) under control in the world ?
<narr> Narrative :
Relevant documents should contain data or outbreaks of the polio disease (large or small scale), medical protection against the disease, reports on what has been labeled as "post-polio" problems. Of interest would be location of the cases, how severe, as well as what is being done in the "post-polio" area.
</top><top>
<num> Number : 303
<title> Hubble Telescope Achievements
<desc> Description :
Identify positive accomplishments of the Hubble telescope since it was launched in 1991.
<narr> Narrative :

documents are relevant that show the Hubble telescope has produced new data, better quality data than previously available, data that has increased human knowledge of the universe, or data that has led to disproving previously existing theories or hypotheses. documents limited to the shortcomings of the telescope would be irrelevant. Details of repairs or modifications to the telescope without reference to positive achievements would not be relevant.
</top><top>
<num> Number : 304
<title> Endangered Species (Mammals)
<desc> Description :
Compile a list of mammals that are considered to be endangered, identify their habitat and, if possible, specify what threatens them.
<narr> Narrative :
Any document identifying a mammal as endangered is relevant. Statements of authorities disputing the endangered status would also be relevant. A document containing information on habitat and populations of a mammal identified elsewhere as endangered would also be relevant even if the document at hand did not identify the species as endangered. Generalized statements about endangered species without reference to specific mammals would not be relevant.
</top>

Tab. A.1 – Requêtes 301 à 304 de TREC6

Numéro de la requête	**Titre de la requête**	**Nb. docs pertinents**
301	International Organized Crime	448
302	Poliomyelitis and Post-Polio	65
303	Hubble Telescope Achievements	10
304	Endangered Species (Mammals)	196
305	Most Dangerous Vehicles	35
306	African Civilian Deaths	332
307	New Hydroelectric Projects	210
308	Implant Dentistry	4
309	Rap and Crime	3
310	Radio Waves and Brain Cancer	13
311	Industrial Espionage	182
312	Hydroponics	11
313	Magnetic Levitation-Maglev	93
314	Marine Vegetation	44
315	Unexplained Highway Accidents	67
316	Polygamy Polyandry Polygyny	34
317	Unsolicited Faxes	14
318	Best Retirement Country	113
319	New Fuel Sources	170
320	Undersea Fiber Optic Cable	6
321	Women in Parliaments	203
322	International Art Crime	34
323	Literary/Journalistic Plagiarism	61
324	Argentine/British Relations	161
325	Cult Lifestyles	24
326	Ferry Sinkings	46
327	Modern Slavery	11
328	Pope Beatifications	8
329	Mexican Air Pollution	35
330	Iran-Iraq Cooperation	60
331	World Bank Criticism	213
332	Income Tax Evasion	254
333	Antibiotics Bacteria Disease	65
334	Export Controls Cryptography	9
335	Adoptive Biological Parents	66
...	...	...

Tab. A.2 – Caractéristiques des requêtes TREC6

	MAP		R-précision		P@5
inq102	0,4226	inq102	0,4524	brkly7	0,7600
citya1	0,4012	citya1	0,4217	inq102	0,7440
brkly7	0,3714	brkly7	0,4152	citya1	0,7400
inq101	0,3659	inq101	0,4088	citya2	0,7320
assctv2	0,3539	assctv2	0,3999	assctv2	0,7280
assctv1	0,3504	assctv1	0,3948	assctv1	0,7080
crnlea	0,3419	crnlea	0,3890	eth002	0,6880
citya2	0,3373	citya2	0,3816	crnlla	0,6800
crnlla	0,3302	westp1	0,3780	westp1	0,6800
westp1	0,3157	crnlla	0,3768	pircs1	0,6600

	P@10		P@15		
inq102	0,7220	inq102	0,6867		
brkly7	0,7120	brkly7	0,6813		
citya1	0,7120	citya1	0,6787		
citya2	0,6940	citya2	0,6560		
assctv2	0,6760	assctv2	0,6453		
inq101	0,6580	assctv1	0,6253		
assctv1	0,6540	inq101	0,6213		
brkly6	0,6380	crnlla	0,6160		
crnlla	0,6360	westp1	0,6040		
eth002	0,6360	eth002	0,5973		

Tab. A.3 – Performances initiales des 10 meilleurs systèmes pour TREC3

	MAP		R-précision		P@5
ETHme1	0,3070	uwgcx0	0,3444	ETHme1	0,6160
uwgcx1	0,2954	uwgcx1	0,3392	genrl3	0,5800
uwgcx0	0,2944	ETHme1	0,3305	Cor5M2rf	0,5760
Cor5M2rf	0,2832	LNmFull2	0,3115	uwgcx0	0,5560
LNmFull2	0,2809	LNmFull1	0,3013	uwgcx1	0,5520
LNmFull1	0,2705	Cor5M2rf	0,3012	LNmFull2	0,5360
genrl3	0,2687	CLCLUS	0,2985	LNmFull1	0,5280
LNaDesc2	0,2585	genrl3	0,2941	genrl4	0,5240
LNaDesc1	0,2547	LNaDesc2	0,2929	LNaDesc1	0,5080
genrl4	0,2522	LNaDesc1	0,2771	LNaDesc2	0,4960

	P@10		P@15		
ETHme1	0,5460	ETHme1	0,5147		
Cor5M2rf	0,5080	uwgcx1	0,4707		
uwgcx1	0,5000	uwgcx0	0,4627		
uwgcx0	0,4880	Cor5M2rf	0,4547		
LNmFull2	0,4780	LNmFull2	0,4333		
LNaDesc2	0,4780	LNaDesc2	0,4280		
genrl3	0,4700	CLCLUS	0,4187		
LNaDesc1	0,4700	genrl3	0,4173		
LNmFull1	0,4600	LNaDesc1	0,4173		
genrl4	0,4460	LNmFull1	0,4093		

Tab. A.4 – Performances initiales des 10 meilleurs systèmes pour TREC5

	MAP		R-précision		P@5
uwmt6a0	0,4631	uwmt6a0	0,4896	CLAUG	0,7120
CLAUG	0,3742	CLAUG	0,3914	uwmt6a0	0,7080
CLREL	0,3514	CLREL	0,3639	CLREL	0,7000
anu6min1	0,3044	anu6min1	0,3427	anu6min1	0,6400
LNmShort	0,2902	gerua1	0,3241	gerua2	0,6160
city6at	0,2876	city6at	0,3220	LNmShort	0,6120
gerua1	0,2783	LNmShort	0,3188	gerua1	0,5760
anu6alo1	0,2602	gerua2	0,3022	anu6alo1	0,5400
iss97man	0,2576	anu6alo1	0,2995	Mercure1	0,5360
gerua2	0,2559	iss97man	0,2975	aiatB1	0,4920

	P@10		P@15		
uwmt6a0	0,6820	uwmt6a0	0,6387		
CLAUG	0,6120	CLAUG	0,5493		
CLREL	0,5960	CLREL	0,5280		
anu6min1	0,5600	anu6min1	0,5120		
gerua2	0,5480	gerua2	0,4867		
gerua1	0,5200	gerua1	0,4680		
LNmShort	0,5000	LNmShort	0,4547		
Mercure1	0,4640	Mercure1	0,4080		
anu6alo1	0,4480	aiatB1	0,4040		
city6at	0,4380	anu6alo1	0,4027		

Tab. A.5 – Performances initiales des 10 meilleurs systèmes pour TREC6

	MAP		R-précision		P@5
CLARIT98COMB	0,3702	t7miti1	0,4392	CLARIT98CLUS	0,7600
t7miti1	0,3675	CLARIT98COMB	0,4140	iit98ma1	0,7320
uwmt7a2	0,3587	uwmt7a2	0,4012	uwmt7a2	0,7240
CLARIT98CLUS	0,3525	CLARIT98CLUS	0,3730	CLARIT98COMB	0,6920
CLARIT98RANK	0,3351	CLARIT98RANK	0,3726	harris1	0,6880
iit98ma1	0,3333	iit98ma1	0,3688	uwmt7a1	0,6840
ok7ax	0,3033	uwmt7a1	0,3402	acsys7mi	0,6720
uwmt7a1	0,2983	ok7ax	0,3390	CLARIT98RANK	0,6720
att98atdc	0,2961	Brkly26	0,3370	t7miti1	0,6640
att98atde	0,2940	LNmanual7	0,3190	uoftimgr	0,6600

	P@10		P@15		
CLARIT98CLUS	0,6940	CLARIT98COMB	0,6613		
CLARIT98COMB	0,6940	CLARIT98CLUS	0,6360		
uwmt7a2	0,6540	CLARIT98RANK	0,6320		
CLARIT98RANK	0,6440	t7miti1	0,6213		
iit98ma1	0,6400	iit98ma1	0,6027		
t7miti1	0,6400	uwmt7a2	0,5960		
uwmt7a1	0,6280	uwmt7a1	0,5760		
harris1	0,6160	harris1	0,5533		
uoftimgr	0,5980	ok7ax	0,5347		
acsys7mi	0,5960	INQ502	0,5333		

Tab. A.6 – Performances initiales des 10 meilleurs systèmes pour TREC7

TREC3		TREC5		TREC6		TREC7	
	MAP		MAP		MAP		MAP
combz	0,4243	combEXP	0,3254	uwmt6a0	0,4631	best_sys	0,4304
combSQRT	0,4236	combLIN	0,3229	best_sys	0,4611	CLARITCOMB	0,3702
inq102	0,4226	combPOW	0,3229	combLIN	0,3753	combSQRT	0,3208
best_sys	0,4181	combz	0,3186	combPOW	0,3753	combz	0,3203
combEXP	0,4106	combSQRT	0,3174	combEXP	0,3702	combEXP	0,3102
combLIN	0,4019	best_sys	0,3100	combSQRT	0,3351	combLIN	0,3081
combPOW	0,4019	ETHme1	0,3070	combz	0,3344	combPOW	0,3081
citya1	0,4012	combAVG	0,0051	combAVG	0,001	combAVG	0,0368
brkly7	0,3714						
combAVG	0,0955						
	R-Prec		R-Prec		R-Prec		R-Prec
inq102	0,4524	best_sys	0,3481	uwmt6a0	0,4896	best_sys	0,4714
best_sys	0,4491	combLIN	0,3474	best_sys	0,4872	t7miti1	0,4392
combSQRT	0,4462	combPOW	0,3474	combLIN	0,3977	combLIN	0,343
combz	0,4438	uwgcx0	0,3444	combPOW	0,3977	combPOW	0,343
combEXP	0,434	combz	0,3436	combEXP	0,3853	combEXP	0,3398
combLIN	0,4277	combEXP	0,342	combz	0,3561	combSQRT	0,3397
combPOW	0,4277	combSQRT	0,3314	combSQRT	0,3561	combz	0,3375
brkly7	0,4152	combAVG	0,0051	combAVG	0,003	combAVG	0,069
assctv2	0,3999						
combAVG	0,1559						
	P5		P5		P5		P5
best_sys	0,7882	best_sys	0,6522	best_sys	0,7457	best_sys	0,8054
combz	0,776	combLIN	0,62	CLAUG	0,7120	CLARITCLUS	0,76
combSQRT	0,772	combPOW	0,62	combLIN	0,64	combz	0,636
combEXP	0,772	ETHme1	0,616	combEXP	0,64	combSQRT	0,632
brkly7	0,76	combEXP	0,608	combPOW	0,64	combLIN	0,6
combLIN	0,748	combz	0,596	combSQRT	0,62	combEXP	0,6
combPOW	0,748	combSQRT	0,588	combz	0,616	combPOW	0,6
combAVG	0,08	combAVG	0,004	combAVG	0,02	combAVG	0,08
	P10		P10		P10		P10
combSQRT	0,75	best_sys	0,5748	best_sys	0,6921	best_sys	0,7407
best_sys	0,7447	ETHme1	0,546	uwmt6a0	0,6820	CLARITCLUS	0,694
combz	0,742	combEXP	0,528	combEXP	0,566	combSQRT	0,6
combEXP	0,74	combLIN	0,524	combLIN	0,562	combz	0,598
combLIN	0,726	combPOW	0,524	combPOW	0,562	combLIN	0,576
combPOW	0,726	ombSQRT	0,52	combz	0,538	combPOW	0,576
inq102	0,722	combz	0,514	combSQRT	0,536	combEXP	0,572
combAVG	0,088	combAVG	0,006	combAVG	0,02	combAVG	0,078
	P15		P15		P15		P15
combz	0,7293	best_sys	0,5272	best_sys	0,6534	best_sys	0,7064
combSQRT	0,7267	ETHme1	0,5147	uwmt6a0	0,6387	CLARITCOMB	0,6613
best_sys	0,7077	combSQRT	0,4893	combLIN	0,5093	combSQRT	0,5427
combEXP	0,7067	combz	0,488	combPOW	0,5093	combEXP	0,5413
combLIN	0,6987	combEXP	0,4747	combEXP	0,508	combz	0,5387
combPOW	0,6987	combLIN	0,4613	combSQRT	0,4893	combLIN	0,5387
inq102	0,6867	combPOW	0,4613	combz	0,4853	combPOW	0,5387
combAVG	0,1013	combAVG	0,0053	combAVG	0,0147	combAVG	0,076

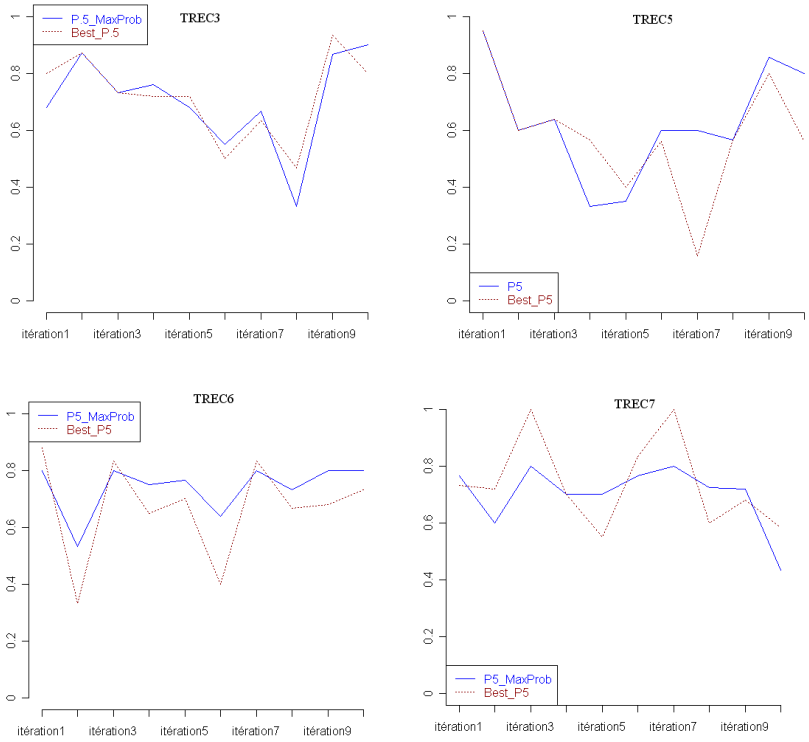

FIG. A.4 – Comparaison de la P5

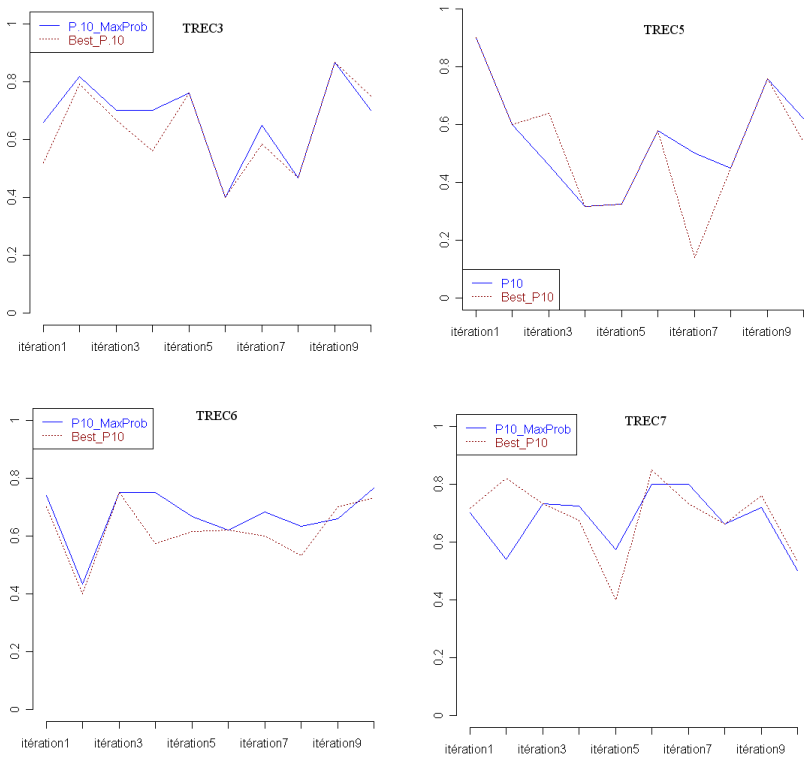

FIG. A.5 – Comparaison de la P10

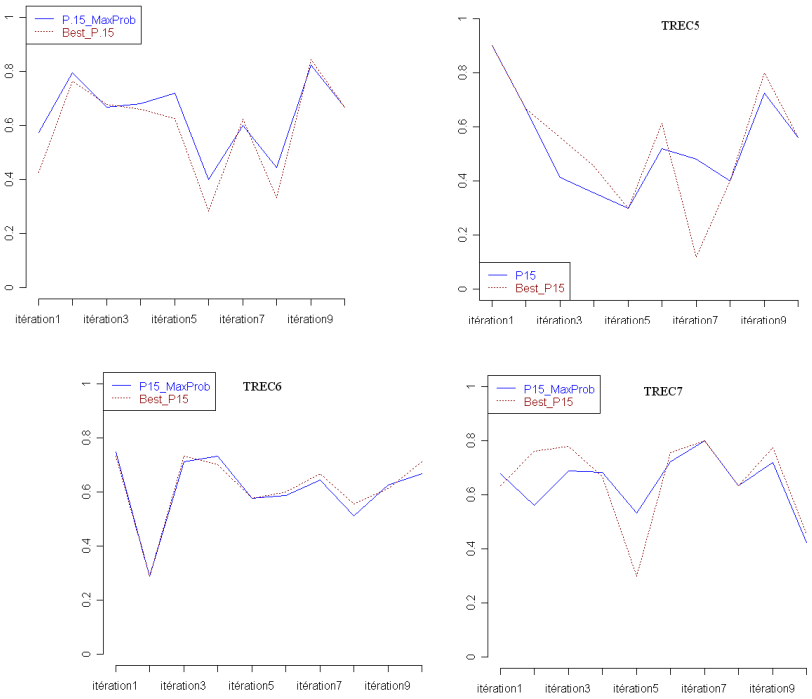

FIG. A.6 – Comparaison de la p15

Annexe B

Annexes

Méthodes non supervisées de classification et de partitionnement de données en RI

La classification consiste à regrouper une collection d'objets dans des classes tels que les objets appartenant à la même classe possèdent un certain degré de similarité.

Dans nos travaux, nous appliquons la classification aux requêtes, afin de les regrouper selon leurs caractéristiques linguistiques. Nous donnons plus de précisions sur la classification des requêtes dans la section 3.4 où nous présentons quelques travaux utilisant la classification des requêtes associée à des techniques de fusion de données. Les travaux présentés dans [HP96], [JvR71], [TVVR02], par exemple, montrent que la classification peut permettre d'améliorer les performances de la recherche en augmentant le rappel. Nous décrivons dans la section B les principes de la CAH , et nous terminons cette section par la présentation de la méthode des k-means dans la section B.

La classification ascendante hiérarchique - CAH

Dans cette section, nous donnons quelques fondements théoriques nécessaires à la compréhension de la CAH [LMP06] et à son application en RI [Wil88].

Le principe de la CAH consiste à répartir des individus dans des classes homogènes. Les individus peuvent être soit des documents [ZGZW01], soit des requêtes [BYH07], ou d'autres critères [KMDB07b]. La répartition se fait selon un processus itératif au cours duquel les individus sont regroupés entre eux ou avec d'autres groupes d'individus déjà formés. Au départ de l'algorithme de CAH , tous les individus sont considérés comme étant des classes à part entière et, grâce à un calcul de distance, les classes les plus proches sont agrégées pour former une nouvelle classe. A l'issue de la classification, un arbre ou *dendrogramme* représente l'arborescence des regroupements successifs, et regroupe tous les individus dans une seule classe et qui représente la décroissance de la hauteur de chaque saut (ou écart de distance), opéré à chaque regroupement. L'exemple B.1 présente la structure d'un *dendrogramme*.

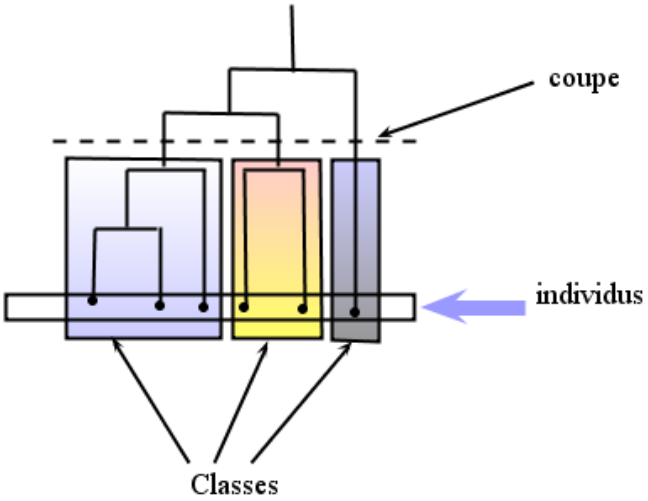

FIG. B.1 – Exemple de dendrogramme

Le nombre de classes est déterminé par le niveau de coupe du *dendrogramme*. L'algorithme de la CAH peut se définir de la manière suivante :

1. regrouper (en une classe) les deux éléments les plus proches et considérer la classe comme un nouvel élément remplaçant les deux anciens.
2. recalculer les distances entre ce nouvel élément et les anciens éléments.
3. retourner en 1. tant qu'il reste plus d'un élément.

Pour pouvoir effectuer le regroupement en classes, il est nécessaire de calculer les distances entre les individus et les classes. Plusieurs mesures existent pour calculer la distance d(A,B) entre deux éléments appartenant à deux classes (A et B) :

- *saut minimum* : la distance entre deux classes est la plus petite distance séparant un élément issu de la première classe, d'un élément issu de la seconde. Ce saut se calcule selon la formule suivante :

$$d(A,B) = min_{(i,j)\in(A\times B)}(d_{i,j})$$

- *saut maximum* : la distance entre deux classes est la plus grande distance séparant un élément issu de la première classe, d'un élément issu de la seconde. Ce saut se calcule selon la formule suivante :

$$d(A,B) = sup_{(i,j)\in(A\times B)}(d_{i,j})$$

- *saut moyen* : la distance entre deux classes correspond à la moyenne arithmétique de toutes les distances entre un élément de la première classe et un élément de la seconde classe.

$$d(A,B) = \frac{1}{card(A).card(B)} \sum_{(i,j)\in(A\times B)} d_{i,j}$$

– *distance des barycentres* : les distances sont calculées entre les vecteurs moyens représentant les différentes classes.

$$d(A, B) = d(g_A, g_B),$$

avec g_A le barycentre de la classe A.

– *saut de ward* [War63] : Cette mesure minimise à chaque regroupement la décroissance de la variance interclasse [LMP06].

$$d(A, B) = \frac{w_A.w_B}{w_A + w_B} d(g_A, g_B)$$

avec w_B correspondant à la pondération des éléments de la classe B.

Nous avons utilisé le critère de ward dans notre étude.

La classification ainsi réalisée génère un arbre que l'on peut couper à différents niveaux pour obtenir des classes plus ou moins importantes ou plus ou moins nombreuses.

Une fois le nombre de classes choisi, on peut mettre en œuvre un algorithme de type agrégation autour de centres mobiles (voir Lebart et al., 2006), encore appelé k-means, de façon à stabiliser les classes obtenues.

La méthode de partitionnement des k-means

La classification par k-means a été proposée par [Mac67]. Elle fait partie des méthodes non supervisées de partitionnement, basées sur le principe de réallocation dynamique des centres des classes. Contrairement à la CAH , la méthode des k-means nécessite la détermination du nombre de classes *à priori*, et se base sur le principe de réallocation dynamique des individus dans les classes. Cet algorithme est itératif ; il initialise les centres des classes en tirant aléatoirement k individus. L'algorithme s'arrête lorsque le critère de convergence [1] est satisfait. En général, une dizaine d'itérations permet d'atteindre la convergence. Cet algorithme itératif peut être décrit de la manière suivante :

1. Déterminer le nombre de classes k.
2. Initialiser les centres des classes correspondants aux centres de gravité des k classes.
3. Affecter chaque individu à la classe dont le centre est le plus proche.
4. Calculer des k centres des classes ainsi constituées, correspondant aux différents barycentres, à chaque fois qu'un individu est affecté à une classe.
5. Repéter les points 3. et 4. jusqu'à ce que les centres ne varient plus.

La qualité de la méthode des k-means dépend des centres des classes qui sont initialement choisis (différents centres produisent différents résultats).

[1] [BB05] : *Le critère (la variance interclasse) est majoré par la variance totale. Il est simple de montrer qu'il ne peut que croître à chaque étape de l'algorithme, ce qui en assure la convergence. Il est équivalent de maximiser la variance interclasse ou de minimiser la variance intraclasse. Cette dernière est alors décroissante et minorée par 0.*.

Liste des publications personnelles

Conférences et workshops internationaux / International conferences articles

Nongdo Désiré Kompaore, Josiane Mothe, Ludovic Tanguy. Combining indexing methods and query sizes in information retrieval in French. Dans : International Conference on Enterprise Information Systems (ICEIS 2008), barcelone, 12-JUN-08-16-JUN-08, ICM, 2008 (à paraître).

Nongdo Désiré Kompaore, Josiane Mothe. Probabilistic fusion and categorization of queries based on linguistic features. Impact on high precision measures. Dans : ACM International Workshop for Ph.D. Students in Information and Knowledge Management (ACM PIKM 2007), Lisboa, Portugal, 09-NOV-07, ACM, p. 63-68, novembre 2007.

Nongdo Désiré Kompaore, Josiane Mothe, Alain Baccini, Sébastien Déjean. Query clustering and IR system detection. Experiments on TREC data. Dans : Large-Scale Semantic Access to Content (Text, Image, Video and Sound) (RIAO 2007), Pittsburgh, USA, 30-MAY-07-01-JUN-07, Centre de hautes études internationales d'Informatique Documentaire (C.I.D.), (support électronique), mai 2007.

Conférences et workshops nationaux / National conferences articles

Gilles Hubert, Nongdo Désiré Kompaore, Josiane Mothe. Réinjection de pertinence pour la fusion de systèmes. Dans : Colloque Veille Stratégique Scientifique et Technologique (VSST 2007), Marrackech (Maroc), 21-OCT-07-25-OCT-07, octobre 2007 (à paraître).

Nongdo Désiré Kompaore, Josiane Mothe, Alain Baccini, Sébastien Déjean. Prédiction du SRI à utiliser en fonction des critères linguistiques de la requête. Dans : Conférence francophone en Recherche d'Information et Applications (CORIA 2007), Saint Etienne, 28-MAR-07-30-MAR-07, Association Francophone de Recherche d'Information et Applications (ARIA), p. 239-254, mars 2007.

Nongdo Désiré Kompaore, Josiane Mothe, Lemoing Emmanuel. Fusion de systèmes pour la recherche de passages dans les textes. Dans : Conférence francophone en Recherche d'Information et Applications (CORIA 2006), Lyon, 15-MAR-06-17-MAR-06,

Association Francophone de Recherche d'Information et Applications (ARIA), p. 295-300, mars 2006.

Nongdo Désiré Kompaore, Josiane Mothe. Etude bibliométrique des quatre éditions du colloque VSST (de 1995 à 2004). Dans : VSST, Veille Stratégique Scientifique et Technologique, Toulouse, NA, p. 229-230, octobre 2004.

Conférences sans actes publiés / Conference articles without published proceedings

Nongdo Désiré Kompaore. Fusion et systèmes de recherche adaptatifs. Dans : Colloque EDIT, Toulouse, 11-APR-05-12-APR-05.

Index