ACADÉMIE D'AIX-MARSEILLE
UNIVERSITÉ D'AVIGNON ET DES PAYS DE VAUCLUSE

THÈSE

Apprentissage et Décision Automatique en Recherche Documentaire : prédiction de difficulté de requêtes et sélection de modèle de recherche

Présentée et soutenue publiquement le 31 mai 2006 pour obtenir le grade de Docteur en Sciences de l'Université d'Avignon et des Pays de Vaucluse

SPÉCIALITÉ : INFORMATIQUE

par
Jens GRIVOLLA

Composition du jury :

Mme	Josiane Mothe	PR, IRIT, Toulouse	Rapporteurs
Mme	Brigitte Grau	MC-HDR, LIMSI, Paris	
M	Marc El-Bèze	PR, LIA, Avignon	Examinateur
M	Renato De Mori	PR, LIA, Avignon	Directeurs de thèse
M	Pierre Jourlin	MC, LIA, Avignon	

École Doctorale Science et Agronomie
Laboratoire d'Informatique n°EA931

Résumé

Cette thèse se situe dans la problématique de la recherche documentaire. Dans ce domaine, chaque besoin en information est exprimé par un utilisateur sous la forme d'une requête en langage naturel. Il existe différentes approches pour traiter ces requêtes, mais les systèmes actuels utilisent généralement une méthode unique, indépendante des caractéristiques de la requête. On peut pourtant montrer de façon expérimentale que la performance relative d'une technique de recherche sur une autre peut varier considérablement suivant la requête traitée. On peut aussi constater, toujours de façon expérimentale, que toute amélioration globale d'un système de recherche d'information se fera au détriment de certains cas pour lesquels les performances se dégradent.

Nous avons abordé cette thématique en proposant des méthodes qui permettent de repérer automatiquement les requêtes qui posent des difficultés particulières au système utilisé, afin de permettre un traitement spécifique et adapté. Ce thème de recherche était encore très peu exploré à l'origine des travaux relatés dans ce document, mais a connu un développement très important ces toutes dernières années.

Ce document présente un grand nombre d'indicateurs de performance, chacun étant accompagné d'une évaluation et comparaison approfondie de leur pouvoir de prédiction. Ceci inclut de nombreuses mesures relatives à la requête, qui décrivent sa spécificité et son ambiguïté. Les mesures sont basées sur des caractéristiques sémantiques ou statistiques des termes de la requête (par ex : synonymes, hyponymes, hypéronymes, fréquences des termes issus d'un vocabulaire spécialisé, etc). D'autres mesures étudiées sont basées sur le résultat de la recherche. Nous avons utilisé en particulier l'homogénéité de l'ensemble des documents trouvés par le système (décrit par la similarité cosinus ou l'entropie), ainsi que des informations issues du processus de recherche, telles que les similarités entre requêtes et documents.

Nous avons ainsi dégagé un certain nombre de fonctions de prédiction de qualité qui obtiennent des résultats comparables à ceux publiés récemment par d'autres équipes de recherche. Mais le pouvoir individuel de classification des requêtes par niveau de "difficulté", reste néanmoins limité.

La particularité et originalité de ce travail a consisté à étudier la combinaison de ces différentes mesures. En utilisant des méthodes de classification automatique, nous avons obtenu des prédictions relativement fiables sur la base de mesures qui individuellement ont

un pouvoir de discrimination considérablement plus faible. En particulier, les méthodes de classification fondées sur les Support Vector Models (SVM) et ou celles fondées sur les arbres de décision se sont révélées très performantes pour ce type d'application.

Nous avons évalué nos méthodes pour estimer de manière automatique la qualité des résultats fournis par différents systèmes de recherche documentaire. Pour certains des systèmes ayant participé à la campagne d'évaluation TREC 8 adhoc, nous obtenons une fiabilité de nos prédictions allant jusqu'à 86% (pour une décision binaire facile/difficile).

Nous avons également adapté, avec des résultats très encourageants, notre approche à d'autres contextes d'application. Nous avons ainsi développé une méthode de décision pour l'application sélective de techniques d'enrichissement de requêtes, ainsi que pour la sélection du modèle de recherche le plus approprié pour chaque requête.

La robustesse de notre approche est aussi démontrée par les bonnes performances obtenues sur un corpus en français, issu des campagnes question/réponse EQueR, sur la base des résultats fournis par le système SIAC, développé au sein du Laboratoire Informatique d'Avignon (LIA).

Remerciements

Je tiens à remercier Brigitte Grau et Josiane Mothe qui m'ont fait l'honneur d'avoir rapporté sur ce mémoire et ont pris le temps de me faire parvenir des commentaires détaillés et très utiles sur le manuscrit. Je remercie également Monsieur Marc El-Bèze qui en plus de sa participation au jury a contribué à la correction et l'amélioration de ce document.

Je remercie bien évidemment mes directeurs de thèse Pierre Jourlin et Renato de Mori qui m'ont permis de travailler de manière autonome et poursuivre mes idées et intérêts, tout en étant disponibles et prêts à apporter leur expérience et connaissances en cas de besoin.

De façon générale, je remercie tous les membres du personnel du Laboratoire Informatique d'Avignon, qui m'ont permis de travailler (et vivre) dans une ambiance et bonne humeur qui me semble inégalable. En particulier, je tiens à remercier Christian Raymond et Mireille Palpant, avec qui j'ai vécu cette expérience depuis notre année de DEA. Je n'ose pas aller plus loin dans les remerciements individuels, trop de personnes me tenant à coeur pour se limiter ici à une petite sélection...

Merci à mes parents et toute ma famille, et également à Kadi pour m'avoir donné de la force dans les derniers moments de ma thèse, qui n'étaient pas les plus faciles.

Table des matières

1. Introduction

1.1. Problématique

La recherche documentaire est un domaine d'importance croissante depuis déjà un certain nombre d'années. Avec l'omniprésence de bases de documents gigantesques, en particulier l'internet, il y a un besoin de systèmes performants pour rendre ces données utilisables.

Dans les dix dernières années, beaucoup de progrès ont été faits et les systèmes de recherche dont nous disposons aujourd'hui sont bien supérieurs aux systèmes plus anciens en termes de précision et rappel des documents trouvés.

Cependant les résultats sont loin d'être parfaits et on note une certaine stagnation dans les performances depuis trois ou quatre ans, du moins pour les recherches classiques représentées par la partie *adhoc* des campagnes TREC (voir chapitre 4). Cette tendance est illustrée dans la figure 1.1 (voir [Harman, 2000]). On y trouve les performances obtenues par différentes générations du système de Cornell University, sur les différentes collections de requêtes (et documents) des campagnes TREC.

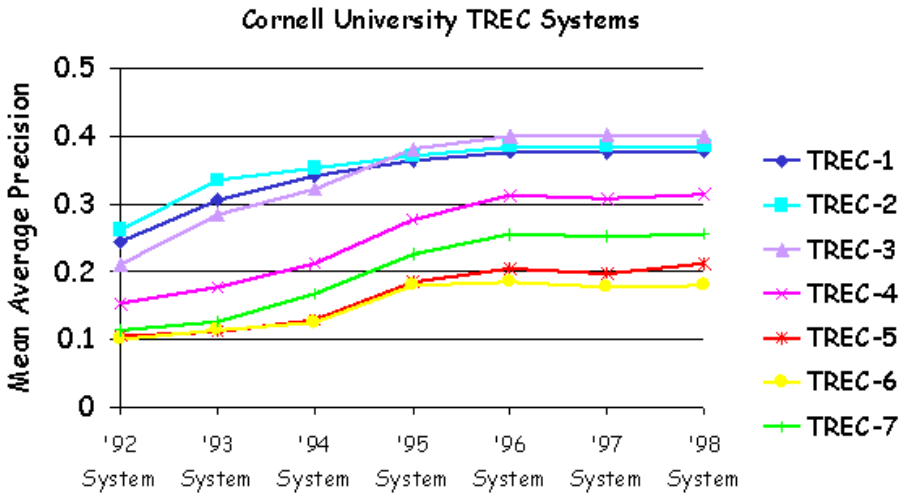

FIG. 1.1.: Évolution des performances d'un participant à TREC (adhoc)

Pour pouvoir améliorer encore les performances, il nous semble donc nécessaire d'en-

visager d'autres approches, en particulier des approches adaptées dynamiquement à la requête posée. Dans le cas d'un échec des méthodes automatiques il est envisageable d'aiguiller l'utilisateur vers un approche interactive, permettant de préciser et améliorer les requêtes par un dialogue avec l'utilisateur.

Au-delà du développement de nouveaux modèles et de nouvelles théories fondamentales pour la recherche documentaire (dont certains sont présentés très brièvement dans la section 5), beaucoup d'améliorations ont été apportées aux modèles existants. Les performances ont ainsi pu être considérablement augmentées, en particulier par des méthodes d'enrichissement de requêtes, c.à.d. des modifications des requêtes basées sur un ajout de termes (voir le la section 5.7) afin de trouver des documents pertinents supplémentaires.

Malheureusement, ces méthodes ne sont efficaces que pour une partie des requêtes et peuvent même dégrader les résultats pour d'autres (voir la section 8). D'autre part, pour certaines requêtes aucun système n'arrive à fournir de réponses satisfaisantes.

Deux problématiques découlent des points précédents :

- la déterminaison du traitement optimal pour chaque requête
- l'identification des requêtes pour lesquelles la recherche échoue afin de poursuivre, par exemple, une approche interactive

1.2. Notre approche

L'approche que nous proposons se base sur l'utilisation d'un système de recherche documentaire existant. Dans le fonctionnement classique d'une recherche *ad hoc*, l'utilisateur soumet une requête qui est traitée par le système qui répond par une liste de documents, classée par ordre décroissant de probabilité de pertinence (voir figure 1.2).

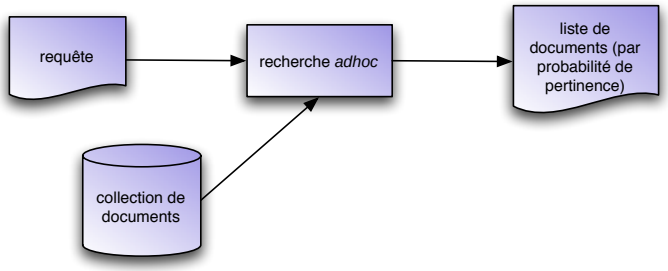

FIG. 1.2.: Recherche *ad hoc* classique

À ce fonctionnement nous rajoutons une analyse de la requête ainsi qu'éventuellement les résultats obtenus. Sur la base de cette analyse le système pourra être adapté spécifiquement à la requête donnée, par exemple par l'application contrôlée de techniques d'enrichissement des requêtes, de manière à obtenir un traitement optimal de chaque requête

(figure 1.3).

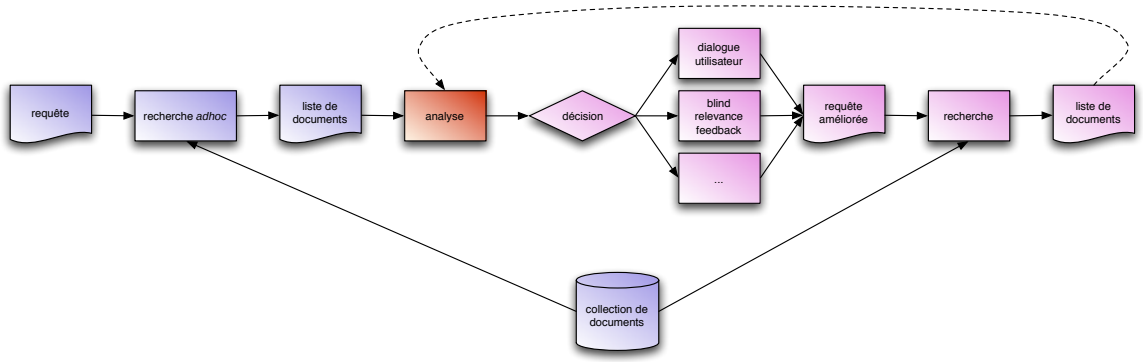

FIG. 1.3.: Recherche avec processus de décision

Ceci peut en principe être appliqué de manière itérative, si plusieurs techniques d'adaptation sont disponibles. Le processus de décision estime alors la probabilité de succès de chaque méthode afin d'appliquer les techniques les plus adaptées à la requête.

1.3. Plan

Ce document est structuré en deux grandes parties :

La première partie décrit de manière très succincte le domaine de la recherche documentaire. Le lecteur désireux d'approfondir ses connaissances pourra se reporter sur un grand nombre d'ouvrages bien plus complets dans cette thématique, par exemple [Baeza-Yates, 1999] ou [Salton, 1989].

La deuxième partie traite spécifiquement de la problématique abordée dans ces travaux. C'est dans cette partie, nettement plus longue que la précédente que nous développons nos méthodes, notre contexte expérimental et nos résultats.

En annexe se trouvent des travaux sur une thématique différente, effectués dans le cadre d'une coopération industrielle.

Première partie .

La recherche d'information

Il y a seulement quelques décennies, la quantité d'information que l'on pouvait manipuler par ordinateur était tellement faible qu'il était indispensable de transformer manuellement des données brutes, pour les mettre sous une forme très formelle (par ex : bases de données SQL). Le document était donc complété par des informations supplémentaires comme un résumé ou des descripteurs prédéfinis. Il était alors relativement simple d'accéder aux informations voulues en formulant une requête *logique* sur la base de donnée disponible.

Avec le développement des moyens de stockage numérique et des réseaux, la quantité d'information disponible à partir d'un poste de travail a suivi une croissance remarquable. En particulier, la quantité de données *brutes* (textes en langage naturel, enregistrements audio, images, vidéo) a suivi un accroissement phénoménal. Cela n'a pu se réaliser que grâce à une absence de contraintes vis-à-vis de la structure et de la cohérence de ces informations.

Ainsi, une part grandissante des travaux de recherche en informatique consistent aujourd'hui à développer des méthodes et des techniques permettant de structurer automatiquement ces informations (*Extraction d'information*) ou de faciliter la localisation d'une information recherchée par un utilisateur (*Recherche documentaire*).

Dans ce mémoire, nous nous focalisons dans un sous-domaine précis de la recherche d'information : la recherche documentaire textuelle.

2. Un très bref historique

Si on considère le domaine de la recherche documentaire textuelle au sens large, l'historique remonte à l'année 1247, quand Hugo de Saint Caro employa 500 moines pour écrire le premier index de la Bible[Wheatley, 1879] (référencé par [Swanson, 1988]). Par souci évident de concision, nous allons nous limiter à un historique de ces 40 dernières années, et aux méthodes qui appartiennent au domaine de l'informatique *moderne*. Nous allons aussi réduire cet historique aux seuls aspects qui ont un rapport raisonnablement direct avec le but que nous poursuivons : prédire la qualité du résultat de la recherche documentaire. Le lecteur intéressé par un état de l'art plus complet pourra se référer, par exemple, à [Baeza-Yates, 1999].

2.1. 1962-1992

Au début de cette période, les efforts étaient concentrés, non sur la recherche documentaire à proprement parler, mais plutôt sur l'indexation des documents. Ainsi, en 1962, les premières expérimentations sur les langages d'indexation furent publiés : Les tests de Cranfield [Cleverdon, 1962] et [Cleverdon *et al.*, 1966] montraient que l'indexation automatique était comparable à l'indexation manuelle. Parallèlement, la base expérimentale des tests Cranfield (1400 documents et 225 requêtes) était développée. Pendant les 30 années qui ont suivi, un nombre toujours croissant de chercheurs ont eu une approche expérimentale de la recherche documentaire, basée sur la collection de Cranfield, la collection *CACM* [Fox, 1983], la collection *NLP* [Spärck Jones et Webster, 1979].

Dans les années 70, les premiers modèles pour la recherche sur les documents entiers en langage naturel ont été développés, tels que le modèle vectoriel (Smart, [Salton, 1971]) ou le modèle probabiliste (Okapi, [Robertson et Spärck Jones, 1976]).

2.2. 1992-2006

En 1992, le *National Institute of Technology* (NIST) a démarré une série de campagnes d'évaluations, nommées *Text REtrieval Conference* (TREC) afin de combler deux faiblesses récurrentes dans les expérimentations antérieures. Le premier élément manquant était un cadre d'expérimentation commun, qui permette un effort de recherche concerté et

des comparaisons fiables des différents systèmes, même si les collections de Cranfield et CACM étaient utilisées par de nombreux chercheurs. La deuxième faiblesse était la taille des collections de documents, considérée comme non-réaliste dès le début des années 90 [Harman, 1992]. Ainsi, la collection CACM ne contenait que 3204 documents, Cranfield 1400, et NLP 11429. De plus, ces collections étaient considérablement moins généralistes que celles introduites à TREC.

Au regard du premier objectif, le nombre de systèmes participant à TREC est passé de 25 en 1992 à 103 en 2004. Au regard du second objectif, le nombre de documents a peu varié : il était de 700.000 en 1992 (*adhoc track*) et de 500,000 en 2004 (*robust track*). Le nombre de requêtes (*topics*) est lui est passé de 50 en 1992 (*adhoc track*) à 250 en 2004 (*robust track*). D'autres pistes comme le *web track* ont été rajoutées et comprennent des collections de documents plus grandes.

Mais le plus important est le nombre de jugements de pertinence : il y avait, en moyenne, 1500 documents jugés chaque année pour chaque requête. L'existence de ces jugements est un atout remarquable pour les chercheurs dans le domaine. En outre, un nombre considérable de contextes ont été explorés : piste interactive, manuelle, automatique, documents parlés, web, multi-langues, etc. C'est donc dans ce cadre expérimental que nous situons nos propres travaux.

3. Pertinence et mesures de performance

3.1. Pertinence

La performance en recherche documentaire ou la *difficulté* de la recherche est bien entendu dépendante de la requête, de la base documentaire, et du système de recherche. Pour quantifier cette performance, il faut faire des choix qui se rapprochent le plus possible de l'idée que se fait un utilisateur de la qualité du résultat de la recherche et par conséquent de la pertinence des documents retrouvés pour la requête qu'il a formulé.

Or, la pertinence d'un document est une notion extrêmement complexe (voir, par exemple, [Mizzaro, 1998]) et donc difficile à définir. Pour simplifier, on pourrait la voir, pour une requête donnée, comme une valeur d'un espace continu, éventuellement à plusieurs dimensions, reflétant le degré de satisfaction de l'utilisateur.

Cependant, il est assez traditionnel (et fréquemment utile) d'aborder les problèmes complexes par l'angle de la statistique. Il faut alors trouver un bon compromis entre nombre de variables aléatoires à estimer et nombre d'observations disponibles pour les estimer. Ainsi, dans leur immense majorité, les modèles de recherche documentaire considèrent la pertinence comme une valeur binaire : un document est pertinent ou il ne l'est pas. On dira aussi qu'un document est désirable (ou indésirable). Dans tous les cas, la pertinence est une valeur qui n'est valide en théorie que pour une requête et un utilisateur donné, voire même pour un utilisateur donné à un instant t donné.

Une fois de plus, la problématique est traditionnellement simplifiée, cette fois avec l'hypothèse selon laquelle la pertinence est une valeur universelle, une application qui a tout couple document-requête associe une valeur binaire et une seule. Certaines études montrent à quel point cette simplification est fragile (voir, par exemple, [Tombros et Crestani, 1999]), mais le nombre de données disponibles ne permet toujours pas aujourd'hui d'estimer des pertinences dépendantes de l'utilisateur.

Travaillant essentiellement sur la base des données issues de TREC, nous avons adopté la simple notion binaire de pertinence sur laquelle se base l'évaluation lors des campagnes.

3.2. Mesures de performance

Nous nous plaçons donc dans le contexte le plus traditionnel : pour chaque requête, le système renvoie une liste ordonnée de documents. Pour le système, plus un document est proche de la tête de liste, plus il a de chance d'être considéré par l'utilisateur comme pertinent.

Si l'on dispose d'une collection de documents, d'un ensemble de requêtes et de jugements de pertinence, il est possible d'estimer la performance globale de différent systèmes. Traditionnellement, on cherchera à améliorer deux mesures inversement dépendantes :

- La *précision* : parmi les documents présents dans la listes renvoyée par le système, le pourcentage de ceux qui sont pertinents.
- Le *rappel* : parmi les documents pertinents dans la collection, le pourcentage de ceux que l'on retrouve aussi dans la liste renvoyée par le système.

Ces deux mesures dépendent de la longueur de la liste qui est retournée par le moteur de recherche. Bien entendu, on peut augmenter le rappel en augmentant la longueur de la liste (100% dans le cas d'une liste qui contient tous les documents de la collection), ce qui aurait pour effet de ramener la précision au pourcentage global de documents pertinents dans la collection. Inversement, si l'on fixe la longueur de la liste à un seul document on aura seulement 2 cas possibles :

- Le document retourné est pertinent : la précision est de 100%, le rappel est de 100 divisé par le nombre de documents pertinents dans la collection.
- Le document retourné n'est pas pertinent : la précision et le rappel sont tous deux de 0%.

Selon l'application, on peut fixer la taille de la liste à une longueur N qui représente le nombre de documents qu'il semble utile de présenter à l'utilisateur. La précision sur cette liste de document est souvent notée $P@N$.

Pour obtenir une mesure relativement indépendante de la longueur de la liste, on utilise traditionnellement la *précision moyenne*. La précision moyenne tient compte de différents aspects de la performance de recherche : précision, rappel et classement relatifs des documents retrouvés. Elle est définie de la façon suivante :

Soit D l'ensemble des documents de la collection, $Pert_r \in D$ l'ensemble des documents pertinents pour une requête r donnée. $Rang_r(d)$ est la position du document d dans la liste ordonnée, renvoyée pour r par le système de recherche documentaire. Soit $P_r(n)$ la précision pour la requête r en considérant seulement les n documents les mieux classés.

La précision moyenne $PM(r)$ est alors définie comme :

$$PM(r) = \frac{\sum_{d \in Pert_r} P_r(Rang_r(d))}{|Pert_r|}$$

Cette mesure peut paraître peu intuitive, mais elle est largement acceptée comme un bon

indicateur de performance dans une grande variété de contextes applicatifs.

Pour la comparaison des performances de différents systèmes de recherche documentaire, ou de différentes versions d'un même système, on considère généralement la moyenne des précisions moyennes sur un ensemble de requêtes (*mean average precision*).

Dans certains contextes d'autres mesures sont plus appropriées, en particulier quand on s'intéresse au performances pour les requêtes les plus difficiles. Dans le cadre du *robust track* de TREC différentes mesures ont été introduites, dont la moyenne géométrique des précisions moyennes. Une comparaison plus détaillée de ces différentes mesures est présentée dans le cadre de l'étude sur l'enrichissement des requêtes (chapitre 8) et la sélection de modèle.

4. Le contexte expérimental

Nos expérimentations ont été menées sur des données issues des campagnes TREC (*Text REtrieval Conference*[1]) du NIST (*National Institute for Standards and Technology*). Ces données sont composées (pour la tâche *adhoc*) :

- d'une collection d'environ 500.000 documents
- d'un ensemble de requêtes (environ 50 par campagne annuelle pour la piste *adhoc*)
- des jugements de pertinence pour un grand nombre de documents retournés par l'ensemble des systèmes participants.

Les campagnes TREC ont été lancées en 1992, donnant l'opportunité à un grand nombre d'équipes de recherche de tester et de comparer leurs systèmes et leurs progrès, d'années en années jusqu'à aujourd'hui. Les résultats des équipes participantes à chaque campagne ont été évalués manuellement et les jugements de pertinence ainsi produits ont été rendus disponibles, formant une base pour l'évaluation automatique de systèmes de recherche documentaires modifiés, voire nouveaux.

Il a été montré dans [Voorhees et Harman, 1999] ou, avec de plus amples détails, dans [Zobel, 1998] que malgré quelques imperfections, cette évaluation automatique était adéquate, ce que nous avons pu vérifier en réalisant une évaluation manuelle d'une partie des documents retrouvés par nos systèmes.

Bien que TREC soit composé de plusieurs *pistes* telles que, par exemple, *question/réponses* ou *documents parlés*, notre travail est principalement fondé sur les données de la piste *adhoc*. Dans ce cadre précis, la recherche documentaire est automatique, non-interactive, et considère des requêtes exprimées en langage naturel écrit (en langue anglaise).

Les requêtes sont disponibles en plusieurs versions :

- *Titre* : quelques mots clés.
- *Description* : petites phrases décrivant le sujet
- *Narration* : description détaillée faisant apparaître précisément le besoin en information.

Nous avons utilisé les *soumissions* de plusieurs équipes ayant participé à la campagne TREC-8 (voir [Voorhees et Harman, 1999]). Nous avons également en partie utilisé les résultats obtenus par notre propre système de recherche documentaire, en particulier pour les expérimentations concernant l'enrichissement des requêtes.

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

Nous avions initialement travaillé sur les données de TREC-6 (1997) jusqu'à TREC-8 (1999), mais ceci posait un certain nombre de problèmes (changement des systèmes participants, modification de la construction des requêtes). Pour cela, et à cause de travaux préalables effectués par Claude de Loupy [de Loupy, 2000] qui avaient également utilisé la collection de TREC-6, ce document fera à certains endroits référence à des résultats obtenus sur ce corpus, surtout quand ceux-ci sont contraires aux observations sur TREC-8.

5. Les modèles de recherche documentaire

Sommaire

Les systèmes de recherche documentaire qui se sont imposés au cours des campagnes TREC reposent tous sur des méthodes numériques basées sur un comptage des occurrences des termes de la requête dans les documents de la collection. Les méthodes plus sophistiquées, basées par exemple sur la transformation de la requête en formule booléenne se sont en général montrées assez peu prometteuses dans le cadre d'un processus de recherche documentaire entièrement automatique.

Nous allons ici présenter quelques modèles qui ont eu un succès remarquable. En définitive, chaque modèle aboutit à une formule qui permet de calculer le score d'un document pour une requête donnée. Un système renvoie donc à l'utilisateur la liste des documents rangés par ordre décroissant de score. Si les formules sont similaires, les approches sont assez différentes et méritent donc d'être évoquées. On notera aussi que chaque modèle peut être fondé sur des mesures simples de présence/absence des termes de la requête dans les documents.

Cependant, des améliorations significatives ont été obtenues par l'ajout d'une pondération sur les termes qui prend en compte la fréquence d'un terme à l'intérieur d'un document[1] et la fréquence de l'existence d'un terme dans les documents de la collection[2]. Du fait de ces améliorations, tous les modèles ont adopté, à un moment ou un autre, sous une forme ou une autre, ce type de pondération.

Voici les trois modèles les plus utilisés (par degré de formalisme croissant), avec une description très brève de leur philosophie :

[1] en anglais : Term Frequency (TF)
[2] en anglais : Inverse Document Frequency (IDF)

- *Smart* [Salton, 1971] : Le score est calculé comme une distance entre les vecteurs unigrammes du document et de la requête.
- *Okapi* [Robertson et Spärck Jones, 1976] : Le score est calculé comme une estimation de la probabilité de pertinence d'un document
- InQuery [Callan *et al.*, 1992] : le score est calculé à partir d'un réseau bayésien d'inférences.

Ces modèles se sont tous montrés d'un niveau de performance sensiblement équivalent. Cependant, les scores obtenus, s'il permettent d'obtenir un classement relatif des documents satisfaisant, ne constituent pas en soi un bon indicateur absolu de la probabilité qu'un document soit pertinent pour une requête donnée. Comme nous le montrerons en section 7.1.5, la corrélation entre la valeur absolue des scores et la précision moyenne est assez variable selon les systèmes.

Pour certaines de nos expérimentations, les données des systèmes ayant réellement participé à TREC se sont révélées insuffisantes, ou trop rigides. Nous avons alors utilisé une implémentation spécifique du système Okapi, développée dans le cadre de la piste *Recherche de documents parlés* [Spärck Jones *et al.*, 2001] de TREC entre 1998 et 2000. Le système a été légèrement adapté pour la piste *adhoc*.

L'utilisation d'un système relativement classique nous permet d'obtenir des résultats qui devraient se vérifier aisément pour un grand nombre d'autres systèmes. Nous avons d'ailleurs parfois pu le confirmer par des comparaisons avec les résultats des systèmes ayant participé à TREC.

5.1. Modèles booléens

Dans les modèles booléens, le paradigme consiste à retrouver tous les documents qui satisfont une formule booléenne construite à partir des occurences des termes de la requête : par example, les documents contenant au moins 1 occurence du terme n°1 **ET** une occurence du terme n°2 **OU** une occurrence du terme n°3. Le résultat de cette formule appliquée aux documents est une liste non-ordonnée de documents considérés comme pertinents et une liste disjointe et non-ordonnée de documents considérés comme non-pertinents. L'évaluation de tels résultats en terme de précision/rappel est relativement simple, mais souvent peu flatteuse. La catégorisation des documents en 2 classes est d'une telle rigidité que les conjonctions de termes dans la requête peuvent améliorer la précision mais au prix d'une possible diminution drastique du taux de rappel. Inversement, les disjonction de termes dans la requête peuvent améliorer le rappel mais au prix potentiel d'une chute considérable de la précision.

Qui plus est, il peut être assez difficile de dériver automatiquement une formule booléenne à partir d'une requête exprimée en langage naturel. Mais même en se plaçant dans

un cadre semi-automatique, où l'utilisateur a la compétence nécessaire pour exprimer lui-même son besoin en information sous la forme d'une requête booléenne, il n'a pas été établi que ces modèles présentaient un intérêt réél par rapport aux systèmes que nous allons présenter plus loin.

Ces derniers affectent un score à chaque document pour chaque requête et présentent à l'utilisateur, une liste ordonnée de documents. Dans ce cadre, on suppose que l'utilisateur évalue les documents les uns après les autres, jusqu'à ce qu'il ait satisfait son besoin en information. On peut pressentir que ces systèmes seront plus souples, les occurences de chaque terme de la requête dans un document pouvant apporter une contribution plus ou moins importante au score final.

On trouvera une critique de ce type d'approche dans, par exemple, [Cooper, 1988].

5.2. Modèles vectoriels

Dans ce modèle, les documents et les requêtes sont considérés comme des vecteurs dont les composantes correspondent aux occurences des termes. Dans le modèle élémentaire, chaque document est représenté comme un vecteur dont chaque élement D_t peut prendre la valeur 0 (absence du terme t de la requête) ou 1 (présence du terme t de la requête). Il est alors facile de définir une distance entre documents et requêtes. Le système retourne donc à l'utilisateur, une liste de documents classés par ordre de distance décroissante.

Un des modèles vectoriels les plus connus, qui pondère les termes par un poid de type $TF.IDF$, est décrit dans [Salton, 1971, Salton, 1989].

5.3. Modèles probabilistes

Dans le modèle probabiliste, on cherche à estimer la probabilité relative qu'un document soit pertinent, en prenant comme variables statistiques la présence ou l'absence des termes de la requête. L'approche est un peu plus *formelle* que dans le cas des modèles vectoriels et le choix des formules de scoring est plus restreint : toute formule de scoring issue d'un modèle probabiliste peut être utilisée comme distance pour un modèle vectoriel, mais il n'y a pas réciprocité. Dans la pratique pourtant, les formules qui se sont révélées les plus efficaces dans les 2 types de systèmes sont très similaires.

Un modèle probabiliste de référence est décrit dans [Robertson *et al.*, 1996].

5.4. Réseaux d'inférences

Ce modèle est à la fois plus formel que le modèle probabiliste et plus général que le modèle vectoriel. Il s'agit de représenter la recherche documentaire comme un processus d'in-

férences dans un réseau bayésien d'inférences. Encore une fois, si le spectre des formules possibles est plus large que les 2 modèles précédents, les formules réellement efficaces sont très similaires à celles obtenues pour les modèles probabilistes et vectoriels.

Un des systèmes les plus connus basé sur ce type de modèle est décrit dans [Callan *et al.*, 1992].

5.5. Modèles de langages

Dans ce cadre, on construit un modèle stochastique du langage ayant virtuellement généré à la fois les documents et les requêtes. La similarité entre documents et requêtes sera vue comme la similarité entre les paramêtres des modèles de langages sous-jacents. Encore une fois, les formules efficaces de scoring dérivées de cette approche sont assez similaires à celles obtenues avec les autres modèles.

Des expérimentations basées sur ce type d'approche sont commentées dans [Ponte et Croft, 1998].

5.6. Amélioration des modèles de base

Il existe un ensemble de modifications et d'améliorations des 4 modèles décrits ci-dessus. On notera par exemple, la pondération relative des termes de la requête par leur fréquence respective dans la collection de document (*inverse document frequency (IDF)* ou *collection frequency weight (CFW)*), la prise en compte du nombre d'occurences des termes de la requête dans chaque document (*term frequency*), la normalisation des scores par la longueur des documents (*document length normalisation* ou *term density*), la recherche des passage les plus pertinents dans un même document, la recherche d'optimalité pour les différents paramètres du système, etc.

Selon le cas, ces améliorations peuvent être ajoutées pour complémenter le modèle de base, ou alors être fondamentalement intégrées dans la théorie du modèle.

5.7. Techniques d'enrichissement des requêtes

Un problème important rencontré en recherche documentaire dans un contexte ouvert, est le fait que les mots choisis par l'utilisateur dans sa requête ne correspondent pas forcément à ceux contenus dans les documents pertinents (*word mismatch*). Dans certains contextes, avec une indexation manuelle, une liste fermée de mots clés peut être définie, mais dans la majorité des situations ceci n'est pas possible.

Un certain nombre d'approches ont été développées pour traiter ce problème. Ceci inclut l'utilisation de thesaurus, mais aussi des méthodes à base de cooccurrences de termes dans

le corpus de documents.

Dans certains cas il est possible d'affiner la recherche sur la base de documents connus comme étant pertinents, afin de trouver d'autres documents similaires (retour de pertinence, *relevance feedback*). Ceci nécessite l'interaction avec l'utilisateur qui devra désigner les documents pertinents dans une liste proposée par le système.

Il existe une méthode performante et très utilisée pour enrichir les requêtes automatiquement (sans interaction avec l'utilisateur) : la méthode du Blind Relevance Feedback (BRF)[3] ou Pseudo-Relevance Feedback [Walker et de Vere, 1990].

La méthode est fondée sur l'hypothèse selon laquelle, dans la liste de documents retournés par le système, les documents les mieux classés ont une forte probabilité d'être pertinents. Il s'agit d'ajouter à la requête, un nombre t de termes, qui caractérisent au mieux les d *meilleurs* documents. Pour le système que nous avons utilisé dans le chapitre 8, les termes sont choisis suivant le critère de plus fort *offer weight*[4](OW) [Spärck Jones *et al.*, 1998] :

$$ow_t = r_t \log \frac{(r_t + 0.5)(N - N_t - B + r_t + 0.5)}{(N_t - r_t + 0.5)(B - r_t + 0.5)}$$

Les t termes qui obtiennent les ow_t les plus élevés sont ajoutés à la requête initiale. Si la requête initiale a donné d'assez bons résultats en terme de précision sur les D meilleurs documents, alors la requête enrichie peut permettre d'augmenter le score des documents pertinents et donc, d'améliorer la précision et le rappel. Une autre formule pour ce type d'expansion des requêtes est décrite dans [Rocchio, 1971].

Un grand nombre d'expérimentations ont montré que suivant les requêtes, la méthode BRF pouvait causer une petite dégradation des performances sur les requêtes les plus difficiles, mais qu'en faisant une moyenne sur plusieurs dizaines de requêtes , on pouvait constater une amélioration significative de la précision moyenne.

Le chapitre 8 présente une étude plus détaillée sur l'effet du blind relevance feedback, ainsi que du paramétrage, pour aboutir sur l'application sélective de l'enrichissement selon la requête à traiter.

5.8. Comparaison des formules de scoring et des résultats

Il a été noté précédemment (voir p.ex. [Spärck Jones, 1999]) que les différent modèles utilisés en recherche documentaire étaient finalement assez proches dans leur manière de comparer les documents aux requêtes. Une faible distance vectorielle entre document et requête correspondra ainsi généralement à une forte probabilité de pertinence du document dans le modèle probabiliste, un modèle de langage proche pour requête et document, et

[3] littéralement : *retour de pertinence en aveugle*
[4] littéralement : poids d'offre.

ainsi de suite.

Par contre, pour l'estimation de la difficulté de la requête, de petites variations peuvent avoir une grande importance. Ainsi des normalisations différentes des scores utilisés pour classer les documents, un comportement différent des systèmes selon la longueur de la requête, ou encore des différences dans le traitement de termes sémantiquement proches, peuvent provoquer des différences considérables pour certaines requêtes.

On constate ainsi que des systèmes proches en termes de performance globale peuvent avoir un comportement très différent selon les requêtes. Aucun système ne parvient à traiter de manière optimale chaque requête. À part pour quelques requêtes particulièrement difficiles (ou pour lesquelles le corpus ne comprend pas de documents pertinents), dans quel cas tous les systèmes échouent, et quelques requêtes pour lesquelles tous les systèmes obtiennent des résultats d'assez bonne qualité, la majorité des requêtes sera bien traitée par certains systèmes et moins bien par d'autres.

La figure 5.1 (présentée par Donna Harman entre autre dans [Harman, 2000] et plus récemment à [Carmel *et al.*, 2005]) montre pour chaque requête de TREC 8 la performance du meilleur système participant pour cette requête, ainsi que les performances de deux participants à la campagne, les systèmes Okapi (ok8alx) et Claritech (CL99XT).

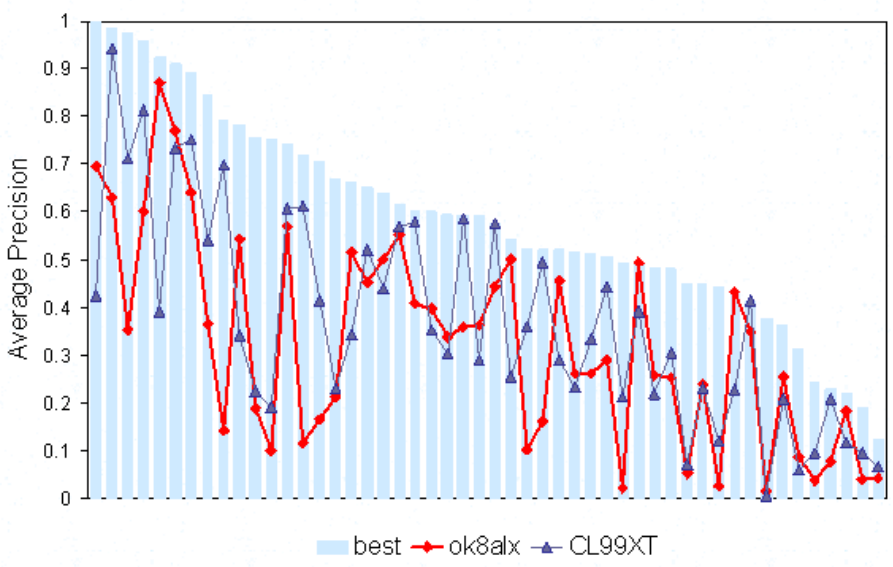

FIG. 5.1.: Différence des performances de différents systèmes par requête

Alors que les deux systèmes ont obtenu de bonnes performances globalement, ils diffèrent fortement selon les requêtes. Pour certaines requêtes, le système Okapi est considérablement plus performant que Claritech, alors que pour d'autres le contraire est le cas.

Le chapitre 7 montre que les performances de différents systèmes sont corrélés de ma-

nières différentes avec certaines caractéristiques des requêtes. Le chapitre 8 illustre comment exploiter ce fait pour choisir le modèle de recherche le plus adapté pour chaque requête individuelle.

Deuxième partie .

Prédiction de la Qualité de Recherche

Au-delà de fournir à l'utilisateur d'un système de recherche documentaire la meilleure liste possible de documents pour sa requête, il est aussi important d'estimer la qualité des résultats.

Avoir une bonne estimation de la performance en termes de précision pour une requête spécifique peut permettre de proposer à l'utilisateur des stratégies adaptées pour améliorer la qualité des résultats pour des requêtes pour lesquelles on obtient initialement une précision faible. Ces stratégies peuvent en particulier inclure de l'interaction avec l'utilisateur ou l'application de techniques coûteuses en temps de calcul ou autres ressources qu'on préférerait éviter quand elles ne sont pas nécessaires.

L'analyse de la difficulté d'une requête peut ainsi être intégrée dans le processus de recherche, qui se décompose alors en recherche initiale, analyse, et éventuellement une nouvelle phase de recherche adaptée spécifiquement à la requête. L'estimation de difficulté peut aussi se placer plus simplement à la fin du processus et fournir des informations supplémentaires à l'utilisateur sans intervenir directement dans la recherche d'information.

Une indication de l'importance d'un traitement spécifique pour les requêtes particulièrement difficiles est donnée par l'introduction de la tâche "robust" ([Voorhees, 2003]) dans les campagnes TREC 2003. Cette tâche cherche à concentrer les efforts sur les requêtes pour lesquelles les systèmes de recherche documentaire traditionnellement obtiennent de faibles performances, et introduit pour cela des mesures nouvelles de performance globale, ainsi qu'une collection de requêtes dont une partie sont connues pour être difficiles à traiter.

Nous nous sommes intéressés essentiellement à trois scénarios fondamentaux :

- la prédiction de la difficulté d'une requête en termes de décision facile/difficile (chapitre 7)
- l'application sélective de techniques d'enrichissement des requêtes (en particulier le BRF) (chapitre 8)
- question/réponse (EQUER) : estimation de la qualité d'une réponse courte, sur un corpus français (chapitre 9)

Dans le chapitre 6 nous allons présenter les différentes problématiques avec les travaux publiés dans ce domaine.

6. État de l'Art

Sommaire

Bien que l'importance de l'analyse des requêtes ait été reconnue depuis quelques années, peu d'articles approfondis dans ce domaine semblent avoir étés publiés, malgré une certaine augmentation depuis 2002.

Les premiers articles traitant du sujet de la difficulté de requêtes abordent la question de manière assez générale, s'intéressant plus à l'évaluation de corpus de requêtes qu'à la prédiction de difficulté pour des requêtes individuelles. Quelques observations préliminaires sur la corrélation de certaines caractéristiques des requêtes avec les résultats obtenus par des systèmes de recherche documentaire sont mentionnées, mais le sujet n'est pas approfondi. La section 6.1 présente certains de ces travaux.

Plus récemment, différentes équipes ont cherché à développer des fonctions de prédiction de difficulté. Il s'agit là de trouver une fonction mathématique basée sur des propriétés de la requête qui serait fortement corrélée avec une mesure de difficulté (par exemple la précision moyenne obtenue). Ces travaux sont présentés dans la section 6.2.

Finalement, la section 6.3 rassemble les travaux s'intéressant à l'application des prédictions, en particulier pour des décisions sur le traitement de requêtes. Des travaux sur l'intégration et la combinaison de prédicteurs multiples se trouvent également dans cette partie.

6.1. Études générales ou préliminaires

Dans TREC-5, [Voorhees et Harman, 1996] notent de fortes différences entre les requêtes courtes (titres seulement) et les requêtes entières qui donnent de bien meilleurs résultats. Par rapport à TREC-4, les résultats obtenus sur les requêtes courtes ont déjà fortement évolué et certains systèmes optimisés pour cette tâche obtiennent de très bonnes performances.

Dans ce même article, une mesure de difficulté des requêtes est introduite[1], car une forte

[1] réintroduite, pour être exact – elle avait été utilisée en TREC-2 à titre expérimental

chute des performances est visible entre TREC-4 et TREC-5. De premières investigations pour trouver des facteurs déterminant la difficulté ont montré qu'il n'y avait pas de corrélation significative avec la longueur des requêtes, ni le nombre de documents pertinents. L'article conclut que les requêtes sont plus complexes dans un sens général du terme et qu'il est difficile de prédire la difficulté d'une requête. Il est suggéré que plus de recherches devront être menées dans ce domaine.

Dans TREC-6, il est noté que des requêtes très courtes (titres uniquement) contenant des mots clés bien choisis peuvent donner de bons résultats, alors que l'absence de ces mots (requêtes sans les titres) détériore les résultats [Voorhees et Harman, 1997]. Il est suggéré que selon la longueur de la requête la stratégie de recherche devra être adaptée.

Un article publié en 1998 par Bank, Over et Zhang traite le problème de classification des requêtes par rapport aux performances obtenues par différents systèmes de recherche utilisant six méthodes statistiques [Banks *et al.*, 1998]. Les résultats ne sont pas satisfaisant, comme il est noté dans l'article : "None of the work we have done using the six approaches discussed here has provided the sort of insights we were seeking [...]".

Dans le cadre de TREC-8, Karen Spärck Jones note une forte corrélation entre la performance des systèmes de recherche et la qualité d'information sur la requête ainsi que la "difficulté de la requête" (dans un sens général, non technique) [Spärck Jones, 1999]. Par contre, d'après les résultats de TREC-7 et TREC-8, les résultats n'utilisant que le titre sont pratiquement aussi bons qu'avec les descriptions longues.

Également suite à TREC-8, Rorvig évalue la difficulté des ensembles de requêtes des différentes années [Rorvig, 1999]. Il réussit à établir des critères permettant des prédictions sur la difficulté de l'ensemble. Par contre, les mesures ne permettent pas d'estimations pour des requêtes individuelles. Rorvig confirme encore que les paramètres simples, comme la longueur de la requête, ne permettent pas de faire d'estimations :

> "Factors that cannot describe query difficulty are : (1) topic components (concepts, narratives, etc.), (2) topic length, (3) and topic construction (creating topics without regard to existing documents vs. the contrary practice). Document uniqueness is the only quantitative measure so far offered. Indeed, topic hardness appears to rest in that zone of phenomena that many can mutually observe, but cannot describe in terms that would eventually permit control."

6.2. À la recherche d'une fonction de prédiction

Un article de Claude de Loupy et Patrice Bellot dans le cadre de LREC 2000 étudie quelques paramètres permettant des estimations de difficulté pour des requêtes individuelles [de Loupy et Bellot, 2000]. Une continuation de ce travail ce trouve dans la thèse de doctorat de Claude de Loupy [de Loupy, 2000]. Certaines des ces mesures sont décrites

en plus de détail dans la section 7.1, en particulier des scores basés sur le nombre de sens ou la synonymie des termes de la requête. De Loupy constate une corrélation entre quelques-unes des mesures et la précision moyenne obtenue, utilisant les données de TREC 6, mais cette corrélation n'est pas quantifiée de manière exacte.

Une des premières tentatives ayant eu du succès semble être un score décrivant la clarté (*clarity*), c.à.d. l'absence d'ambiguité d'une requête, qui a été développé à l'Université du Massachusetts [Cronen-Townsend *et al.*, 2002]. *Clarity* est basé sur le calcul de l'entropie relative des documents retrouvés pour la requête par rapport au modèle de langage de la collection de documents entière et est supposé refléter la cohérence thématique des documents contenant les termes de la requête. On trouve dans la figure 6.1 le modèle de langage associé à la requête "Show me any predictions for changes in the prime lending rate and any changes made in the prime lending rates", en comparaison avec le modèle de langage de la collection (les 50 premiers termes).

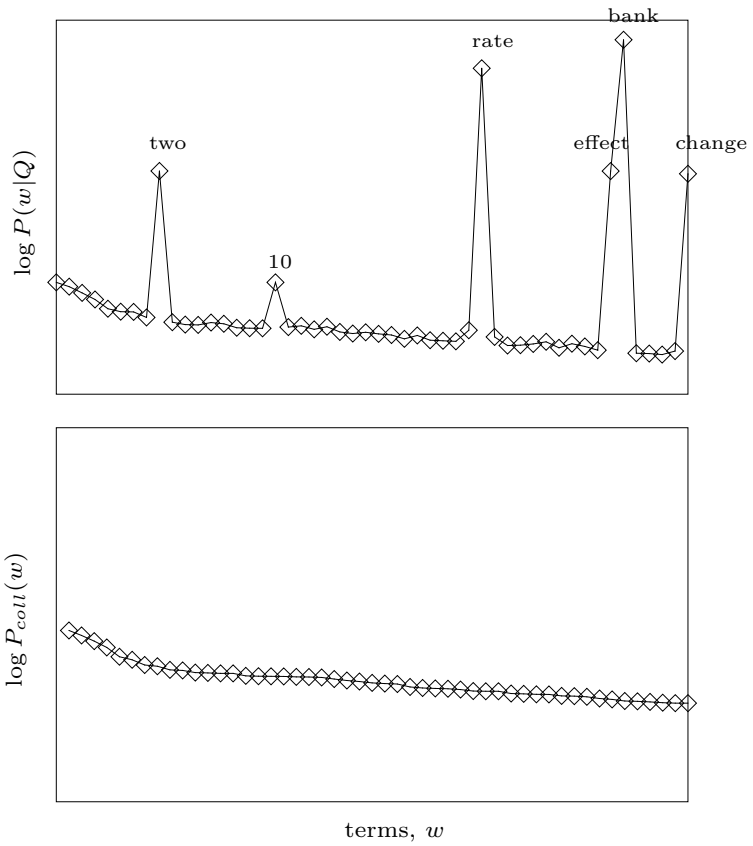

FIG. 6.1.: Exemple de modèle de langage utilisé pour le calcul du "clarity score"

Connaissant le modèle de langage pour la requête ($P(w|Q)$) ainsi que le collection entière ($P_{coll}(w)$), on obtient la clarté comme divergence Kullback-Leibler entre les deux modèles :

Collection	Queries	Num.	R	P-value
AP88+89	$101-200$	100	0.368	1.2×10^{-4}
TREC-4	$201-250$	50	0.490	3.0×10^{-4}
TREC-5	$251-300$	50	0.459	6.5×10^{-4}
TREC-7	$351-400$	50	0.577	2.7×10^{-5}
TREC-8	$401-450$	50	0.494	2.7×10^{-4}
TREC-7+8	$351-450$	100	0.536	4.8×10^{-8}

TAB. 6.1.: corrélation de rangs entre *clarity* et précision moyenne

$$\text{clarity} = \sum_{w \in V} P(w|Q) \log_2 \frac{P(w|Q)}{P_{coll}(w)}$$

Cette mesure est dans l'esprit assez proche des mesures que nous avons développées basées sur l'entropie ou la similarité sur l'ensemble des document retourné par le système de recherche documentaire pour une requête donnée (voir section 7.1.4).

L'efficacité du score de clarté pour l'estimation de difficulté de requêtes a été évaluée en termes de corrélation de rangs de Spearman avec la précision moyenne obtenue, sur différentes collections de test. Les résultats sont résumés dans la figure 6.1.

Amati, Carpineto et Romano proposent une mesure appelée $INFO_{DFR}$ qui est assez fortement corrélée avec la précision moyenne [Amati *et al.*, 2004], basée sur la distribution des termes de la requête dans les documents classés premiers par le système de recherche documentaire. Ils proposent également une mesure utilisée pour prédire l'effet d'expansion de requête ($INFOQ$). Leurs résultats montrent de légères amélioration pour certaines requêtes et un effet moins évident sur l'ensemble des requêtes (gain relatif d'environ 2%).

Parmi les publications les plus récentes se trouve [He et Ounis, 2004], qui présente plusieurs prédicteurs de performance des requêtes qui peuvent être calculés avant la phase de recherche de documents. Leurs corrélation avec la précision moyenne obtenue est évaluée pour les collections de TREC 7 et 8. Contrairement aux mesures présentées par Cronen-Townsend *et al.* ou Amati *et al.*, ils se limitent à des informations qui peuvent être obtenues de manière peu coûteuse avant de lancer le processus de recherche de documents.

Les mesures comparées dans cette étude incluent la longueur de la requête (sans mots vides), la distribution informative des termes de la requête (σ_{idf} ainsi que $\frac{idf_{max}}{idf_{min}}$), une version simplifiée du *clarity score*, ainsi que la proportion de documents dans la collection qui contiennent au moins un terme de la requête (*query scope* $\omega = -\log(\frac{n_Q}{N})$). L'évaluation a été effectuée par rapport à la corrélation linéaire entre ces mesures et la précision moyenne obtenue par deux systèmes de recherche (basés sur PL2 ainsi que BM25).

Josiane Mothe et Ludovic Tanguy de l'IRIT ont analysé 16 caractéristiques linguistiques des requêtes, recherchant des corrélations significatives avec le rappel et la précision obtenus par différent systèmes de recherche documentaire ayant participé à TREC

[Mothe et Tanguy, 2005]. Chacune de ces caractéristiques décrit une propriété linguistique spécifique de la requête, morphologique, syntactique ou sémantique. Deux de ces attributs (la distance de liens syntactiques ainsi que la polysémie) montrent un impact significatif sur rappel et précision obtenus. Bien que les valeurs de corrélation ne soient pas très fortes, elles indiquent un lien prometteur entre certaines propriétés linguistiques et la difficulté de requêtes. L'évaluation a été effectuée à l'aide de la corrélation de Pearson des attributs avec la précision obtenue. Des corrélations significatives (p-value < .05) ont été constatées sur certaines de collections de test utilisées, en particulier une corrélation de -0.396 (p-value < 0.001) de la distance syntactique (SYNTDIST) sur la collection TREC 5.

Craig Macdonald, Ben He et Iadh Ounis se concentrent sur la prédiction de difficulté pour des recherches intranet [Macdonald *et al.*, 2005]. Ils ont évalué deux prédicteurs, l'un basé sur l'IDF moyenne des termes de la requête, l'autre qui tient compte du nombre de documents dans la collection contenant au moins un terme de la requête par rapport au nombre total de documents. Ces prédicteurs se sont avérés très efficaces pour des requêtes d'un ou deux termes, mais la qualité de prédiction est moindre pour des requêtes plus longues.

6.3. Méthodes avancées et applications

Sehgal et Srinivasan s'intéressent à des applications dans le contexte particulier des requêtes sur les gènes (humains) dans des collections spécialisées (LocusLink, PubMed) [Sehgal et Srinivasan, 2005]. Comme dans le contexte de la recherche documentaire généraliste, ils constatent que certaines stratégies visant à améliorer les résultats dégradent la qualité de la réponse pour un certain nombre de requêtes (voir figure 6.2).

Cet effet est directement lié à la qualité du résultat initial. Une estimation de la qualité de celui-ci permettrait donc d'appliquer les méthodes supplémentaires de manière ciblée et d'éviter une dégradation éventuelle des réponses.

Pour prédire la précision moyenne, Sehgal *et al.* utilisent un modèle de régression basé sur 12 attributs, qui tient compte de la distribution des termes de la requête dans la collection, de la spécificité des termes dans les documents trouvés, ou encore de l'ambiguïté des termes de la requête. Leur méthode montre une corrélation significative avec la précision moyenne et réduit d'environ 35% l'erreur de prédiction par rapport à la *baseline* (prédiction uniforme de la moyenne des précisions obtenues sur le corpus d'apprentissage). L'impact de l'application de ces prédictions pour une application sélective des stratégies de reclassement et de modification de requêtes n'est pas évalué.

Dans le cadre de TREC 12, Amitay, Carmel et Darlow ont présenté une méthode appelée *"Query-Sensitive Tuning"* (QUEST), qui sert à adapter le classement des documents selon le type de requête ([Amitay *et al.*, 2003]).

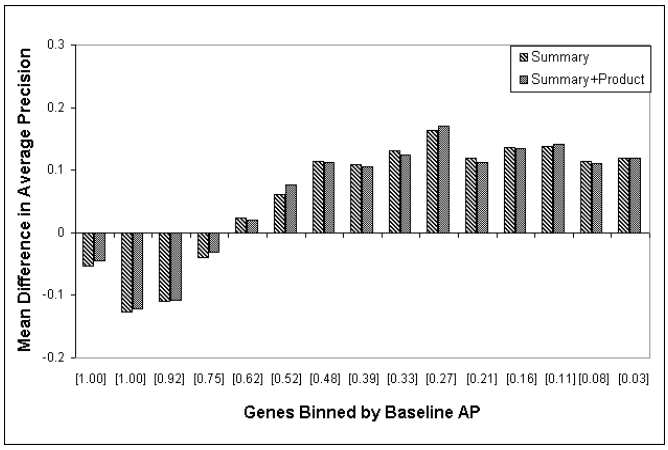

FIG. 6.2.: Effet de méthodes de reclassement par rapport à la précision moyenne initiale (AP)

Elad Yom-Tov, Shai Fine, David Carmel et Adam Darlow constatent que les requêtes donnant des bons résultats sont généralement celles dont les termes sont en accord sur les documents trouvés [Yom-Tov *et al.*, 2005a]. Dans le cas où les documents retournés pour les sous-requêtes (consistant en un terme de la requête) sont très différents, ou que certains termes de la requête fournissent des documents très divergeants du reste des termes, la qualité du résultat est généralement assez faible.

Le recoupement entre les listes de documents des différentes sous-requêtes et de la requête complète est défini comme étant le nombre de documents appartenant aux deux listes (limitées aux N premiers documents). Une mesure de recoupement entre les listes de documents obtenues pour les sous-requêtes est ensuite utilisé pour la construction d'un classifieur automatique de type arbre de décision (modifié) ou sous forme d'histogramme du nombre de recoupements.

Ces mesures montrent une corrélation fortement significative avec la précision moyenne (*AP*), ainsi qu'avec la précision sur les 10 premiers documents (*P@10*). Ces corrélations sont considérablement plus fortes que celles obtenues par différentes autres mesures proposées dans la littérature.

Ils présentent trois applications de leur méthode : l'amélioration des performances de recherche documentaire, l'identification des requêtes n'ayant pas de documents pertinents dans la collection, ainsi que la combinaison des résultats dans un scénario de recherche distribuée.

La première application est très proche des travaux présentés dans cette thèse. En effet, Yom-Tov et al. proposent d'appliquer de manière sélective l'expansion de requêtes selon les prédictions faites à l'aide de SVM sur la base de l'histogramme des recoupements.

Dans le cas où les requêtes sont disponibles sous forme d'un titre ainsi qu'une description, ils proposent de se baser sur le titre uniquement pour les requêtes (estimées comme étant) les plus difficiles, et d'utiliser la description pour les autres. D'autre part ils évaluent l'influence d'une adaptation de certains paramètres de leur système de recherche documentaire (Juru) selon l'estimation de difficulté de la requête.

Le tableau 6.3 montre les améliorations de performance obtenues par l'application sélective de méthodes d'expansion automatique de la requête (AQE), ainsi que l'utilisation des différentes formes de la requête.

Run name	MAP	P@10	%no
Description only	0.281	0.473	9.5
Description with AQE	0.284	0.467	11.5
Description with selective AQE	0.285	0.478	9.5
Description with modified parameters	0.282	0.467	9.5
Title only	0.271	0.437	10.0
Title + Description	0.294	0.484	8.5
Switch title-description	0.295	0.492	6.0

FIG. 6.3.: Yom-Tov *et al.* : application sélective d'expansion de requête (AQE)

Nous allons revenir sur leurs résultats dans les sections concernant les problématiques traitées, surtout en ce qui concerne l'application sélective d'expansion de requêtes (dans le chapitre 8).

Ils ont également utilisé leur méthode de prédiction de difficulté pour la fédération de plusieurs sources, par exemple pour la combinaison des résultats trouvés sur différentes collections de documents ou par différents systèmes de recherche [Yom-Tov *et al.*, 2005b]. L'estimation de difficulté est utilisée pour pondérer les scores ou rang des documents issus de chacune des sources.

K.L. Kwok de la City University de New York a traité le problème de l'identification des requêtes les plus faibles et les plus fortes [Kwok, 2005]. Il a montré qu'une combinaison d'attributs de la requête, dont l'IDF (*inverse document frequency*) des termes et la distribution des moyennes des fréquences des termes, permettait de prédire correctement environ 1/3 à 1/2 des 6 requêtes les plus fortes et plus faibles parmi 50 requêtes. Il propose d'utiliser le web pour l'expansion de requêtes difficiles, mais constate que les résultats de la prédiction ne sont pas encore suffisamment fiables pour améliorer la recherche ad-hoc.

Un certain nombre des travaux ci-dessus ont été présentés dans le cadre du workshop intitulé "Predicting query difficulty – methods and applications" qui réunissait lors de la conférence SIGIR 2005 de nombreux chercheurs du domaine.

Un autre évènement important pour la recherche sur les requêtes difficiles et l'adaptation

des systèmes de recherche documentaire à ces requêtes était un workshop de 6 semaines organisé par le NIST en 2003, sur le thème du "Reliable Information Access (RIA)", l'accès fiable aux informations. Les résultats ont été présentés à SIGIR 2004 et sont résumés dans [Harman et Buckley, 2004].

L'objectif du workshop RIA était l'analyse précise des échecs de systèmes de recherche, afin d'en déterminer les causes et proposer des solutions au problèmes trouvés. D'autre part, une série importante d'expérimentations a été conduite au sujet de l'expansion de requête, dans le but de comparer les approches proposées par différentes équipes et implémentées dans différents systèmes. Ainsi, pour les techniques de retour de pertinence en aveugle (blind relevance feedback), l'effet du nombre de documents pris en compte, du nombre de termes ajoutés à la requêtes, et surtout de la manière de choisir ces termes, a été analysé en détail, et comparé entre les différents systèmes participants.

Le workshop RIA fournit pour les recherches futures une collection de données énorme (plus 40Go) avec, en plus de l'analyse d'échec manuelle de 45 requêtes, les résultats de douzaines de configurations de systèmes différentes sur 9 collections de test différentes.

7. Notre approche

Sommaire

Une différence fondamentale entre d'autres approches pour la prédiction de performance et celle que nous proposons est l'utilisation d'une multitude de mesures afin d'exploiter leur combinaison. Nous regardons la prédiction de performance comme un problème de classification, où un ensemble d'attributs sont utilisés pour assigner une étiquette ou valeur de confiance à une requête.

Une fois un ensemble d'attributs potentiels déterminé, des algorithmes d'apprentissage automatique sont utilisés pour entraîner des classifieurs pour la tâche de prédiction. Des arbres de décision (*Decision Trees, DT*), des machines à vecteur support (*Support Vector Machines, SVM*) ainsi que d'autres méthodes ont été utilisés. L'entraînement a été effectué sur des corpus de données connues, issus des résultats publiés suite aux évaluations TREC.

Nous avons ainsi constitué un ensemble d'attributs aidant à prédire la qualité du résultat fourni par un système de recherche donné, sur une collection de documents donnée, pour une requête spécifique. Certains de ces attributs sont calculés sur la base de la requête seule, ou (en fonction de l'information disponible)

- la liste de documents trouvés
- des connaissances sur le processus de recherche (les scores utilisés pour le classement des documents, etc.).

Ce chapitre traite les expériences faites sur la tâche adhoc de TREC, et les résultats donnés sont essentiellement ceux obtenus sur TREC 8.

Les chapitres suivant montreront d'autres applications et leurs spécificités, aussi bien au niveau des attributs utilisés, des méthodes de classification, ou même de l'objectif de la classification.

7.1. Les attributs

7.1.1. Constatations générales

En considérant les requêtes les plus faciles et les plus difficiles, on remarque que dans les deux groupes les requêtes sont très variées. Comme le montrent aussi les analyses plus détaillées de certains paramètres, il n'y a pas de caractéristique qui à elle seule permette de faire des estimations sur la difficulté des requêtes.

On constatera également que la difficulté en termes de performance de recherche dépend surtout des attentes de l'utilisateur, du *besoin en information*, qui ne se reflètent pas toujours directement dans les requêtes mais sont à la base de l'évaluation des résultats livrés par le système de recherche. Dans le cas des campagnes TREC, cet aspect est visible dans le narratif fourni avec les requêtes et dans certains cas utilisé également pour la recherche.

Le narratif est difficile à exploiter de manière automatique car il ne fournit généralement pas de mots clés utile à la recherche. Il représente habituellement un approfondissement

de la requête et permet surtout d'identifier les documents *non* souhaités par l'utilisateur.

Malgré ces problèmes fondamentaux et probablement insolubles de manière automatique, nous avons pu trouver un certain nombre d'indicateurs permettant de faire des estimations sur la qualité du résultat fourni par un système de recherche documentaire.

7.1.2. La qualité d'un indicateur de performance

Dans un premier temps, il est nécessaire de quantifier la qualité d'un prédicteur de la précision moyenne obtenue pour une requête.

De manière générale, il est relativement difficile de quantifier la qualité d'un score x en tant que prédicteur d'un attribut y (dans notre cas de la précision moyenne obtenue pour une requête). Il y a une variété de mesures, allant de la simple corrélation linéaire à des méthodes statistiques plus complexes.

Nous avons utilisé essentiellement deux mesures que nous présenterons plus bas :

- l'impureté (ou surtout la réduction de l'impureté en séparant les données en classes) qui reflète assez intuitivement les résultats qu'on pourrait attendre en utilisant des algorithmes de classification et décision automatique.
- la corrélation de rangs de Spearman qui est un moyen reconnu de quantifier les dépendances entre différents attributs

7.1.2.1. Corrélation linéaire

Une des mesures les plus simples (mais clairement pas la meilleure) pour décrire le potentiel d'un paramètre comme prédicteur de performance de recherche est la corrélation linéaire entre le paramètre et la précision moyenne obtenue pour la requête.

$$r(x,y) = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{s_x s_y}$$

où s_x and s_y représentent l'écart type de x et y :

$$s_\alpha^2 = \frac{1}{n}\sum_{i=1}^{n}(\alpha_i - \overline{\alpha})^2$$

Nous avons utilisé cette mesure initialement comme possibilité simple de quantifier la qualité d'un paramètre, mais l'avons rapidement abandonnée en échange d'autres mesures qui nous semblent plus pertinemment représenter le potentiel de classification d'un paramètre.

7.1.2.2. L'impureté

Afin de trouver une mesure relativement simple et intuitive, nous avons choisi de calculer l'impureté des classes obtenues en séparant les exemples selon le score choisi. L'impureté est définie comme la somme des variances des précisions moyennes dans chaque classe, pondérée par le nombre d'exemples, normalisée par la variance globale des précisions moyennes :

$$Impurity = \frac{\sum_{c \in C} |c| \cdot \sigma_{AP_c}}{|Q| \cdot \sigma_{AP}}$$

avec Q l'ensemble des exemples/requêtes, C l'ensemble des sous-ensembles disjoints $c \subset Q$, tel que $\bigcup c \in C = Q$, et σ_{AP_c} la variance des précisions moyennes des éléments de c, σ_{AP} la variance des précisions moyennes sur l'ensemble Q.

Pour construire les classes nous avons décidé de procéder par succession de séparations binaires optimales. L'optimalité de cette séparation est définie comme l'obtention de la plus faible impureté. La séparation consiste à déterminer une seuil sur le score x résultant en deux classes : $x < limite$ et $x \geq limite$.

Chaque classe est ensuite à nouveau séparée jusqu'à obtention du nombre désiré de classes. À cause du faible nombre d'exemples (50 requêtes) pour chaque système, nous nous sommes contentés de quatre classes.

Il est clair que cette approche ne garantit pas l'optimalité du résultat, mais la détermination de la séparation optimale en deux classes est triviale, alors qu'une séparation n-classes optimale nécessite des algorithmes beaucoup plus lourds. Ceci ne nous semblait pas nécessaire dans ce contexte.

7.1.2.3. Le coefficient de corrélation de Spearman

Afin de comparer nos résultats à ceux publiés par d'autres chercheurs, en particulier dans [Cronen-Townsend *et al.*, 2002], nous avons décidé de calculer également le coefficient de corrélation de rangs de Spearman ρ. Il s'agit là d'une mesure de corrélation reconnue et couramment utilisée.

Le coefficient est défini ainsi :

$$\rho = 1 - \frac{6 \sum D^2}{N(N^2) - 1}$$

avec D la différence entre les rangs de valeurs correspondantes de X et Y, et N le nombre de paires de valeurs.

Nous avons également déterminé la *P-value*, c.à.d. la probabilité que la distribution des données soit celle que l'on constate dans l'hypothèse d'une absence de corrélation. Cette *P-value* est basée sur le score de corrélation et le nombre d'exemples (de requêtes) utilisés

pour le calculer : plus cette valeur est basse, plus on peut être certain qu'il existe réellement une corrélation.

Nous avons ainsi à notre disposition la plus "classique" *rank correlation* qui est un moyen reconnu pour quantifier les dépendances entre différents attributs, et l'impureté qui reflète assez intuitivement les résultats qu'on peut attendre dans le cadre de l'utilisation d'algorithmes de classification et décision automatique.

7.1.3. Attributs simples de la requête seule

7.1.3.1. Longueur de la requête

Confirmant les constatations mentionnées dans certains articles [Rorvig, 1999, Spärck Jones, 1999, Voorhees et Harman, 1997], les expérimentations n'ont pas montré de corrélation forte entre la longueur des requêtes et les performances des systèmes de recherche.

Ceci contredit des expérimentations faites préalablement entre autres par Lu et Keefer[Lu et Keefer, 1994], ce qui suggère que selon le système utilisé, la longueur des requêtes peut avoir un effet sur les performances, mais que ce n'est pas le cas sur l'ensemble des systèmes participants à TREC.

Plusieurs aspects rendent difficile de comparer différentes études sur l'impact de la longueur des requêtes sur la performance de recherche. Les systèmes de recherche documentaire ont évolué au fil des années, et ne réagissent pas de la même manière à différentes formulations des requêtes. En particulier, les systèmes plus récents sont bien plus performants sur des requêtes courtes (éventuellement dû à des meilleures techniques d'enrichissement des requêtes). De plus, les collections de documents et de requêtes de TREC ont changé, et les requêtes n'ont pas toujours été créées de la même manière et avec la même structure.

Enfin, certaines études ont comparé différentes versions des requêtes, parfois créées artificiellement en raccourcissant des requêtes, alors que d'autres comparent des requêtes qui ont "naturellement" une longueur différente. Alors que dans le premier cas on se trouve confronté à une perte d'information, dans le second cas le nombre plus faible de mots clés utilisables dans les requêtes plus courtes peut souvent être compensé par le fait qu'il s'agit en effet de requêtes plus facilement exprimables en peu de mots.

7.1.3.2. Nombre de sens des mots de la requête

Ce paramètre a été utilisé avec beaucoup de succès pour l'estimation de difficulté de requêtes dans [de Loupy et Bellot, 2000, de Loupy, 2000].

Par contre, ayant initialement travaillé indépendemment de ces résultats, nous n'avions pas constaté une telle corrélation (pour le système utilisé). En approfondissant les investi-

gations, il s'est montré que les résultats obtenus par de Loupy et Bellot étaient très fortement liés d'une part à la base de requêtes traitée (ils avaient travaillé comme nous sur des données de TREC, mais en n'utilisant que la version très courte des requêtes de TREC 6) et d'autres part de leur manière particulière d'attribuer un score en fonction du nombre de sens des mots.

La méthode proposée par Claude de Loupy et Patrice Bellot est la suivante :

> "Le score de la requête est incrémenté de 1 pour chaque terme ayant un seul sens (un mot inconnu de WordNet est considéré comme ayant un seul sens). Si tous les termes ont un seul sens, le score est de 3. Si tous les termes ont 3 sens ou plus, le score est de -1 et si un mot n'est pas présent dans la base, le score est de -2."[1]

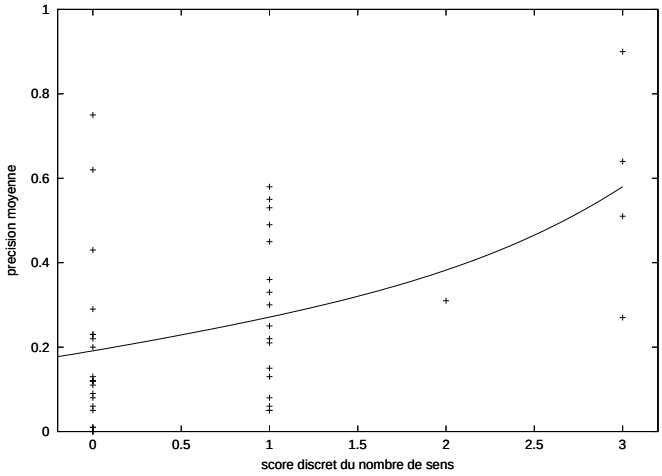

FIG. 7.1.: Score discret du nombre de sens – requêtes courtes de TREC 6

Il est évident qu'une telle manière d'attribuer un score est fortement liée à la condition que les requêtes soient très courtes (1 à 4 mots dans ce cas). Dans cette situation on peut supposer qu'il s'agit de mots clés significatifs pour la recherche, ce qui n'est pas le cas pour des requêtes plus longues, contenant un narratif en langage naturel.

En effet, nous avons pu reproduire les résultats en utilisant (pratiquement) la même méthode sur la même base de requêtes (avec une corrélation linéaire de 0.522), mais il ne semble pas qu'il y ait de moyen direct d'appliquer ce score de manière utile à d'autres types de requêtes (corrélation linéaire de 0.122 sur l'ensemble des requêtes, en incluant les requêtes longues).

Un résultat plus étonnant est que de légères variations dans la manière de calculer le

[1] De plus, ils catégorisent les requêtes en *faciles*, *moyennes* et *difficiles*, mais cela n'a pas d'importance dans ce contexte.

score (p.ex. de soustraire des points pour des mots ayant beaucoups de sens) diminuent dramatiquement la corrélation, ce qui montre une fois de plus la difficulté d'établir des paramètres utilisables.

Nous avons également pu constater que les résultats sont beaucoup moins nets sur des requêtes de construction semblable de TREC 7 et 8 (la corrélation linéaire est de 0.185 sur TREC 8, 0.158 sur TREC 7). Il semble donc que le score utilisé est très spécifiquement optimisé sur un certain corpus de requêtes. Étant donné qu'on note tout de même une relation entre ce score et la difficulté des requêtes (d'ailleurs plus sur TREC 8 que sur TREC 7), il reste utile d'inclure ce paramètre pour la classification.

Nous avons fait des tests en utilisant le nombre moyen, maximal et minimal de sens des mots contenus dans la requête ainsi que le score décrit ci-dessus, sur la base des résultats soumis par différentes équipes participant à TREC 8. On constate une certaine corrélation négative avec la précision moyenne obtenue (une corrélation positive dans le cas du score discret). Mais pour la majorité des systèmes celle-ci n'est pas fortement significative.

7.1.3.3. Nombre de synonymes des mots de la requête

Alors qu'il est assez évident qu'un mot n'ayant qu'un seul sens aura plus de chance de permettre de bons résultats de recherche par rapport à des mots plus ambigus, l'effet que peuvent avoir les synonymes est moins clair. Un système ancien aura certainement des problèmes de rappel s'il ne trouve pas les documents pertinents contenant des synonymes du mot de la requête, mais les systèmes actuels utilisent souvent des méthodes d'expansion de requêtes comme le *blind relevance feedback* [Walker et de Vere, 1990], permettant de rajouter des mots sémantiquement proches des termes de la requête (synonymes, hyponymes et autres).

Nous avons constaté que pratiquement tous les systèmes ayant participé à TREC 8 montrent une corrélation négative avec le nombre moyen de synonymes des mots de la requête, mais celle-ci n'est significative que pour quelques rares systèmes.

7.1.3.4. Spécificité

La spécificité des mots de la requête est un indicateur important pour l'estimation de difficulté. Plus les termes sont spécifiques, plus des documents contenant ces termes ont de chances de répondre aux besoins de l'utilisateur, alors que des termes très génériques ne pourront souvent pas suffisamment déterminer les documents pertinents.

Différentes mesures peuvent permettre de donner une indication de la spécificité d'une requête. Il faut cependant noter que les notions de spécificité et de généricité sont très relatives à la thématique documentaire.

7.1.3.4.1. Nombre d'hyponymes des mots de la requête Une de ces mesures est le nombre d'hyponymes, c'est à dire de termes plus spécifiques englobés par le mot donné. On peut raisonnablement supposer que plus un mot a d'hyponymes, moins il est spécifique. Ces propriétés ont d'ailleurs été prises en compte entre autre dans [Jourlin *et al.*, 2000].

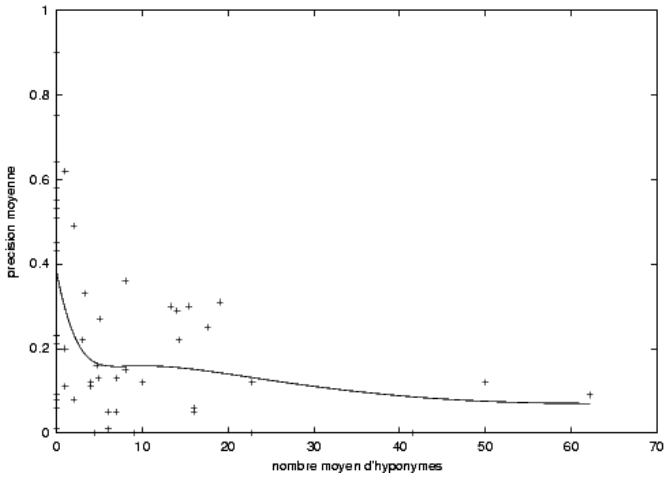

FIG. 7.2.: Nombre moyen d'hyponymes – requêtes courtes de TREC 6

Comme beaucoup d'autres mesures, le nombre d'hyponymes ne semble être utilisable que pour des requêtes très courtes, alors que sur l'ensemble de toutes les requêtes il n'y a pas de corrélation visible avec la difficulté des requêtes (voir figure 7.2).

7.1.3.4.2. IDF Une mesure très utilisée dans la recherche documentaire pour estimer l'apport d'information d'un mot est l'IDF (*inverse document frequency*) :

$$IDF(\lambda) = -\log\left(\frac{documents\ contenant\ x}{nombre\ total\ de\ documents}\right)$$

Les scores simples dérivés des IDF des termes de la requête (IDF moyenne, maximale, etc.) ne montrent à eux seuls pas de très forte corrélation avec la précision moyenne obtenue.

Néanmoins, en combinaison avec d'autres mesures, l'IDF peut servir à développer de bons scores pour la prédiction de difficulté.

7.1.3.4.3. Fréquence des termes (score discret) Claude de Loupy et Patrice Bellot ont montré que la fréquence des termes peut être un bon indicateur de la difficulté d'une requête, du moins pour les requêtes courtes. Ils proposent de calculer le score suivant (cita-

tion raccourcie) :

> "Un mot apparaissant dans moins de n documents (nous avons choisi, de façon empirique, $n = 1000$) est considéré comme *un point de fixation* pour la réponse à la requête[...]
>
> Pour chaque mot de la requête apparaissant moins de n fois, le score est incrémenté de 1. Si tous les mots de la requête sont dans ce cas, le score de la requête est de 3 qui représente le score maximal que peut obtenir une requête. Si l'un des mots ne se trouve pas dans la base, la requête se voit affecter arbitrairement le score minimal de -2. Dans le cas où tous les mots apparaissent plus de 10000 fois dans la base, le score est égal à -1 (valeur arbitraire). Dans tous les autres cas, le score est de 0."

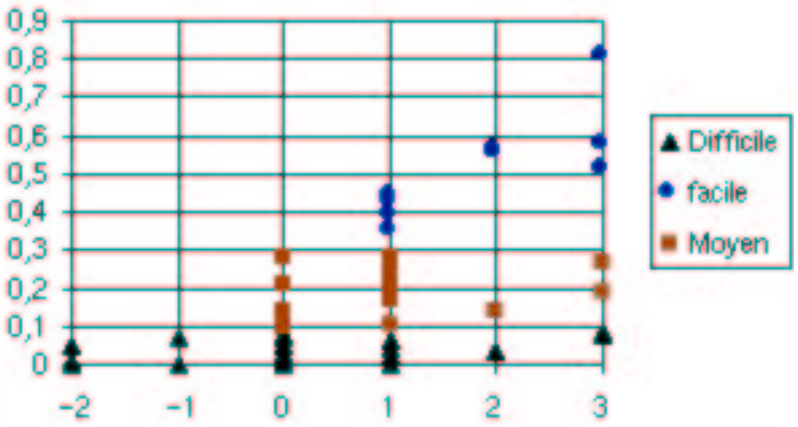

FIG. 7.3.: Score discret utilisant la fréquence des termes – requêtes courtes de TREC 6

La figure 7.3 présente les résultats obtenus par de Loupy et Bellot.

Comme le score discret pour le nombre de sens, ce score ne peut pas être appliqué directement à des requêtes plus longues, et ne semble malheureusement pas très robuste par rapport à d'autres collections de requêtes ou documents. En particulier, les constantes choisies empiriquement devront être adaptées car elles dépendent fortement du nombre de documents dans la collection.

7.1.3.4.4. Coefficient de *Bookstein* Une autre mesure de la spécificité d'un terme, le coefficient de Bookstein $B(\lambda)$, est défini ainsi :

$$B(\lambda) = \frac{N(\lambda)}{E(\lambda)}$$

où $N(\lambda)$ est le nombre de documents contenant λ et $E(\lambda)$ le nombre théorique qui de-

vraient contenir λ si la répartition était de type aléatoire.

$$E(\lambda) = D \cdot \left(1 - \left(1 - \frac{1}{D}\right)^{T(\lambda)}\right)$$

où D est le nombre total de documents dans la base et $T(\lambda)$ le nombre total d'occurrences de λ dans la base (compté autant de fois qu'il apparaît dans un document pour tous les documents).

Le coefficient de Bookstein n'a pas été étudié individuellement, mais est utilisé dans le score 7.1.3.5.

7.1.3.5. Score de *de Loupy & Bellot*

Claude de Loupy et Patrice Bellot ont proposé des scores combinant plusieurs paramètres :

$$score1(Q) = \max_{x \in Q} \left(\frac{IDF(x)}{S(x)}\right)$$

où $S(x)$ et le score sur le nombre de sens présenté dans 7.1.3.2,

$$score2(Q) = \max_{x \in Q} \left(\frac{IDF(x)}{B(x) \cdot S(x)}\right)$$

où $B(x)$ est le coefficient de Bookstein.

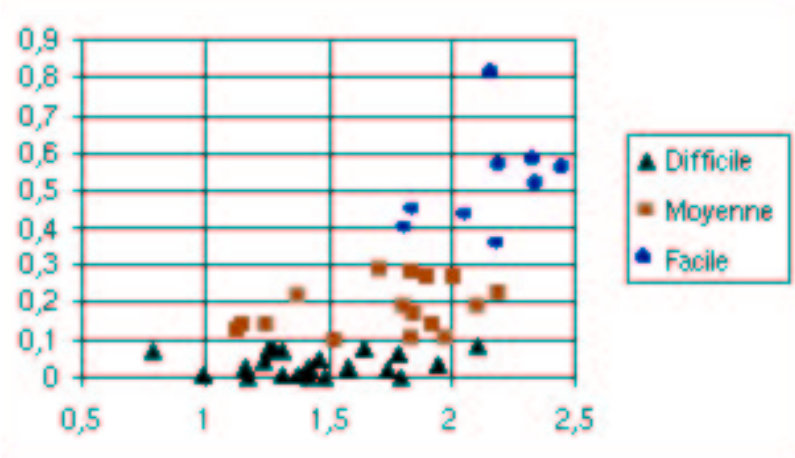

FIG. 7.4.: Score basé sur IDF, nombre de sens et coefficient de Bookstein ($score2(Q)$) – requêtes courtes de TREC 6

Ces scores (surtout la deuxième version) donnent de très bons résultats pour les requêtes

courtes issues de TREC 6, mais bien moindres sur d'autres collections.

7.1.3.6. Catégories de mots

Parmi les premiers résultats convainquants obtenus sur l'ensemble complet des requêtes dans ces expériences sont ceux basés sur l'utilisation de dictionnaires spécifiques pour certaines catégories de mots (présentées plus bas). Nous avons au début manuellement construit ces dictionnaires spécifiquement pour les requêtes traitées (issues de TREC 6).

Les performances des systèmes de recherche sont très nettement supérieures sur des requêtes contenant des mots de certaines catégories spécifiques par rapport au reste des requêtes, ce qui permet des prédictions assez bonnes sur un certain nombre de requêtes.

Pour les classes de mots analysées, seulement environ 10 à 15% des requêtes contiennent des mots d'une certaine catégorie, environ 30% contiennent des mots d'au moins une catégorie. En utilisant de bons dictionnaires spécialisés et incluant d'autres catégories on pourrait sans doute arriver à une couverture supérieure.

Étant donné que l'estimation de difficulté des requêtes devra se faire sur une combinaisons de plusieurs paramètres, il n'est pas absolument essentiel que chaque critère individuel utilisé recouvre un très grand pourcentage des requêtes. Les classes de mots peuvent donc être extrêmement utiles comme indicateurs positifs pour les requêtes concernées.

Nous avons essayé de remplacer les dictionnaires construits par nous-mêmes par des dictionnaires spécialisés disponibles en ligne, mais les résultats obtenus semblent moins clair. Il est à noter que des mesures comme la corrélation de rangs n'est pas bien adapté pour évaluer la qualité de ces indicateurs. C'est pour cela que nous présentons les résultats d'une manière différente des autres attributs. Les performances indiquées sont celles obtenues par notre système de recherche documentaire, décrit en plus de détail dans le chapitre 8. Vu le faible nombre d'exemples couverts par les catégories ci-dessous, nous ne présentons pas les résultats sur la collection TREC 8 uniquement, mais sur l'ensemble des années de TREC 6 à TREC 8.

7.1.3.6.1. Noms de pays Un très bon indicateur de bonnes performances pour une requête donnée est le nombre d'occurrences des *noms de pays* dans le termes de la requête. Sur le corpus analysé, 13% des requêtes contiennent de tels mots, et on constate des performances 22% supérieures au reste des requêtes (R-Prec = 0.55 par rapport à 0.45, précision moyenne = 0.27 par rapport à 0.22).

7.1.3.6.2. Noms de personnes Les occurrences de noms de personnes ne sont pas tout à fait aussi puissantes comme indicateur (performances environ 15% supérieures, avec une grande variance) et ne recouvrent qu'environ 5% des requêtes. On pourrait envisager (avec

des pertes de couverture additionnelles) de se limiter aux personnes célèbres, mais les tests faits jusqu'à présent ne tiennent pas compte de cet aspect.

7.1.3.6.3. Abréviations L'occurrence d'abréviations dans les requêtes est également assez fortement corrélée (positivement) avec les performances de recherche (R-Prec = 0.55 par rapport à 0.46, précision moyenne = 0.34 par rapport à 0.22) et distingue environ 9% des requêtes.

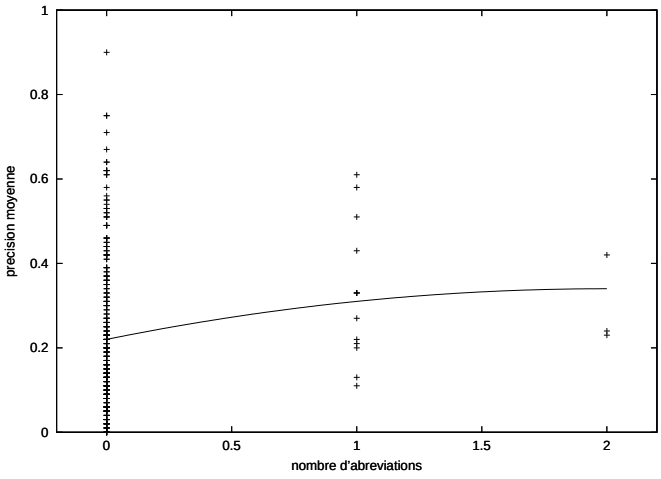

FIG. 7.5.: Occurrences d'abréviations – toutes requêtes, TREC 2 à 6

La grande variance des performances à l'intérieur des classes limite un peu la qualité des estimations sur cette base, mais l'occurrence d'abréviations est tout de même un indicateur très utile.

L'avantage de ce paramètre par rapport à d'autres classes de mots est qu'il est facile de reconnaître les abréviations automatiquement (sans dictionnaire) d'une manière assez sure.

7.1.3.7. Négations dans le narratif

Dans les campagnes TREC, les attentes de l'utilisateur sont le mieux exprimées dans le narratif, qui concrétise la demande formulée dans titre et description de la requête. Les expériences ont montré que l'apport du narratif à la recherche automatique est très limité, pourtant il permet à l'évaluateur humain de mieux déterminer la pertinence des documents trouvés.

De brèves expérimentations ont été faites pour essayer de trouver des critères formels qui décrivent l'influence que les attentes exprimées dans le narratif ont sur les performances, en particulier en observant l'occurrence de négations (*'not relevant'*, ...) dans le

narratif qui indiquerait que certains documents correspondant à la requête au niveau de titre et description ne sont, en fait, pas pertinent. Les résultats n'ont malheureusement pas été concluants, et nous avons abandonné cette voie.

7.1.4. Attributs basés sur l'ensemble des documents retournés

Prenant en compte les résultats d'une recherche initiale, il est possible d'ajouter des mesures plus sophistiquées qui dépendent de l'ensemble de documents retournés.

Une telle mesure est, par exemple, l'entropie des K premiers documents pour une requête, une autre est la similarité cosinus entre paires documents.

7.1.4.1. Entropie

On peut supposer qu'un bon résultat de recherche produira un ensemble de documents plus homogène. Par conséquent, l'entropie de l'ensemble des documents retournés (ou d'un sous-ensemble) devrait être plus élevé quand la performance de recherche pour une requête donnée est faible. Nous avons calculé différentes mesures d'entropie et observé leur corrélation avec la précision moyenne obtenue.

L'ensemble des K documents classés en tête pour une requête peut être considéré comme une source d'information. L'entropie de cette source peut être calculée utilisant un modèle de langage statistique (LM) de la manière proposée dans [Carpineto *et al.*, 2001]. On peut concevoir des LM à partir de n-grammes de mots. Il est évident que les probabilités d'unigrammes ($n = 1$) sont mieux estimées que pour $n > 1$.

Si l'entropie de l'ensemble est élevée la structure linguistique des documents est fortement variable. Nous supposons qu'une grande variabilité linguistique est liée à un plus grand risque que certains documents retournés ne soient pas pertinents. La probabilité que l'ensemble contienne des documents non-pertinents augmente avec K. Considérant qu'un seul document long et non-pertinent peut causer une augmentation significative de l'entropie, il semble préférable de se limiter à des valeurs de K relativement basses.

7.1.4.1.1. Entropie sur tous les unigrammes L'entropie dans le cas d'un modèle de langage est définie ainsi :

$$H = -\sum_{w \in W} P(w) \cdot \log P(w)$$

avec W l'ensemble de mots que nous traitons, et $P(w)$ la probabilité du mot w dans l'ensemble de documents respectif.

Nous avons d'abord décidé d'estimer cette probabilité comme suit :

$$P(w) = \frac{\sum_{d \in D} N_d(w) + \epsilon}{\sum_{v \in W} \sum_{d \in D} N_d(v) + |W|\epsilon}$$

avec D l'ensemble de documents sur lequel on calcule l'entropie, $N_d(w)$ le nombre d'occurrences de w dans d, et ϵ une constante servant à éviter le problème d'estimation à fréquence nulle.

Pour nos premiers tests nous avons choisi $W = V$, V étant le vocabulaire complet de notre collection de documents (sans mots vides, et après application du stemming de Porter). La corrélation entre entropie et performance de recherche est relativement faible (Figures 7.6, 7.7).

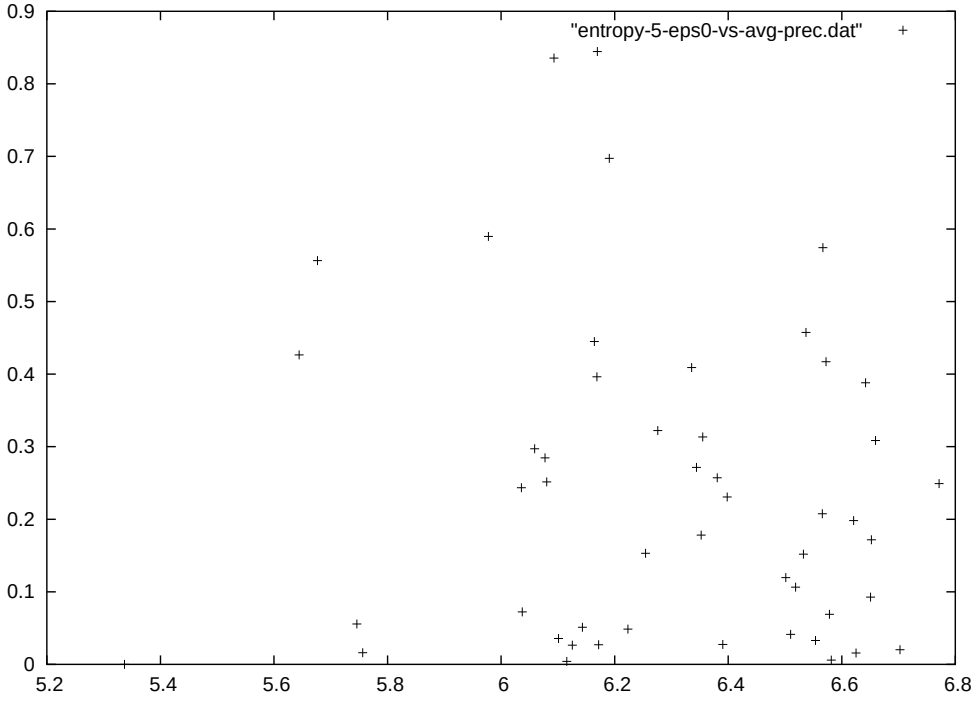

FIG. 7.6.: Entropie / précision moyenne : 5 documents, $\epsilon = 0$, tous les mots

7.1.4.1.2. Entropie sur une sélection d'unigrammes Afin de réduire le "bruit" dans notre mesure d'entropie, nous avons décidé de modifier l'estimation des probabilités de mots.

Plutôt que de calculer l'entropie sur le vocabulaire complet, nous avons choisi un nombre limité de mots. Pour sélectionner N mots, nous prenons les M mots les plus fréquents dans chaque document $d \in D$, produisant des ensembles $W_1 \ldots W_{|D|}$, avec M le plus petit entier tel que $|\bigcup W_i| > N$.

Nous définissons ensuite $W \subseteq \bigcup W_i$ comme contenant les mots les plus fréquents globa-

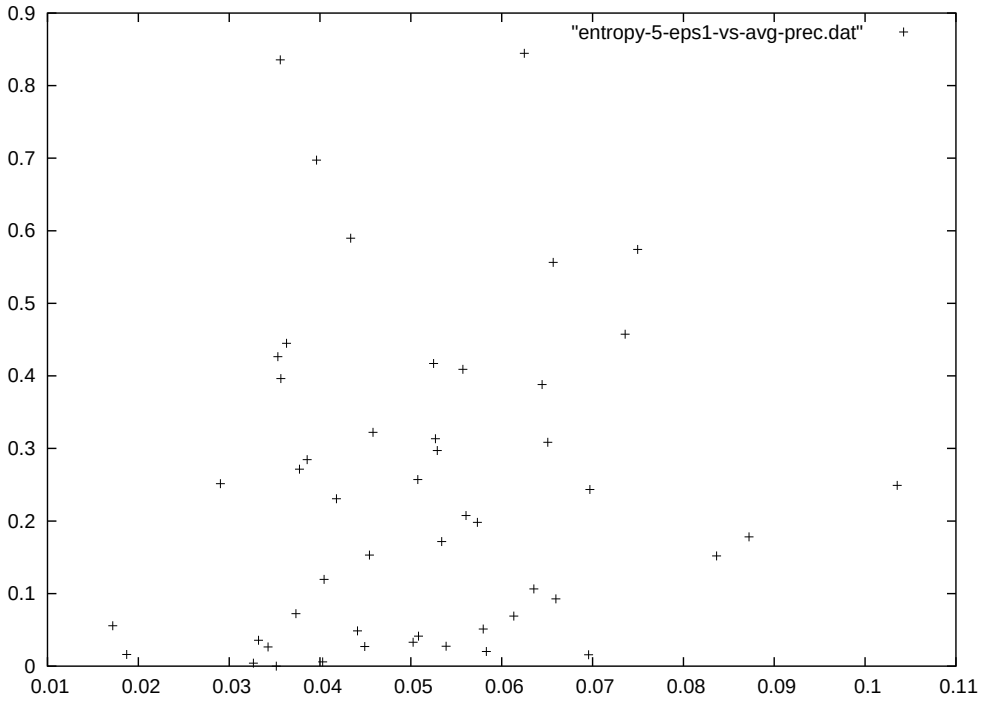

FIG. 7.7.: Entropie / précision moyenne : 5 documents, $\epsilon = 1$, tous les mots

lement issus de $\bigcup W_i$, tel que $|W| = N$.

On constate une certaine corrélation entre cette nouvelle mesure d'entropie et la performance de recherche, p.ex. avec $N = 10$ (Figure 7.8) ou $N = 100$ (Figure 7.9), en particulier pour certains exemples ayant une grande entropie et une précision du résultat de recherche très faible. Une courbe d'interpolation Béziers montre plus clairement la distribution des points dans le graphique.

En particulier sur ce dernier graphique, dans le cas d'une grande entropie, on trouve un nombre d'exemples qui, en accord avec notre hypothèse, obtiennent une faible précision moyenne (en bleu), mais également certains avec une précision relativement haute (en vert).

Nous avons étudié diverses variations du nombre de documents ainsi que du nombre de mots utilisés dans le calcul de l'entropie (Figures 7.10, 7.11). Aucun gain d'information visible n'est obtenu par le fait de se baser sur un nombre plus grand de documents ou d'utiliser un nombre de mots différent.

Nous avons comparé l'entropie à la précision sur les 5 premiers documents ($P(5)$, Fig. 7.12), afin de regarder uniquement la pertinence des documents réellement utilisés pour le calcul de l'entropie, au lieu de l'ensemble (beaucoup plus grand) de documents sur lequel se base la précision moyenne. Ceci montre que malgré la grande entropie des premiers

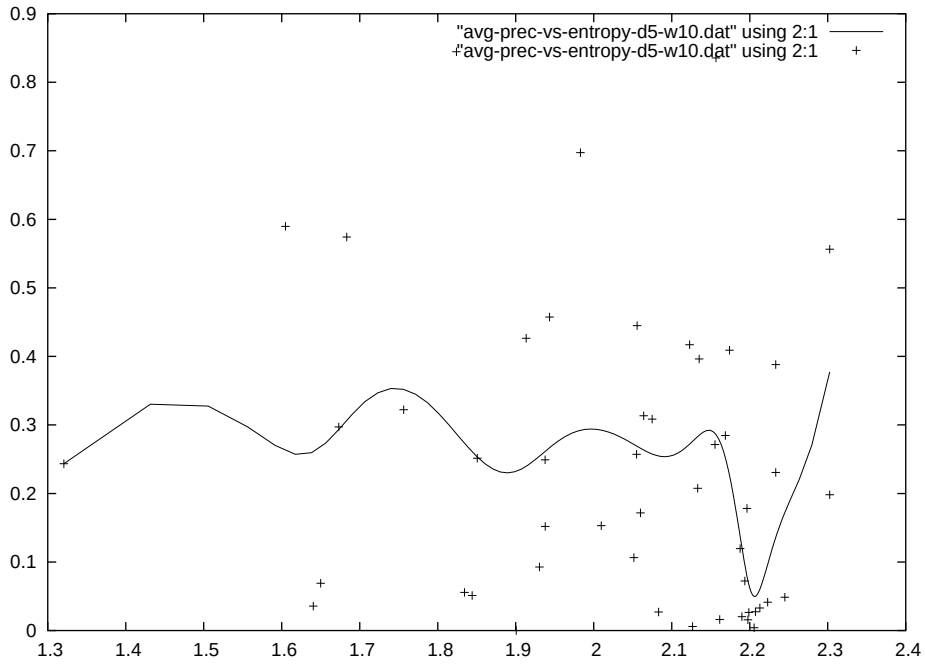

FIG. 7.8.: Entropie / précision moyenne : 5 documents, 10 mots

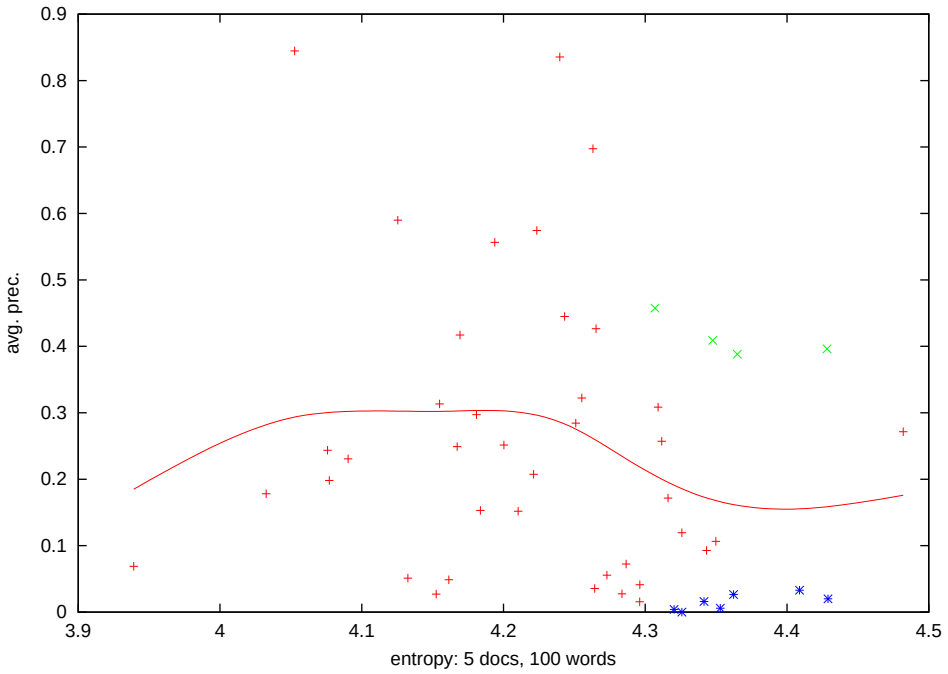

FIG. 7.9.: Entropie / précision moyenne : 5 documents, 100 mots

documents pour les exemples en vert, ces documents sont en fait pertinents.

Nous devons conclure qu'une forte hétérogénéité dans un ensemble de documents n'empêche pas systématiquement qu'un grand nombre de ces documents soit pertinent. Ceci invalide en quelque sorte l'entropie comme seul indicateur de la qualité d'un ensemble de documents (en terme de pertinence pour une requête). L'entropie est néanmoins utile en combinaison avec d'autres attributs, comme nous allons montrer dans la suite.

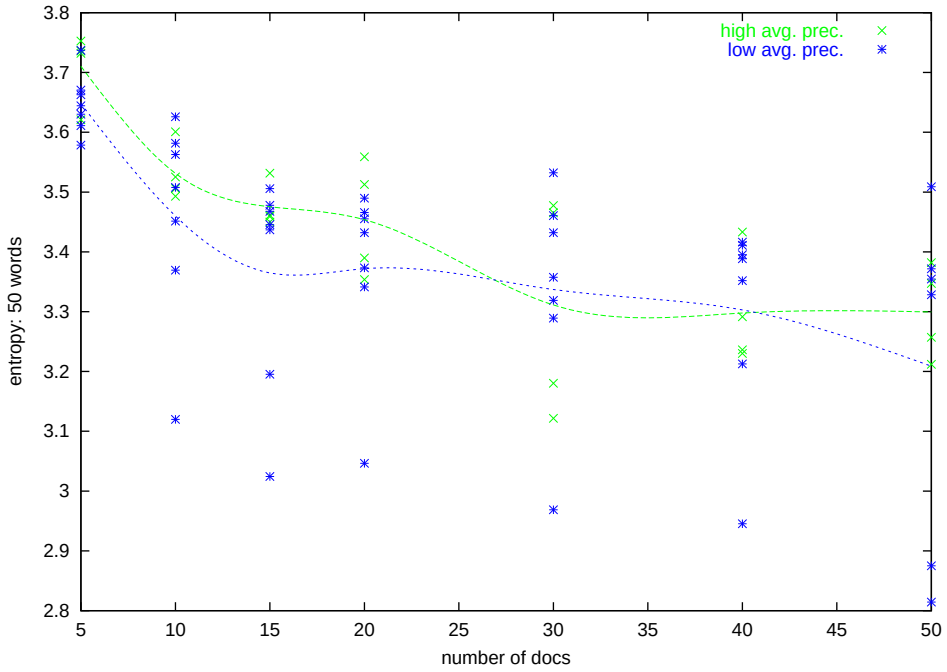

FIG. 7.10.: Entropie en fonction du nombre de documents, 50 mots

7.1.4.2. Similarité Cosinus

Une autre mesure d'homogénéité ou hétérogénéité de l'ensemble de documents retournés est la similarité cosinus moyenne des documents.

Nous avons calculé la similarité pour différentes tailles d'ensembles de documents, appliquant le stemming de Porter et une liste de mots vides, et avec différentes pondérations de type TF.IDF pour les termes. Aucune pondération ne s'est montrée systématiquement plus adaptée à la tâche que d'autres, et nous avons finalement choisi la forme de base :

$$w_{ij} = tf_{ij} * \log \frac{|D|}{df_i}$$

(attribuant le poid w_{ij} au terme T_i dans le document D_j)

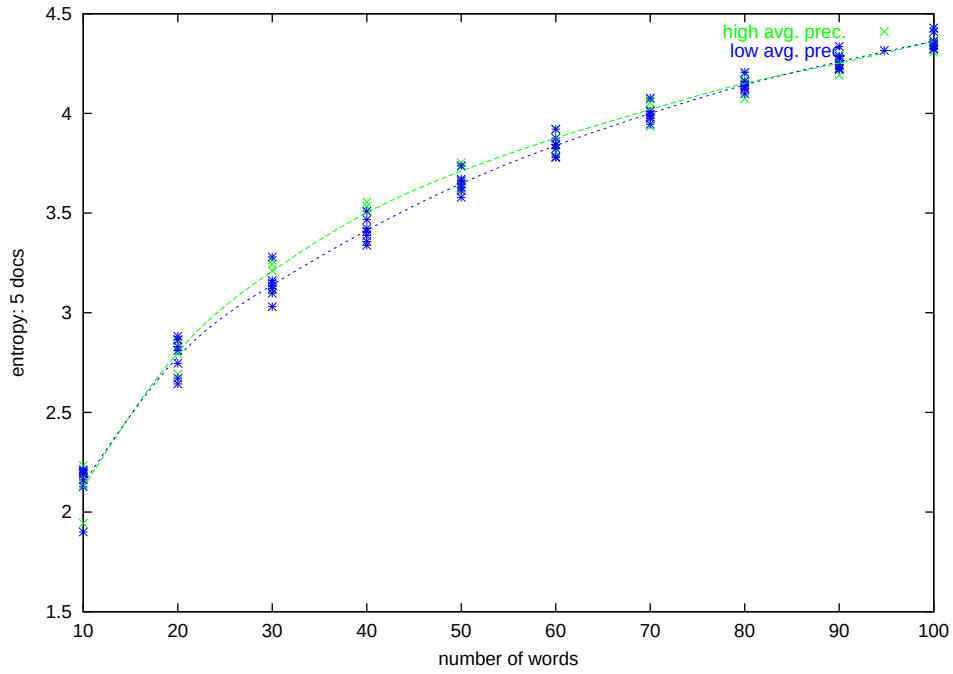

FIG. 7.11.: Entropie en fonction du nombre de mots, 5 documents

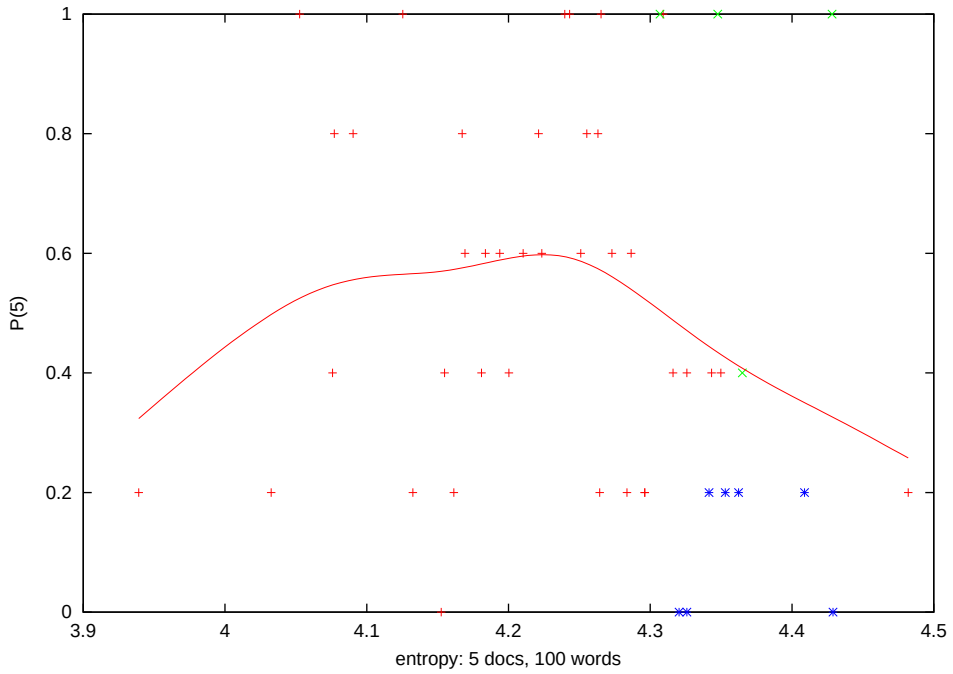

FIG. 7.12.: Entropie / P(5) : 5 documents, 100 mots

On constate une corrélation positive entre similarité moyenne des documents et qualité du résultat pour la totalité des systèmes que nous avons comparés. La corrélation est d'un niveau comparable à celle obtenue avec la mesure d'entropie.

7.1.5. Les scores internes du système de recherche documentaire

Nous avons trouvé une corrélation significative (bien que loin d'être parfaite) entre les scores utilisés pour classer les documents trouvés et la précision moyenne obtenue pour une requête donnée.

Nous avons pu confirmer cette constatation pour un bon nombre de systèmes répandus, fondés sur des paradigmes différents.

Pour une requête donnée, la plupart des systèmes de recherche documentaire classent les documents grâce à un score dérivé plus ou moins directement, soit :

- de la similarité entre documents et requêtes (p.ex. Smart [Salton, 1971])
- de la probabilité bayesienne $P(d|q)$, généralement basée sur des probabilités d'occurrences de termes (p.ex. Okapi [Robertson et Spärck Jones, 1976])
- des réseaux d'inférence bayesienne (p.ex. InQuery [Callan *et al.*, 1992]).

Ces scores sont appropriés pour le classement relatif des documents en réponse à une requête, mais ils ne sont pas supposés donner une mesure absolue de la pertinence (bien que certains modèles en théorie cherchent à estimer la probabilité de pertinence des documents). Ces modèles sont décrits de manière très brève dans le chapitre 5.

Les scores sont généralement disponibles dans les fichiers de résultats soumis aux évaluations TREC par les participants. On trouve ainsi pour chaque requête la liste des 1000 documents ayant obtenu les meilleurs scores, avec les scores associés ayant servi à classer les documents. Ceci nous a permis de comparer les différents scores utilisés par une grande variété de systèmes.

Nous avons utilisé dans nos expériences différentes transformations et normalisations de ces scores. Sur la base des scores attribués par le système de recherche aux documents trouvés, nous avons ainsi calculé des valeurs pour les requêtes. Une requête peut par exemple être représentée par la moyenne des scores des N premiers documents, le score du n^{ieme} document, ou encore le rapport entre les scores du 1^{er} et du n^{ieme} document.

Les résultats présentés ici ont été calculés en utilisant le score moyen sur les 20 premiers documents pour chaque requête.

Nous notons que sur la base des scores de ranking de différents systèmes de recherche documentaire, nous obtenons des mesures fortement liées à la précision moyenne obtenue pour chaque requête. Nous pouvons utiliser ces mesures pour séparer les requêtes en différentes classes qui diffèrent fortement quand à la précision moyenne.

La figure 7.13 montre les classes trouvées pour le système Okapi (ok8amxc) avec leurs limites supérieures et inférieures, la figure 7.14 fait de même pour le système AT&T.

Les bornes supérieures et inférieures que nous trouvons peuvent être très utiles pour décider de la manière de traiter une requête ou les options à présenter à un utilisateur. Pour le système Okapi, aucune requête avec un score en dessous de 3.15 n'obtient une précision moyenne supérieure à 0.28, et aucune requête avec un score en dessous de 3.7 n'atteint une précision moyenne au-delà de 0.59. On peut donc clairement supposer qu'une requête ayant un faible score produira des mauvais résultats de recherche. Elle devra être traitée en conséquence.

Des seuils correspondants peuvent être déterminés pour le système AT&T et d'autres systèmes. Ils sont représentés par des carrés dans les figures correspondantes.

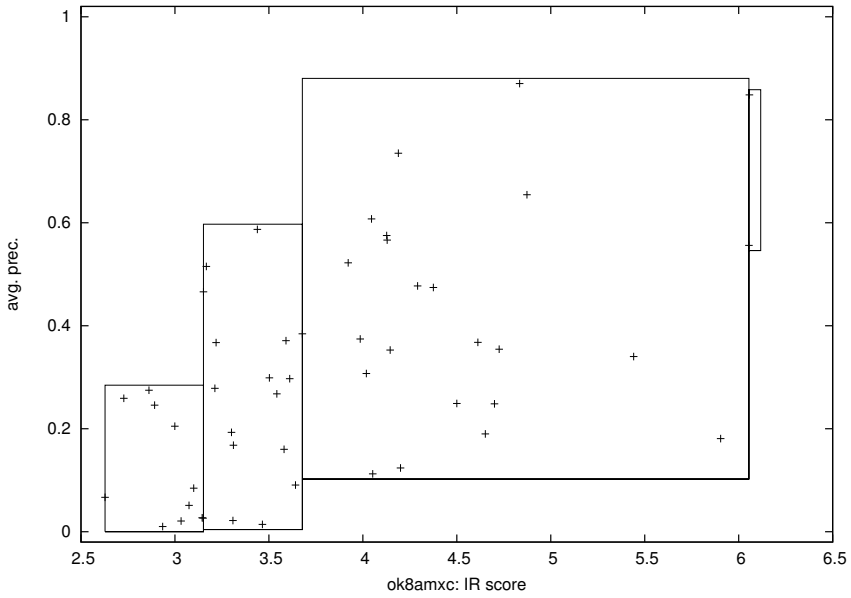

FIG. 7.13.: Score système ok8amxc : séparation en 4 classes

Par contre, le score ne semble pas être du tout corrélé avec la précision moyenne obtenue pour une requête pour certains des systèmes. C'est en particulier le cas pour certains des systèmes manuels, peut-être en raison de l'utilisation d'un langage de requête formel et d'un classement des documents basé initialement sur une recherche booléenne. La figure 7.15 en montre un exemple. Certains systèmes automatiques affichent un comportement semblable (voir figure 7.16). Dans ces cas il n'est pas possible de déduire des seuils utiles sur la performance de recherche attendue.

Ceci démontre que l'estimation de pertinence en termes absolus et le classement relatif des documents pour une requête sont en fait des objectifs très distincts. La forte corrélation trouvée pour les scores utilisés par certains des systèmes nous paraît d'autant plus intéressante.

Nous pensons que l'analyse des différences entre les scores fortement corrélés avec la

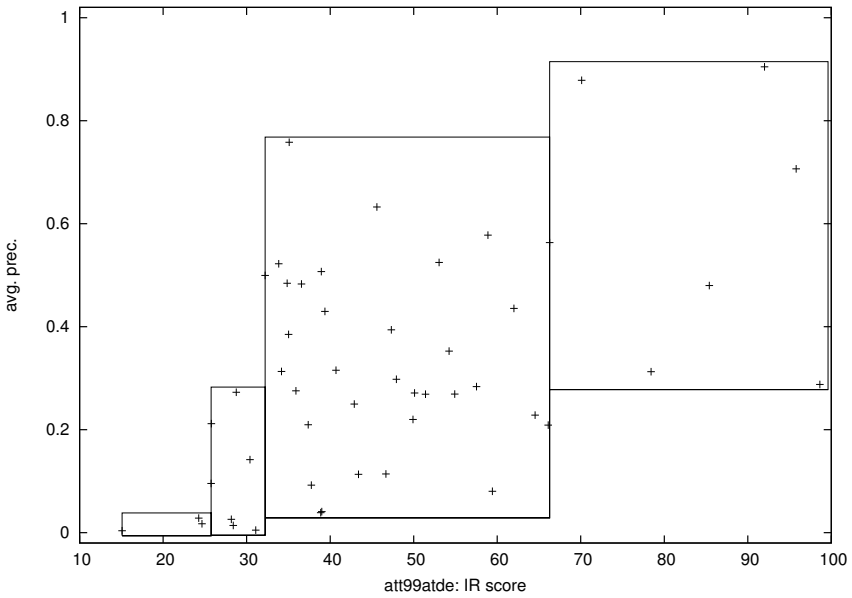

FIG. 7.14.: Score système att99atde : séparation en 4 classes

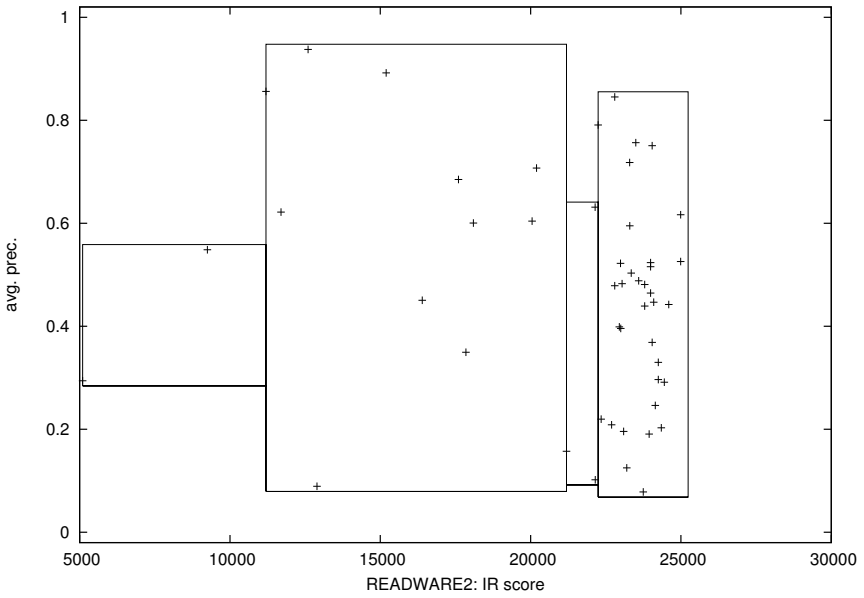

FIG. 7.15.: Score système READWARE2 : séparation en 4 classes

performance absolue et ceux qui ne le sont pas peut apporter des connaissances intéressantes. Celles-ci pourraient être utiles pour divers aspects de la recherche documentaire.

Bien que la *corrélation de rangs* et le quotient $\frac{impurete}{variance}$ montrent un comportement similaire en général, nous notons qu'en particulier les scores *InQuery* ont une forte *corrélation de rangs* alors que l'impureté n'est pas significativement plus basse que pour d'autres systèmes. Il

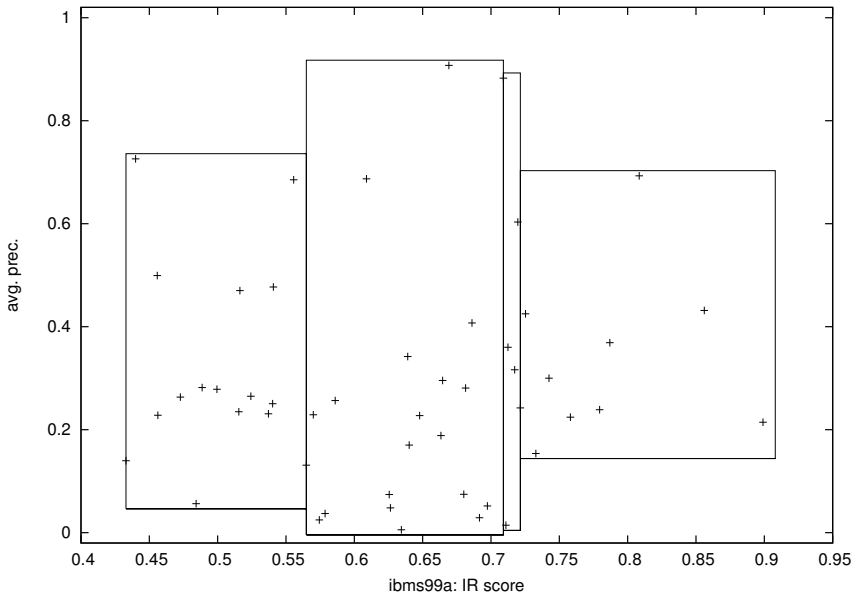

FIG. 7.16.: Score système ibms99a : séparation en 4 classes

n'est pas entièrement clair comment cela se traduit dans une réelle situation d'application en essayant de prendre des décisions basées sur l'estimation de difficulté d'une requête.

Le tableau 7.1 montre la moyenne (sur les 50 requêtes de TREC 8) des précisions moyennes, la variance globale, l'impureté, ainsi que la corrélation de rangs et la valeur *P* pour certains des participants à TREC. Les noms des systèmes sont ceux utilisés lors des soumissions des résultats aux campagnes TREC.

En résumé, nous trouvons généralement une réduction considérable de l'impureté par rapport à la variance globale de la précision moyenne. Beaucoup de ces mesures ont une corrélation de rang avec la précision moyenne plus forte que la plupart des prédicteurs publiés, par exemple le *clarity score* de [Cronen-Townsend *et al.*, 2002].

Il est intéressant de constater qu'il n'y a pas de lien entre la capacité de prédire la performance de recherche sur la base des scores utilisés par différents systèmes, et les performances en tant que système de recherche documentaire. Les scores attribués aux documents par le système *InQuery* (INQ603) sont fortement liés à la qualité du résultat, les scores correspondent donc en quelque sorte à une mesure absolue de pertinence des documents correspondants. Par contre, le système *IBM* (ibms99a) est globalement plus performant, bien qu'il classe les documents d'après un score qui ne donne pas d'indication dans l'absolu de la précision obtenue.

	Name	*Impureté*	*Rho*	*P-Value*	MAP
automatic	INQ603	0.4457	0.7129	$3.26e-08$	0.2658
	tno8d3	0.602	0.5695	$2.24e-05$	0.2921
	MITSLStd	0.5796	0.5490	$4.88e-05$	0.2978
	Sab8A4	0.5243	0.5264	$1.09e-04$	0.2607
	ok8amxc	0.5810	0.5183	$1.43e-04$	0.3168
	att99atde	0.5677	0.4648	$7.64e-04$	0.3165
	Flab8atd2	0.6929	0.3272	$2.07e-02$	0.2929
	pir9Attd	0.7762	0.1331	0.3553	0.3206
	ibms99a	0.7975	0.0653	0.6509	0.3005
	fub99td	0.8237	−0.0238	0.8691	0.3063
manual	READWARE2	0.8313	−0.2392	0.0943	0.4692
	8manexT3D1N0	0.7589	0.1405	0.3291	0.3346
	iit99ma1	0.8599	0.1088	0.4504	0.4103
	CL99XTopt	0.8767	0.1001	0.4877	0.3765
	orcl99man	0.9830	NA	NA	0.4130

TAB. 7.1.: Impureté, corrélation de rangs et moyenne des précisions moyennes

7.1.6. Combinaison et comparaison de plusieurs systèmes

En utilisant plusieurs systèmes de recherche documentaire (ou différentes configurations du même système), il est possible de combiner les listes de documents retournées ou de comparer les différentes listes.

Comme l'ont démontré différentes études sur le "pooling" des résultats dans le cadre de campagnes d'évaluation, les documents pertinents pour une requête ont tendance à être trouvés par plusieurs des systèmes participants. Intuitivement, on peut également supposer que différents systèmes seront plus facilement en accord sur les documents retournés quand les résultats sont de bonne qualité.

Différentes mesures permettent de quantifier la similarité entre plusieurs listes de documents X et Y. Nous avons choisi des variantes de la forme :

$$\mathrm{sim}(X,Y) = \sum_{d \in D} f(\mathrm{rank}_X(d)) \cdot f(\mathrm{rank}_Y(d))$$

où $\mathrm{rank}_X(d)$ est le rang du document d dans la liste X et f une fonction donnant un poids plus élevé aux documents en tête de liste.

Nous avons calculé ces scores pour toutes les combinaisons de deux systèmes. Dans presque tous les cas, on obtient une corrélation très significative avec la qualité du résultat fourni par l'un des systèmes. Pour un grand nombre de combinaisons, la corrélation avec la précision moyenne est considérablement plus forte que pour les autres indicateurs

présentés dans ce document.

Un cas particulier est la comparaison des résultats obtenus par un même système avec des paramétrages différents, par exemple avec ou sans expansion des requêtes. Ceci rejoint d'une certaine manière le prédicteur utilisé par [Yom-Tov *et al.*, 2005a], basé sur les documents trouvés pour les différentes sous-requêtes.

7.1.7. Bilan

Le tableau 7.2 montre les corrélations de rangs ρ ainsi que le *p-values* (pour la corrélation de rang de Spearman) pour une partie des attributs que nous avons utilisés avec la précision moyenne obtenue par une petite sélection de systèmes de recherche documentaire. Nous avons en tout utilisé plus de 50 attributs, le tableau n'en présente donc que quelques variantes.

Les attributs (*features*) sont regroupés selon les informations sur lesquelles elle sont basées.

Comme nous l'avons noté préalablement, aucune de ces mesures ne montre de corrélation parfaite avec les performances obtenues, mais un certain nombre d'entre elles peuvent néanmoins être utile pour l'estimation automatique de difficulté de requêtes, surtout quand elles sont utilisées en combinaison.

Il est intéressant de constater que les corrélations varient considérablement selon le système utilisé.

7.2. Combinaison des différentes mesures

Comme nous l'avons montré dans la section précédente, il est possible de trouver des indicateurs de performance attendue qui sont corrélés (à des degrés différents) avec la qualité réelle du résultat. Néanmoins, aucun des ces indicateurs ne permet à lui seul de prendre des décisions très fiables sur le résultat de la recherche.

Nous avons donc décidé de combiner plusieurs prédicteurs en utilisant des méthodes d'apprentissage et de classification automatique.

7.2.1. Apprentissage et classification automatique

Ayant déterminé un ensemble d'attributs calculables sur la base de la requête même ainsi que (selon les informations disponibles) les résultats de la recherche et éventuellement des connaissances sur le processus ayant mené à ces résultats (scores des documents, etc.), nous représentons chaque requête (ou entité requête-réponse) comme vecteur dont la dimension dépend des attributs choisis.

feature		ok8amxc		INQ603		ibms99a		tno8d4		mds08a2	
		rho	p-value	rho	p-value	rho	p-value	rho	p-value	rho	p-value
docs	entropy (10 docs)	-0.0950	5.19E-01	**-0.2423**	**9.35E-02**	**-0.2841**	**5.07E-02**	**-0.3223**	**3.13E-02**	**-0.3439**	**2.28E-02**
	entropy (15 docs)	**-0.2778**	**5.90E-02**	-0.0607	6.78E-01	-0.1851	2.18E-01	**-0.2562**	**9.33E-02**	-0.0421	7.88E-01
	mcs (50 docs)	0.1359	3.46E-01	**0.2514**	**7.83E-02**	0.0383	7.91E-01	0.1778	2.16E-01	-0.0151	9.17E-01
retrieval score	mean (10 docs)	**0.5194**	**1.39E-04**	**0.7580**	**< 2.2E-16**	0.1802	2.10E-01	**0.5017**	**2.48E-04**	**0.5409**	**6.55E-05**
	ratio (50th doc)	**-0.4464**	**1.29E-03**	**-0.5304**	**9.53E-05**	**-0.3082**	**2.99E-02**	**-0.3688**	**8.76E-03**	**-0.4638**	**7.90E-04**
	score (1st doc)	**0.5043**	**2.28E-04**	**0.5935**	**8.55E-06**	0.1870	1.93E-01	**0.4563**	**9.76E-04**	**0.5458**	**5.49E-05**
	score (5th doc)	**0.5223**	**1.26E-04**	**0.7554**	**< 2.2E-16**	0.1795	2.11E-01	**0.5029**	**2.39E-04**	**0.5148**	**1.62E-04**
query title	length	**-0.2749**	**5.37E-02**	-0.1032	4.75E-01	-0.2149	1.34E-01	-0.2258	1.15E-01	-0.1233	3.93E-01
	senses (de Loupy/Bellot)	0.0997	4.90E-01	0.2139	1.35E-01	0.1270	3.78E-01	0.1095	4.48E-01	0.0876	5.44E-01
	senses (total)	-0.1372	3.41E-01	-0.2029	1.57E-01	-0.1419	3.25E-01	**-0.2758**	**5.29E-02**	-0.1437	3.18E-01
	synonyms (avg.)	-0.1464	3.25E-01	**-0.2499**	**9.04E-02**	-0.1302	3.82E-01	-0.2257	1.27E-01	-0.2026	1.72E-01
	synonyms (total)	-0.1324	3.74E-01	-0.2036	1.70E-01	-0.1158	4.37E-01	**-0.2569**	**8.14E-02**	-0.2079	1.60E-01
query title + description	length	-0.1599	2.66E-01	-0.0415	7.74E-01	-0.0070	9.62E-01	-0.1622	2.60E-01	-0.0714	6.21E-01
	senses (de Loupy/Bellot)	0.0760	5.99E-01	0.1953	1.74E-01	0.1711	2.34E-01	0.0061	9.66E-01	-0.0075	9.59E-01
	senses (total)	-0.1607	2.64E-01	-0.2164	1.31E-01	-0.0784	5.88E-01	**-0.2435**	**8.85E-02**	**-0.2462**	**8.49E-02**
	synonyms (avg.)	-0.0207	8.86E-01	-0.1544	2.84E-01	-0.0259	8.58E-01	-0.1272	3.78E-01	-0.1205	4.04E-01
	synonyms (total)	-0.1095	4.48E-01	-0.1637	2.55E-01	0.0078	9.57E-01	-0.1788	2.14E-01	-0.1824	2.04E-01

TAB. 7.2.: Attributs et leur corrélation avec la précision moyenne sur la collection TREC 8 adhoc

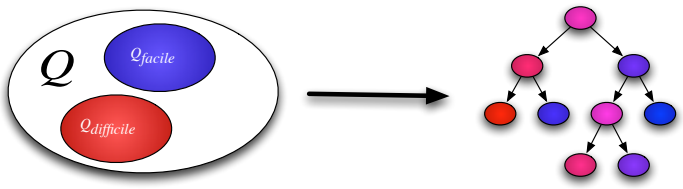

FIG. 7.17.: Apprentissage d'un classifieur

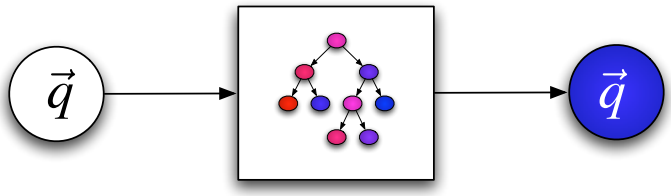

FIG. 7.18.: Application d'un classifieur

À l'aide d'un corpus de requêtes pour lesquelles la performance du système est connue, nous pouvons ensuite appliquer des techniques d'apprentissage et de classification automatique (figure 7.17) afin d'obtenir des prédictions sur le critère de classification (en l'occurrence la difficulté de la requête en terme de précision moyenne obtenue) pour de nouvelles requêtes ne faisant pas partie du corpus d'apprentissage (figure 7.18).

Nous avons utilisé différents algorithmes de classification génériques sur différentes représentations vectorielles de l'information disponible. Le toolkit WEKA [Witten et Frank, 1999] fournit un grand nombre de classifieurs différents, mais nous avons également essayé d'autres implémentations :

- arbres de décision (implémentation de Frédéric Béchet, LIA)
- CAL5 [Müller et Wysotzki, 1997]
- Dipol [Schulmeister et Wysotzki, 1997]
- SVM*light* [Joachims, 1998a]
- WEKA : decision trees, SVM, NaiveBayes, neural net, ...

Les algorithmes à base d'arbres présentent l'avantage de construire des règles de classification lisibles et interprétables. Ceci peut fournir des informations intéressantes sur le processus de classification, utiles pour des améliorations éventuelles de l'approche. Cependant, en partie d'autres classifieurs (et en particulier les machines à vecteur support sur des faibles quantités de données d'apprentissage) obtiennent de meilleures performances de classification.

Nous avons également effectué quelques essais avec des classifieurs conçus spécialement pour la catégorisation de textes, les *Semantic Classification Trees (SCT)* [Kuhn et de Mori, 1995]. À partir d'une collection de textes (dans notre cas les requêtes)

ceux-ci construisent un arbre basé sur des expression régulières (extraites automatiquement) qui servent à diviser les exemples sur les différentes branches de l'arbre.

7.2.2. Les classifieurs utilisés – un bref état de l'art

Les classifieurs les plus intéressants pour nous, par leur fonctionnement ou leur performances, ont été les arbres de décision (DT) et les machines à vecteur support (SVM).

À part pour les SCTs décrits plus bas, la représentation des données a été essentiellement identique pour tous les classifieurs utilisés. À chaque requête est associé un ensemble de valeurs numériques, ce qui permet de voir la requête comme un point dans un espace vectoriel.

7.2.2.1. Arbres de décision

Les arbres de décision sont construits par séparations consécutives de l'ensemble des exemples du corpus d'entraînement en sous-ensembles. Chacune de ces séparation se fait selon un des attributs présents dans la description de la requête.

La séparation est faite de manière à obtenir des noeuds les plus purs possibles, contenant de préférence uniquement des exemples appartenants à une seule classe. Chaque noeud est séparé à nouveau de manière récursive tant que des exemples des plusieurs classes sont présents. Ce processus peut également terminer quand le nombre d'exemples dans un noeud est trop faible ou qu'aucun attribut ne permet d'améliorer la séparation des exemples.

L'objectif est d'obtenir un arbre petit, mais avec un bon pouvoir de classification. Pour cela, à chaque étape, l'algorithme choisit l'attribut susceptible d'obtenir, le plus rapidement possible, une bonne séparation des classes. Dans le cas d'attributs numériques, et pour la construction d'un arbre binaire, ce choix comprend le choix de l'attribut même, ainsi qu'un seuil sur la valeur le l'attribut, séparant les deux sous-ensembles d'exemples.

La mesure utilisée pour évaluer la pureté de noeuds est *l'information* qui est mesurée en *bits*. Il s'agit là de la quantité d'information supplémentaire nécessaire pour déterminer la classe d'un exemple, sachant qu'il appartient à un certain noeud de l'arbre.

Dans le cas d'un ensemble contenant deux classes d'exemples de taille égale sans aucune autre information à priori, 1 bit d'information est suffisant pour déterminer la classe d'un exemple donné. Ce cas rejoint la notion de *bit* comme chiffre binaire, ou chacune des deux valeurs possibles représenterait une des classes.

Dans le cas ou la distribution des classes n'est pas équilibrée de cette manière, moins d'information est nécessaire pour déterminer la classe d'un exemple, le cas extrême étant la situation d'un noeud ne contenant que des exemples d'une seule classe. Dans ce cas, aucune information additionnelle n'est nécessaire, ce qui correspond à 0 bits.

La *valeur d'information* ou *entropie* pour n classes se calcule ainsi :

$$\text{entropie}(p_1, p_2, \ldots, p_n) = -p_1 \log_2 p_1 - p2 \log_2 p_2 \ldots - p_n \log_2 p_n$$

Pour choisir l'attribut à utiliser on s'intéresse au *gain d'information*, c.à.d. le rapport entre la moyenne (pondérée par le nombre d'exemples) des entropies des noeuds enfants et l'entropie du noeud parent. Dans l'exemple illustré dans la figure 7.19 nous avons une entropie initiale de 0.91 sur 40 exemples. Après la séparation en deux classes, nous obtenons, 26 exemples avec une entropie de 1, et 14 exemples avec une entropie de 0. La moyenne est alors $\frac{26 \cdot 1 + 14 \cdot 0}{40} = 0.65$. Malgré l'augmentation de l'entropie pour un des noeuds, le fait d'avoir isolé un certain nombre d'exemples de requêtes faciles a donc permis de réduire considérablement l'entropie moyenne sur l'ensemble des noeuds considérés.

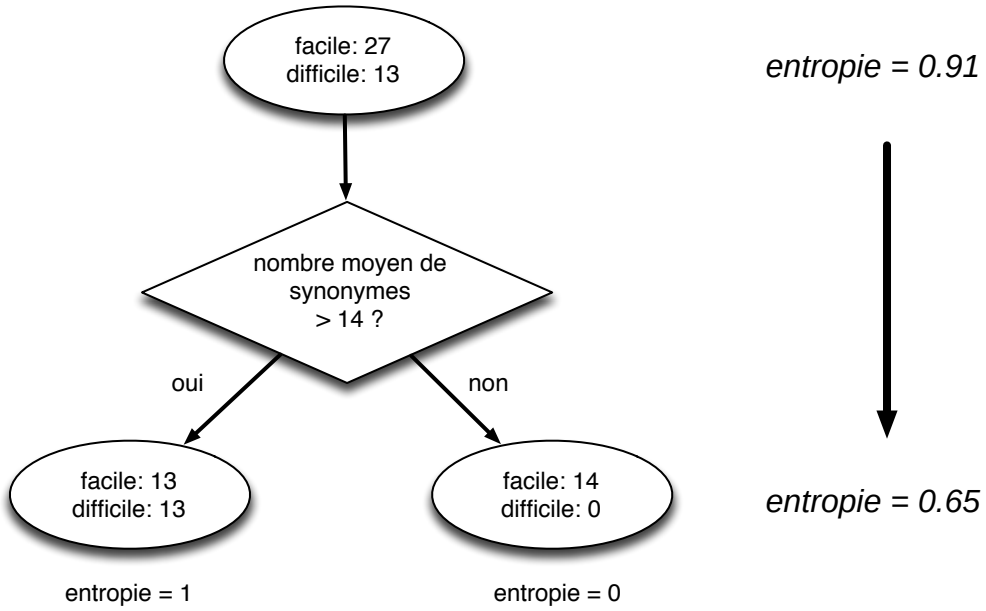

FIG. 7.19.: Fragment d'arbre de décision

À chaque étape de la construction de l'arbre, l'algorithme choisit parmi les attributs disponibles (après discrétisation des valeurs continues) celui qui optimise le gain d'information, éventuellement modéré par l'utilisation du *gain ratio*, surtout dans le cas d'arbres non binaires.

La méthode du *pruning* est utilisée pour éviter le surapprentissage, c.à.d. une adaptation trop forte aux données d'apprentissage. Des paramètres définissant des seuils sur la taille minimale des feuilles, ou sur le gain d'information minimal pour un embranchement, permettent de contrôler la complexité de l'arbre appris et d'adapter la construction aux données disponibles.

7.2.2.2. SVMs

Une méthode beaucoup utilisée aujourd'hui et qui est particulièrement performante même avec une faible quantité de données d'apprentissage, est l'utilisation de machines à vecteurs supports (*support vector machines, SVM*), proposés par Vapnik ([Vapnik, 1982, Vapnik, 1995]).

Les SVMs sont une classe d'algorithmes d'apprentissage qui ont été utilisés avec succès dans de nombreuses tâches d'apprentissage.

Ils offrent en particulier une bonne approximation du principe de minimisation du risque structurel. L'idée de la minimisation du risque structurel est de trouver une hypothèse h pour laquelle l'erreur vraie minimale est garantie. L'erreur vraie de h est la probabilité que h fasse une erreur sur un exemple non-vu et extrait aléatoirement du corpus de test.

Chaque exemple, défini par l'ensemble d'attributs qui lui est associé, est d'abord vu comme un point dans l'espace $\mathbb{R}^n$ (avec n le nombre d'attributs). Ensuite une projection peut être appliquée vers un espace de dimension supérieure, dans le but de rendre linéairement séparables de données qui ne le sont pas dans leur représentation d'origine.

Ils permettent de construire un classifieur à valeurs réelles qui découpent le problème de classification en 2 sous-problèmes : transformation non-linéaire des entrées et choix d'une séparation linéaire *optimale*.

Dans le cas de données linéairement séparables, l'algorithme détermine l'hyperplan maximisant la distance par rapport à l'enveloppe convexe de chaque classe. Cet hyperplan ne dépend que des exemples de chaque classe qui sont les plus proches des autres classes, les vecteurs supports, les autres exemples n'ont aucune incidence sur le résultat. Ceci contribue à la robustesse de l'algorithme et réduit le risque de surapprentissage.

Dans le cas de données qui ne sont pas linéairement séparables, on applique d'abord une projection dans un espace vectoriel différent, avant de déterminer l'hyperplan séparateur dans cet espace.

Le premier sous-problème à traiter est donc celui de travailler dans un espace où les données soient linéairement séparables. Les données sont alors projetées dans un espace de grande dimension par une transformation basée sur un noyau (voir figure 7.20).

Le noyau (*kernel function*) est une fonction qui retourne la valeur du produit scalaire des images des 2 arguments $K(x_1, x_2) = \langle \Phi(x_1), \Phi(x_2) \rangle$. Les noyaux les plus utilisés sont de type linéaire, polynomial ou RBF (en particulier gaussien).

Le deuxième sous-problème est traité dans cet espace transformé. Les classes y sont séparées par des classifieurs linéaires qui déterminent un hyperplan optimal. L'hyperplan optimal est celui qui sépare correctement toutes les données et qui maximise la marge, c.à.d. la distance du point le plus proche à l'hyper-plan (représentée par d dans la figure 7.21). Les hyperplans peuvent être déterminés au moyen d'un nombre de points limité qui

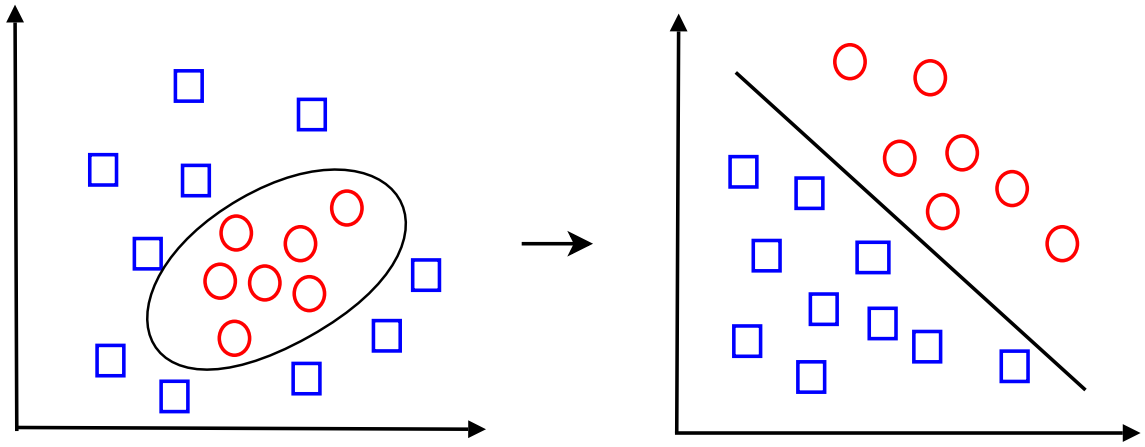

FIG. 7.20.: Projection des données d'entrée dans un espace où elles sont linéairement séparables

seront appelés les « vecteurs supports ».

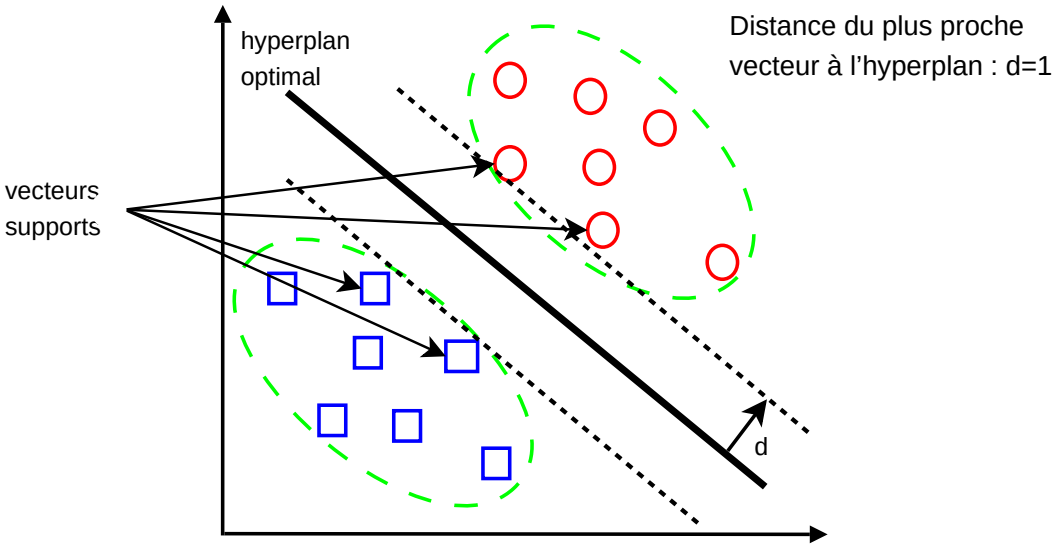

FIG. 7.21.: Hyperplan optimal et marge maximale

De travailler dans un espace à grande dimension pose certains problèmes. D'une part, en classification automatique on est classiquement confronté à du surapprentissage quand les données sont représentés par un trop grand nombre d'attributs car cela peut permettre d'isoler les exemples individuels au lieu de déterminer une description généralisable. Le fait de ne se baser que sur les vecteurs supports réduit ce risque.

D'autre part, la complexité computationnelle de certaines opérations dépend de la di-

mensionnalité et peut être très grande. Dans le cas des SVMs, les opérations coûteuses peuvent être effectuée dans l'espace d'origine à petite dimension. En particulier, le produit scalaire entre les vecteurs de test et les vecteurs supports peut être calculé avant application de la projection (du moins pour les noyaux polynomiaux).

Le poids calculatoire reste donc limité, même après projection dans un espace de dimension élevée qui ne serait pas traitable sans ces particularités.

Les SVMs ont été par le passé appliqués avec succès à des tâches de classification de texte ([Joachims, 1998b]). Pour ceci, la technique la plus simple employée est celle du sac de mots où un texte est représenté par le nombre d'occurrences des mots qu'il contient. À chaque texte correspond ainsi un point dans l'espace $\mathbb{R}^n$, n étant la taille du lexique (donc de tous les mots qui peuvent apparaître dans un texte). Le nombre d'occurrences peut éventuellement être remplacé par une pondération ou normalisation différente, par exemple de type TF.IDF.

Les SVMs sont bien adaptés à cette application en particulier pour différentes raisons :

- Ils ont le potentiel pour gérer ce grand nombre de données.
- Le vecteur représentant chaque phrase contient peu de données qui ne sont pas des 0. Les SVMs sont adaptés aux problèmes de ce type (vecteurs creux).
- La plupart des problèmes de catégorisation de textes sont linéairement séparables.

7.2.2.3. SCTs

Les arbres de classification sémantique (*semantic classification trees*, *SCT*, voir [Kuhn et de Mori, 1995]) sont spécialement conçus pour la classification de textes.

Leur objectif était à l'origine l'utilisation pour la "compréhension" du langage naturel. Pour cela ils permettent d'apprendre des règles représentatives d'une interprétation sémantique à partir d'un corpus d'apprentissage.

La particularité des SCTs par rapport à d'autres approches sémantiques ou syntaxiques est l'utilisation extensive de "trous" (*gaps*), de mots ou groupes de mots qui sont ignorés, par rapport au techniques classiques qui visent à couvrir tous les mots d'une phrase.

Les SCTs ont les propriétés suivantes :

- Les questions dans les noeuds d'un SCT se présentent sous forme d'expressions régulières, composées de chaînes des caractères ainsi que d'un symbole spécial de *gap*.
- La génération et sélection des questions est entièrement automatique.

La nouveauté des algorithmes de SCT par rapport à des arbres de décision génériques réside dans la construction des questions pour l'apprentissage de l'arbre de décision. Cette construction se fait à partir d'un ensemble d'expressions régulières Π_0 associé à la racine de l'arbre :

soient V, l'ensemble des mots du vocabulaire du corpus d'entraînement C_0, $w \in V$ n'importe quel mot du vocabulaire, le symbole « + » indiquant la présence de n'importe quelle

séquence de mots non vide dans l'expression régulière, Π_0 contient les quatre éléments suivants :

$$\Pi_0 = \{w, +w, w+, +w+\}$$

L'ensemble des questions appliquées à la racine, N_0, est obtenu en considérant toutes les expressions régulières Π_0 possibles appliquées sur tous les mots w du vocabulaire V selon les quatre éléments de Π_0. Il y a donc $4 \times |V|$ possibilités où $|V|$ représente la taille du vocabulaire. Chaque expression régulière est alors testée à la racine de l'arbre. Le critère d'impureté de Gini est utilisé pour choisir la meilleure question à chaque nœud N_i. Soient les k classes $c_1, c_2, ..., c_k$ dont les probabilités de répartition sont $p_1, p_2, ..., p_k$ alors le critère de Gini du nœud N_i s'exprime par :

$$G(N_i) = 1 - \sum_{j=0}^{k} {p_j}^2$$

La meilleure question pour N_i est celle qui apporte la plus grande variation d'impureté entre N_i et ses fils. Si les deux enfants de N_i sont notés N_{i+1}^{oui} et N_{i+1}^{non}, la variation d'impureté Δ_i est alors définie par :

$$\Delta_i = G(N_i) - \frac{|N_{i+1}^{oui}| \times G(N_{i+1}^{oui}) + |N_{i+1}^{non}| \times G(N_{i+1}^{non})}{|N_i|}$$

Par exemple, si $+W$ est l'expression régulière qui maximise Δi, la phrase qui est acceptée par cette expression régulière fait partie d'un corpus C_1^{oui} associé au fils de la branche du corpus nommé OUI. Les autres font partie du corpus C_1^{non} associées au fils nommé NON. L'ensemble des questions associées à C_1^{oui} est obtenu par la substitution suivante : $+ \rightarrow \Pi_0$ nous menant à $\Pi_0 W$. En général, étant donnée une expression régulière : $+W_1 + W_2 + \ldots + W_i + \ldots$ où W_i est déjà déterminé par une série de mots, les questions sont générées en remplaçant chaque $+$ par Π_0, c'est à dire :

$$\begin{aligned} &\Pi_0 W_1 + W_2 + \ldots + W_i + \ldots \\ &+W_1 \Pi_0 W_2 + \ldots + W_i + \ldots \\ &+W_1 + W_2 \Pi_0 \ldots + W_i + \ldots \\ &\ldots \end{aligned}$$

Soit $|+|$ le nombre de symboles $+$ dans l'expression régulière originale, le nombre de cas est alors : $4 \times |+| \times |V|$.

Lorsque le SCT est construit, il prend des décisions sur la base de règles de classification

statistique apprises sur ces expressions régulières.

La figure 7.22 montre un exemple d'arbre.

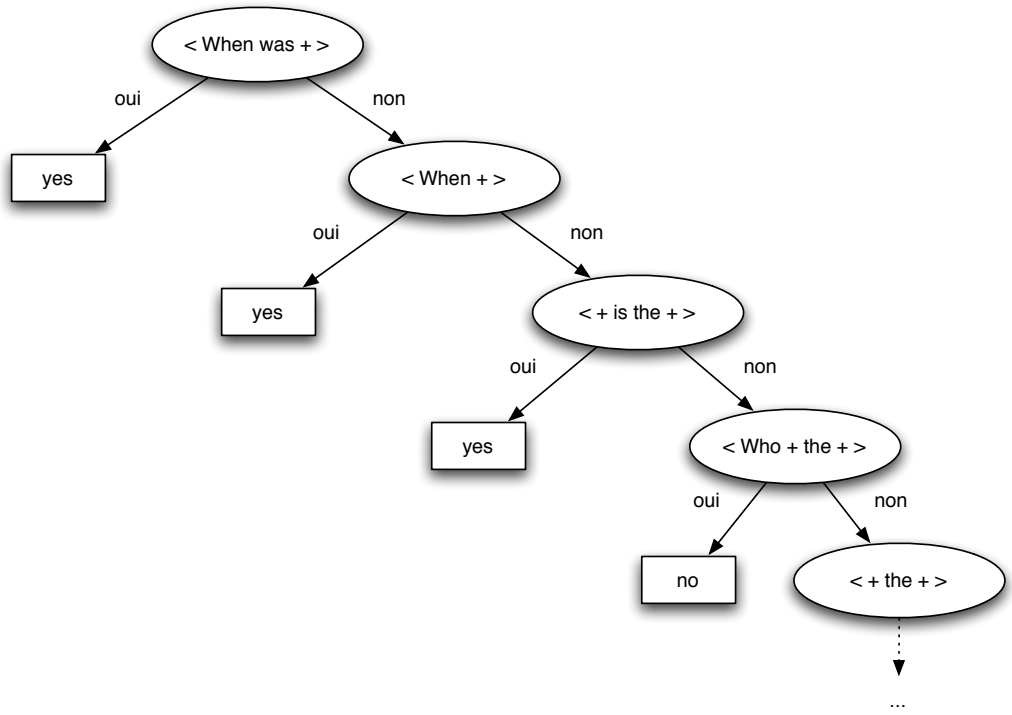

FIG. 7.22.: Schéma simplifié d'un arbre de classification sémantique

7.2.2.4. Les algorithmes de Boosting

Le Boosting est une méthode générale pour améliorer les performances de n'importe quel algorithme d'apprentissage. Un algorithme de Boosting peut théoriquement transformer un algorithme d'apprentissage basique (appelé « weak learner ») avec des performances juste un peu meilleures que le hasard en un algorithme très performant.

Le principe de ces algorithmes de Boosting est de re-pondérer à plusieurs reprises les exemples d'apprentissage et de refaire tourner l'algorithme d'apprentissage sur ces exemples re-pondérés. Les exemples mal classés sont re-pondérés à la hausse tandis que les biens classés sont re-pondérés à la baisse. Ainsi, le boosting force cet algorithme à concentrer ses efforts d'apprentissage sur les exemples les plus difficiles.

L'hypothèse finale est un vote pondéré des différentes hypothèses obtenues à chaque instance de l'algorithme d'apprentissage. L'algorithme AdaBoost [Freund et Schapire, 1996] est présenté dans la figure 7.23.

Entrées :

1. un ensemble d'apprentissage de m exemples $\langle (x_1, y_1), \ldots, (x_m, y_m) \rangle$ avec le label $y_i \in Y = \{1, \ldots, k\}$;
2. un nombre d'itérations T ;
3. un classifieur basique CB.

Initialiser la distribution $D_1(i) = \frac{1}{m}$ pour tout i
Faire pour $t = 1, 2, \ldots, T$

1. Appel à CB avec la distribution D_t.
2. Récupérer une hypothèse $h_t : X \rightarrow Y$
3. Calculer l'erreur de $h_t : \epsilon = \sum_{i:h_t(x_i) \neq y_i} D_t(i)$.
4. Fixer $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
5. Mettre à jour la distribution D_t : $D_{t+1}(i) = \frac{D_t(i)}{Z_i} \times \left\{ \begin{array}{ll} \beta_t & si\ h_t(x_i) = y_i \\ 1 & sinon \end{array} \right\}$
   où Z_t est une constante de normalisation (choisie afin que D_{t+1} soit une distribution)

Sortir l'hypothèse finale : $h_{fin}(x) = \underset{y \in Y}{\text{ArgMax}} \sum_{t:h_t(x)=y} \log \frac{1}{\beta_t}$.

FIG. 7.23.: Algorithme AdaBoost

7.2.3. Résultats – Estimation de difficulté

Utilisant un corpus de requêtes pour lesquelles la performance de système de recherche était connu, nous avons appliqué des techniques de classification automatique pour obtenir des prédictions sur la difficulté d'une requête donnée. L'évaluation de méthodes d'apprentissage automatique nécessite au moins deux ensembles de données séparés, un pour l'apprentissage et un autre pour le test.

Dû au faible nombre de requêtes dans le corpus, nous avons utilisé les méthodes de "leave-one-out" et de validation croisée. Les résultats présentés sont ceux obtenus par validation croisée stratifiée avec 10 plis. Ce choix est justifié aussi bien par la pratique courante qu'en théorie (voir [Witten et Frank, 1999], chapitre 5) et donne une bonne indication de la qualité du classifieur. De plus, nous avons dans certains cas calculé l'intervalle de confiance de nos résultats sur la base de 10 séparations aléatoires différentes du corpus.

La plupart des expérimentations ont été conduites sur la base des données de TREC 8. Nous avons également validé certains résultats sur d'autres corpus, ainsi qu'en utilisant des données de TREC 8 pour l'apprentissage et de TREC 7 pour le test.

Suivant des expérimentations initiales avec une plus grande variété de classifieurs, deux

types de classifieurs, les SVMs et les arbres de décision, ont été retenus pour prédire si une requête est facile (grande précision attendue sur les résultats retournés) ou difficile (faible précision attendue). Pour chaque système participant à TREC 8 des classifieurs ont été appris. Dans certains cas, de la sélection d'attributs a été effectuée au cours du processus d'entraînement.

Sur l'estimation de difficulté des requêtes nous avons séparé les requêtes en deux classes : «faciles» et «difficiles», selon la précision moyenne obtenue par rapport à un seuil fixé.

Nous avons ensuite utilisé la méthode du "leave one out" pour diviser le corpus de requêtes pour l'apprentissage et le test. Pour chaque requête (ou petit sous-ensemble de requêtes) un classifieur a été construit sur la base du reste du corpus et ensuite appliqué à la requête choisie afin d'évaluer la qualité des prédictions obtenues. La précision de classification doit alors être évaluée par rapport à la distribution des classes fixées.

La limite entre les requêtes faciles et difficiles a dû être fixée de manière arbitraire (elle dépendrait du contexte applicatif spécifique). Nous avons choisi la valeur médiane des précisions moyennes obtenues sur l'ensemble de la collection de requêtes pour la plupart de nos expériences. Cette décision a été prise afin d'obtenir une distribution équilibrée des données d'entraînement entre les deux classes. D'avoir une distribution identique pour les différents systèmes pris en compte nous permet également de comparer directement les performances de classification.

Dans la limite des données disponibles, nous avons calculé les ensembles d'attributs pour tous les systèmes ayant participé à TREC 8. Des description détaillées de ces systèmes sont disponibles dans les actes de TREC 8 ([Voorhees et Harman, 1999]).

Le tableau 7.3 donne la précision de prédiction pour une sélection représentative des systèmes ayant obtenu une moyenne des précisions moyennes (*mean average precision / MAP*) supérieure à 20%.

Pour certains systèmes des prédictions fiables sont possibles, avec par exemple une précision de 86% pour le système Okapi sur les requêtes de longueur moyenne (*ok8amxc*).

Un exemple d'un arbre de décision est donné dans la figure 7.24.

Chaque noeud correspond à un test sur un attribut. Chaque branche comporte le résultat de ce test. Les feuilles contiennent le résultat de classification, ainsi que le nombre de classifications correctes et incorrectes.

7.2.3.1. Sélection d'attributs

Nous avons essayé différentes méthodes de sélection d'attributs en amont de l'apprentissage du classifieur (en particulier pour les SVMs). Ceci comprend la sélection des N attributs les plus prédictifs individuellement, la sélection de sous-ensembles d'attributs par CFS (*correlation based feature selection*), ou encore la transformation de l'espace d'attributs

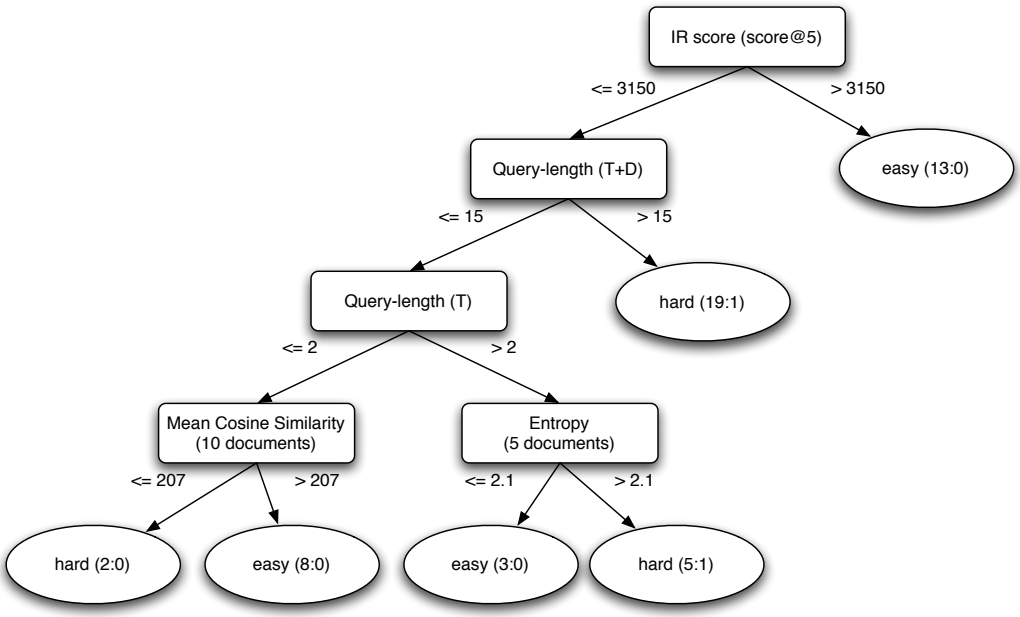

FIG. 7.24.: Arbre de décision pour le système tno8d4. Dans les feuilles, les nombres entre parenthèses indiquent le nombre de classification correctes suivi du nombre d'erreurs de classification

par analyse en composantes principales (ACP). Aucune de ces méthodes ne s'est montrée très efficace de manière systématique (malgré des gains dans certaines configurations).

Pour les arbres de décision, la sélection d'attributs est implicite dans la construction de l'arbre (étant donné le nombre limité d'embranchements, le nombre d'attributs utilisés est également limité). L'algorithme d'apprentissage de l'arbre sélectionne automatiquement les attributs les plus utiles pour la classifications. Le tableau 7.4 donne la liste des attributs utilisés dans la construction d'arbre pour les différent participants à TREC. Ils sont regroupés en quatre ensembles :

- les attributs calculés sur la base des K premiers documents retournés par le système de recherche ;
- les attributs qui dépendent des scores attribués par le système de recherche documentaire pour le classement des documents par rapport à la requête ;
- les attributs qui décrivent la forme courte (titre) de la requête ;
- les attributs concernant la version de longueur moyenne de la requête ;

Le nombre de fois que chaque attribut a été utilisé pour la construction du classifieur. Ceci correspond au nombre de systèmes pour lesquels l'attribut contribue (plus fortement que d'autres attributs disponibles) à la séparation des classes. Un attribut utilisé dans beaucoup de cas peut donc être considéré comme particulièrement utile en tant que prédicteur de performance.

On constate que chaque groupe contient des attributs utilisés de manière assez fréquente. On peut en conclure que les différent types d'attributs ont tous un apport important pour la prédiction de difficulté.

system	*MAP*	*SVM*	*DT*	*DT+SVM*	*coverage*
ok8amxc	32%	56%	86%	82%	66%
mds08a5	23%	50%	82%	79%	56%
mds08a2	24%	56%	82%	77%	70%
ok8asxc	28%	68%	74%	80%	70%
Sab8A4	26%	70%	74%	81%	72%
orcl99man	41%	46%	74%	66%	64%
INQ603	27%	78%	72%	80%	82%
Flab8at	29%	52%	70%	69%	58%
tno8d4	28%	60%	70%	73%	66%
att99atc	29%	66%	68%	70%	86%
uwmt8a2	27%	72%	68%	75%	80%
apl8p	32%	30%	68%	48%	42%
ric8tpx	27%	44%	68%	59%	68%
UT800	26%	56%	68%	71%	56%
weaver1	22%	58%	68%	71%	62%
Mer8Adtnd3	23%	70%	66%	85%	52%
iit99au2	20%	72%	66%	76%	74%
Mer8Adtd1	22%	72%	66%	77%	70%
Mer8Adtd2	22%	72%	66%	77%	70%
tno8d3	29%	52%	66%	67%	54%
ok8alx	32%	66%	64%	78%	54%
INQ602	25%	72%	64%	74%	76%
apl8c221	31%	74%	64%	76%	74%
ic99dafb	25%	70%	62%	70%	80%
acsys8aln2	26%	70%	62%	77%	60%
INQ604	28%	70%	62%	77%	60%
ric8dnx	24%	66%	60%	69%	70%
iit99ma1	41%	70%	60%	73%	66%
Sab8A3	25%	72%	60%	72%	72%
att99atde	32%	74%	60%	76%	66%
GE8ATD3	26%	66%	58%	66%	76%
apl8ctd	29%	68%	54%	65%	74%
iit99au1	23%	70%	54%	68%	68%
ric8dpn	26%	70%	54%	67%	72%
nttd8alx	28%	70%	54%	68%	68%
ibmg99b	26%	72%	54%	70%	66%
ric8dpx	27%	66%	50%	61%	72%
att99ate	28%	74%	50%	67%	72%
nttd8ale	29%	70%	48%	62%	74%
acsys8alo2	26%	74%	46%	69%	52%

TAB. 7.3.: Précision des prédictions pour une sélection de systèmes de recherche documentaire

	score	used by		score	used by
documents	entropy (10 docs)	18	query title	abbreviations	
	entropy (15 docs)	20		technical dictionary	
	entropy (20 docs)	21		hyponyms (avg.)	17
	entropy (5 docs)	28		hyponyms (total)	10
	entropy (50 docs)	29		length (stopped)	
	mcs (10 docs)	14		length (raw)	45
	mcs (15 docs)	14		negations	
	mcs (20 docs)	18		senses (avg.)	15
	mcs (5 docs)	24		senses (de Loupy/Bellot)	47
	mcs (50 docs)	22		senses (total)	12
IR ranking score	mean (10 docs)	13		synonyms (avg.)	8
	mean (15 docs)	2		synonyms (total)	10
	mean (20 docs)	7	query title + description	abbreviations	
	mean (5 docs)	16		technical dictionary	
	mean (50 docs)	6		hyponyms (avg.)	18
	ratio (10th doc)	6		hyponyms (total)	13
	ratio (15th doc)	8		length (stopped)	27
	ratio (20th doc)	9		length (raw)	32
	ratio (5th doc)	16		negations	
	ratio (50th doc)	14		senses (avg.)	11
	score (1st doc)	33		senses (de Loupy/Bellot)	33
	score (10th doc)	5		senses (total)	26
	score (15th doc)	5		synonyms (avg.)	18
	score(20th doc)	8		synonyms (total)	5
	score (5th doc)	25			
	score (50th doc)	10			

TAB. 7.4.: Attributs avec le nombre de systèmes pour lesquels ils sont utilisés par le classifieur d'arbres de décision (MCS est la similarité cosinus moyenne)

8. Décision sur l'enrichissement des requêtes

Sommaire

Au delà de l'estimation de la qualité du résultat attendue, il est possible d'utiliser la classification automatique pour prendre des décisions sur l'application de méthodes d'expansion des requêtes (ou d'autres traitements spécifiques).

Il a été noté (entre autre dans [Xu et Croft, 1996, Carpineto *et al.*, 2001]) que l'effet de techniques d'enrichissement telle le *blind relevance feedback* dépend de la qualité du résultat obtenu avant expansion. Étant donné que l'enrichissement est basé sur les documents trouvés dans une première passe de recherche et qu'il est fondé sur la supposition qu'un nombre important de ces documents est pertinent, cette technique échoue logiquement lorsque le résultat initial est de trop mauvaise qualité. D'autre part, dans le cas d'une bonne qualité initiale, le reclassement des documents a peu de chances d'améliorer le résultat.

Ceci mène à supposer que les méthodes utilisées pour estimer la difficulté de requêtes peuvent également être utiles pour l'application sélective de techniques d'expansion. Plusieurs équipes ont récemment tenter d'exploiter ce fait, tels [Sehgal et Srinivasan, 2005], [Yom-Tov *et al.*, 2005a] et d'autres.

D'autre part, la constatation que différents systèmes dépendent de manière distincte de certaines caractéristiques des requêtes permet d'envisager la sélection du système ou modèle de recherche le plus adapté à la requête donnée.

Ce chapitre présente nos travaux sur ces deux thématiques.

8.1. Environnement

Afin de pouvoir contrôler librement tous les aspects de la recherche et de l'enrichissement des requêtes, nous avons utilisés notre propre système de recherche documentaire. Il s'agit là d'un système de type probabiliste, proche du système Okapi, qui a été implémenté par Pierre Jourlin. Il avait préalablement servi dans le contexte du "spoken document retrieval" (SDR, voir [Spärck Jones *et al.*, 2001]), et a été légèrement adapté à notre tâche.

Nous avons conduit les expériences sur la base de la piste ad-hoc de TREC 8, présentée dans le chapitre 4. Il s'agit là d'une collection de 50 requêtes et d'un corpus d'environ 500 000 documents.

8.2. Évaluation de performance

8.2.1. Pooling et évaluation automatique

Pour évaluer les performances de notre système avec les différents paramétrages possibles, nous nous sommes basés sur les jugements de pertinence issus des campagnes TREC. Utilisant la méthode du pooling (détaillé dans [Zobel, 1998]), des juges humains ont déterminé la pertinence d'un très grand nombre des documents trouvés par les systèmes participants pour chaque requête de la collection. Ces jugements ont ensuite été publiés sous forme d'une liste attribuant à tout document évalué pour une requête donnée une des deux catégories "pertinent" ou "non pertinent", connue sous le nom de *qrel*.

Il est démontré dans [Zobel, 1998] que ces *qrels* constituent une base adéquate pour l'évaluation automatique de systèmes n'ayant pas participé à la campagne TREC. Tous les résultats donnés dans ce chapitre sont ainsi basés sur ces jugements de pertinence, sachant que des documents ne figurant pas dans la liste sont considérés comme étant "non pertinents".

8.2.2. Mesures de performance d'un système de recherche documentaire

L'évaluation de la performance d'un système de recherche documentaire, ou la comparaison de deux systèmes, se divise en plusieurs étapes.

Le système fournissant pour chaque requête une liste de documents, il s'agit tout d'abord de déterminer la pertinence de chacun des documents, ce que nous avons fait à l'aide des *qrels*. Ensuite, on attribue une mesure de qualité au résultat pour chaque requête : nous avons choisi d'utiliser la précision moyenne, décrite dans le chapitre 3.2.

Le dernier pas consiste à attribuer une valeur globale de performance d'un système sur un ensemble de requêtes.

La mesure la plus utilisée, et qui est sans doute la plus simple et intuitive, est la moyenne arithmétique des *précisions moyennes* obtenue pour chaque requête, appelée *mean average*

precision (MAP). Ceci permet de résumer la qualité d'un système de recherche documentaire en un seul nombre, et cette mesure s'est ainsi imposée comme moyen de comparaison des performances de différents systèmes.

Ces dernières années un intérêt grandissant a été apporté aux requêtes particulièrement difficiles et les situations d'échec des systèmes de recherche documentaire. Ceci est spécialement reflété dans la création du *robust track* au sein des campagnes TREC en 2003 (voir [Voorhees, 2003]). Il a ainsi été constaté que l'optimisation des systèmes pour la mesure MAP se fait souvent au détriment des performances obtenues sur les requêtes difficiles (par exemple [Voorhees, 2005]).

En effet, cette mesure qui est utilisée de manière assez systématique pour décrire la performance d'un système de recherche, a tendance à donner du poids essentiellement aux requêtes performantes. Ainsi un système obtenant une précision moyenne de 0.8 pour une requête, comparé à 0.9 pour un système différent, et 0.1 contre 0.02 pour une autre requête, sera considéré comme moins performant que le deuxième système, avec une moyenne de 0.45 par rapport à 0.46. La différence de performance pour la première requête n'est pas forcément importante dans beaucoup d'applications, la qualité de la réponse fournie par les deux systèmes étant très bonne, alors que pour la deuxième il s'agit de la différence entre un résultat utilisable et un résultat inutilisable pour l'utilisateur.

Différentes mesures ont été proposées pour mieux représenter les performances de recherche sur les requêtes difficiles, par exemple le pourcentage des requêtes n'ayant aucun document pertinent parmi les 10 premiers documents de la liste ($P@10 = 0$). La plupart de ces mesures ne sont malheureusement pas suffisamment stables, surtout dans le cas de la piste ad-hoc avec seulement 50 requêtes disponibles.

Une mesure qui s'est avérée plus stable que les autres, tout en représentant bien l'importance des requêtes difficiles, est la moyenne géométrique des précisions moyennes, ou *geometric MAP*. On calcule la moyenne arithmétique des log des précisions moyennes, pour ensuite appliquer l'opération inverse, notamment l'exponentielle. Afin d'éviter des problèmes avec des requêtes ayant une précision moyenne de 0.0, la valeur 0.00001 est ajoutée avant de calculer le log, et ensuite soustraite du résultat après application de l'exponentielle.

Étant donné que nous nous intéressons dans ce chapitre particulièrement au cas où l'expansion de requête échoue, et que ceci est typiquement le cas pour les requêtes difficiles (comme nous allons le montrer dans la section 8.4), cette mesure est très intéressante en complément de la MAP classique.

8.3. Paramétrage de l'expansion et effet sur les performances

Nous avons brièvement étudié l'influence des paramètres D et t sur la précision moyenne obtenue avec différents types de requêtes. Quelques résultats apparaissent en figure 8.1. Ils sont assez similaires à ceux obtenus dans d'autres expérimentations (par ex : [Gauvain *et al.*, 1999]).

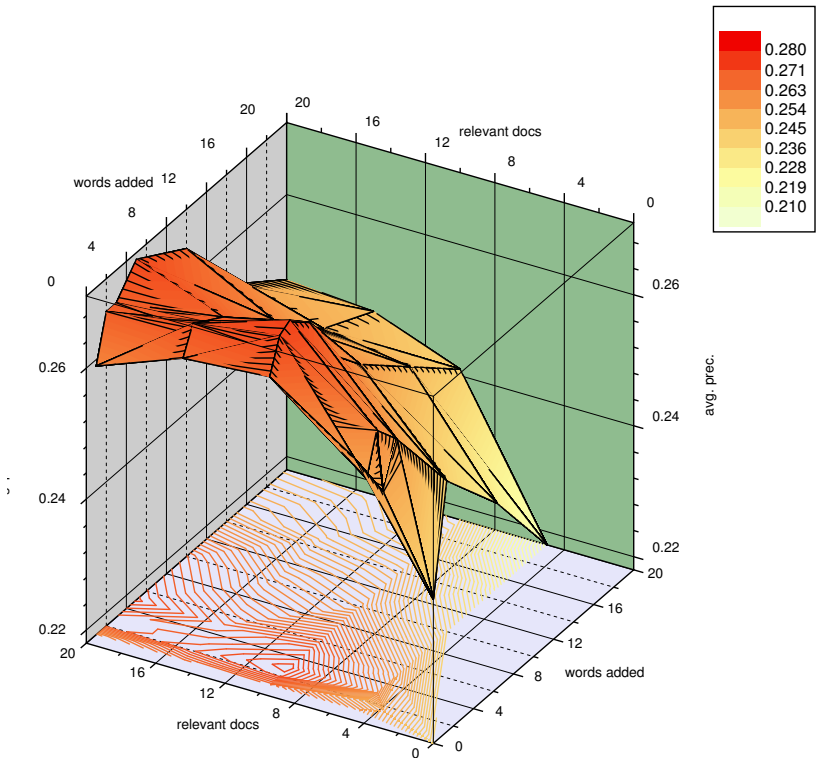

FIG. 8.1.: Précision Moyenne obtenue avec la méthode BRF, pour différentes valeurs de D et t (TREC 8 ad-hoc, titre + description)

Nos résultats montrent que, malgré une augmentation globale de la performance en terme de précision moyenne, l'utilisation de la méthode BRF dégrade les performances pour un pourcentage très important des requêtes (voir tableau 8.1).

Il est particulièrement intéressant de constater que l'effet de la BRF est différent selon que l'on s'intéresse à la mean average precision (MAP) ou à d'autres évaluations de performance. Par contre, contrairement à nos attentes, l'enrichissement a généralement un effet positif sur la gMAP, malgré les pertes constatées pour un certain nombre de requêtes difficiles. Le tableau 8.2 montre les performances obtenues avec différents paramétrages pour les requêtes de type T (titre), D (description), T+D (titre + description), et T+D+N (titre + description + narration).

	t	td	d	tdn
d : 5, t : 5	62%	38%	60%	32%
d : 5, t : 10	62%	46%	66%	40%
d : 5, t : 15	60%	50%	70%	40%
d : 5, t : 20	62%	54%	70%	48%
d : 10, t : 5	48%	40%	57%	28%
d : 10, t : 10	54%	46%	60%	28%
d : 10, t : 15	60%	48%	60%	32%
d : 10, t : 20	54%	50%	62%	42%
d : 15, t : 5	40%	34%	52%	36%
d : 15, t : 10	57%	36%	52%	40%
d : 15, t : 15	52%	42%	57%	44%
d : 15, t : 20	56%	42%	57%	50%
d : 20, t : 5	46%	36%	62%	40%
d : 20, t : 10	50%	42%	57%	40%
d : 20, t : 15	56%	40%	64%	48%
d : 20, t : 20	56%	42%	66%	48%

TAB. 8.1.: Pourcentage de requêtes pour lesquelles l'enrichissement par BRF (t ;D) est contre-productif : données TREC 8, requêtes de type T (titre), D (description), T+D (titre + description), et T+D+N (titre + description + narration).

8.4. Impact de la précision moyenne initiale

Nos travaux portant sur les deux aspects de la difficulté des requêtes posées et de l'effet qu'ont des techniques d'enrichissement selon la requête, nous avons également analysé le lien entre la précision moyenne obtenue (sans BRF) et le gain apporté par l'enrichissement de la requête.

Nos résultats confirment globalement les liens reconnus dans la littérature entre la qualité initiale du résultat de recherche et l'effet de méthodes automatiques d'expansion aveugle.

On constate dans la figure 8.2 que l'expansion de requête fonctionne surtout sur des requêtes pour lesquelles on obtient des performances moyennes sans BRF, alors que pour les requêtes qui donnaient des résultats très bons ou très mauvais l'enrichissement a plutôt tendance à dégrader le résultat. Cette observation est valable également sur les requêtes de TREC 7 et semble donc être générale.

On peut supposer que ceci s'explique principalement par le fait que l'enrichissement utilise des mots extraits des premiers documents dans la liste. Si ceux-ci ne sont pas pertinents, il n'est pas possible de trouver des termes représentatifs de la thématique recherchée, l'expansion de requête ne peut donc pas améliorer les performances. Dans le cas

	T		D		T+D		T+D+N	
	MAP	**gMAP**	**MAP**	**gMAP**	**MAP**	**gMAP**	**MAP**	**gMAP**
base	0.224	0.114	**0.202**	**0.078**	0.240	0.113	0.227	0.091
d: 5, t: 5	0.207	0.103	0.182	0.066	0.254	**0.161**	**0.256**	0.129
d: 5, t: 10	0.192	0.087	0.172	0.061	0.241	0.150	**0.257**	**0.131**
d: 5, t: 15	0.192	0.087	0.164	0.057	0.231	0.139	0.251	0.124
d: 5, t: 20	0.181	0.079	0.161	0.050	0.218	0.130	0.245	0.115
d: 10, t: 5	0.224	0.107	0.193	0.060	**0.268**	**0.154**	0.255	**0.130**
d: 10, t: 10	0.210	0.106	0.187	0.059	0.255	0.149	0.253	0.125
d: 10, t: 15	0.204	0.101	0.183	0.058	0.248	0.148	0.248	0.113
d: 10, t: 20	0.200	0.098	0.175	0.052	0.241	0.139	0.241	0.109
d: 15, t: 5	**0.234**	**0.130**	0.198	0.065	0.265	0.148	0.245	0.120
d: 15, t: 10	0.216	0.113	0.191	0.062	0.258	0.145	0.247	0.115
d: 15, t: 15	0.211	0.111	0.183	0.052	0.250	0.136	0.245	0.107
d: 15, t: 20	0.207	0.108	0.181	0.048	0.246	0.132	0.240	0.102
d: 20, t: 5	**0.227**	**0.126**	0.189	0.058	**0.270**	0.150	0.243	0.115
d: 20, t: 10	0.225	0.117	0.181	0.053	0.265	0.147	0.240	0.101
d: 20, t: 15	0.211	0.104	0.175	0.052	0.253	0.142	0.233	0.095
d: 20, t: 20	0.205	0.101	0.174	0.045	0.247	0.134	0.229	0.095

TAB. 8.2.: Performances moyennes en terme de MAP et gMAP selon le nombre de documents et de mots utilisés pour l'expansion des requêtes

contraire, les documents trouvés étant bons, une nouvelle recherche avec un classement différent des documents risque de déclasser les documents pertinents initialement trouvés et ainsi de détériorer le résultat.

8.5. Expansion sélective

Étant donné l'intérêt évident de l'expansion de requêtes pour augmenter les performances moyennes, mais également le problème de la dégradation des performances surtout pour des requêtes difficiles, nous nous sommes tournés vers l'application sélective de l'expansion sur la base du gain attendu pour chaque requête individuelle.

Afin de déterminer le potentiel d'une telle approche, nous avons d'abord évalué le gain possible supposant une décision parfaite pour chaque requête concernant l'utilité de l'application de BRF.

En particulier, sur les requêtes de taille moyenne (titre et descriptif) de TREC 8, notre système arrive à une moyenne (sur toutes les requêtes) des «précisions moyennes» de 24% sans expansion et 27% avec le meilleur paramétrage choisi globalement pour toutes les requêtes. Par contre, avec un simple choix binaire d'application ou non de BRF avec des paramètres fixés au préalable 29% sont possibles, en choisissant les meilleurs paramètres pour chaque requête on arrive à 33%.

En terme de gMAP, l'effet est encore plus net, allant de 11% sans expansion, 16% avec le meilleur paramétrage global (15% avec les paramètres optimisant la MAP), à 18% de gMAP avec décision.

Le tableau 8.3 montre les résultats potentiels, supposant une décision optimale, pour

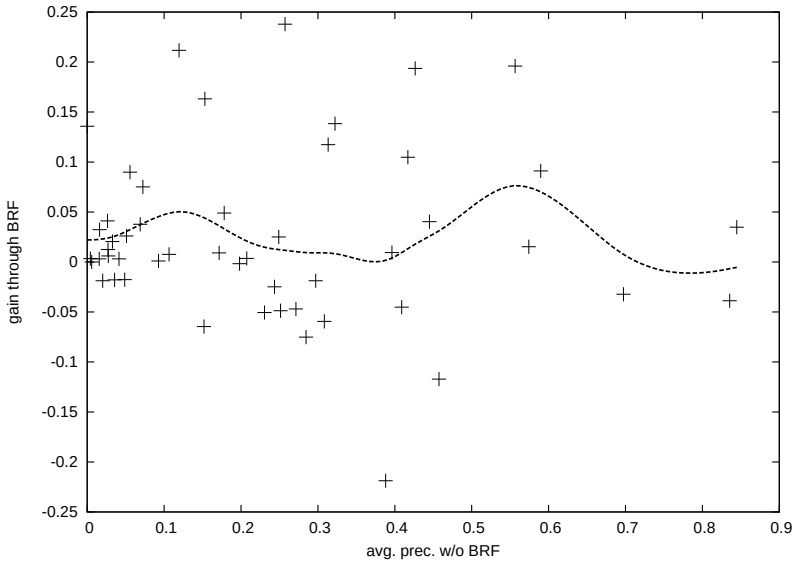

FIG. 8.2.: Précision moyenne et gain par BRF : TREC 8 ad-hoc

différents paramétrages de l'expansion. La décision est prise pour chaque requête entre la *baseline* sans expansion, et l'expansion de la requête avec les paramètres donnés.

	Standard		Sélectif	
	MAP	**gMAP**	**MAP**	**gMAP**
base	0.240	0.113		
d: 5, t: 5	0.254	**0.161**	0.276	0.176
d: 5, t: 10	0.241	0.150	0.276	0.178
d: 5, t: 15	0.231	0.139	0.275	0.176
d: 5, t: 20	0.218	0.130	0.272	0.172
d: 10, t: 5	**0.268**	**0.154**	**0.287**	0.176
d: 10, t: 10	0.255	0.149	0.283	0.177
d: 10, t: 15	0.248	0.148	0.283	0.178
d: 10, t: 20	0.241	0.139	0.283	0.177
d: 15, t: 5	0.265	0.148	0.283	0.171
d: 15, t: 10	0.258	0.145	0.284	0.176
d: 15, t: 15	0.250	0.136	0.284	0.176
d: 15, t: 20	0.246	0.132	0.284	0.178
d: 20, t: 5	**0.270**	0.150	0.284	0.175
d: 20, t: 10	0.265	0.147	**0.289**	**0.182**
d: 20, t: 15	0.253	0.142	0.286	**0.181**
d: 20, t: 20	0.247	0.134	0.285	0.178

TAB. 8.3.: Performances moyennes en terme de MAP et gMAP avec une application sélective optimale de l'expansion, requêtes de taille moyenne (titre + descriptif)

Il est donc possible (en principe) d'obtenir des performances considérablement supérieures aux meilleures performances obtenues sans décision pour chaque requête individuelle. Il est intéressant de noter que les meilleurs résultats avec décision ne sont pas forcément obtenus avec une configuration très bonne quand elle est appliquée systématiquement. Un choix de paramètres donnant de grands gains sur certaines requêtes, mais de

fortes dégradations pour d'autres, peut ainsi être un bon choix si on arrive à l'appliquer de manière sélective.

La décision sur le paramétrage précis de l'expansion à appliquer demande l'utilisation de classifieurs permettant une prédiction numérique des paramètres et suppose que l'estimation des différents paramètres est indépendante (qu'on peut donc estimer les valeurs optimales de d et t séparément).

Nous nous sommes pour cela contentés d'une décision binaire d'application ou non de l'expansion (avec un paramétrage fixé) selon les probabilités d'une amélioration ou dégradation des performances. Les méthodes utilisées sont essentiellement les mêmes que pour la prédiction de difficulté présentée dans le chapitre précédent, partant d'une représentation vectorielle des requêtes par un ensemble d'attributs pour ensuite apprendre des classifieurs de type SVM, arbre de décision, ou autres.

Pour la suite nous avons travaillé sur la base des requêtes de longueur moyenne (titre + descriptif), et nous sommes concentrés sur le choix entre la *baseline* sans expansion et l'expansion basée sur 20 documents, rajoutant 10 mots à la requête.

Cette configuration semble être la plus complémentaire, donnant les meilleurs résultats aussi bien en terme de MAP que de gMAP, à condition d'avoir un processus de décision fiable. Sans décision, les performances sont raisonnables, mais considérablement moins bonnes (surtout pour la gMAP) que d'autre paramétrages peut-être plus conservateurs, se basant sur moins de documents et ajoutant moins de mots à la requête.

8.6. Attributs et classifieurs utilisés

Étant donné le lien entre la précision initiale obtenue et l'effet de l'expansion, les attributs ayant servi à la prédiction de difficulté ont été retenus également pour l'estimation du gain par application de BRF.

Au delà de cet ensemble d'attributs, d'autres informations liées directement au processus d'expansion peuvent être utilisés. Ceci inclut les *offer weights* des termes ajoutés (de manière absolue, ou en relation avec les poids des termes de la requête initiale), la perturbation de la liste de documents trouvée pour la requête étendue par rapport au classement initial des documents, et différentes comparaisons entre les attributs calculés pour la requête initiale et ceux obtenus avec la version enrichie de la requête.

Pour la classification, nous avons entre autre utilisé les mêmes classifieurs que pour la prédiction de difficulté, mais nous avons également exploré d'autres variantes de classification.

En effet, dans le cas de la classification binaire en "facile" et "difficile" nous nous sommes intéressés à une mesure de précision de classification, forçant une séparation nette des deux classes. Les classifieurs appris sur la base de ces classes discrètes, nominales, devaient

donc optimiser la limite entre les deux classes, comme le font en particulier les SVMs qui se concentrent sur les exemples proche de l'hyperplan séparateur. Ce choix avait été fait pour faciliter l'évaluation des résultats obtenus, qui pouvait ainsi se résumer en un simple pourcentage de classification correcte.

Dans le cas de la décision sur l'application ou non de l'expansion, le pourcentage de décisions "correctes" n'est pas forcément essentiel. Une décision "fausse" dans un cas ou les performances avec ou sans BRF sont proches n'a pratiquement pas d'influence sur la performance du système, alors que ce sont les exemples ou l'impact de l'expansion est fort qui sont réellement importants. Plutôt que d'optimiser la séparation nette, on cherche donc à obtenir une classification fiable des cas extrêmes.

Nous avons pour cela représenté le gain (ou la perte) par l'enrichissement des requêtes de plusieurs manières : en deux classes nominales "g" et "p", par la différence entre la précision moyenne avec et sans expansion et, d'autre part, par la différence en terme de pourcentage de la précision moyenne initiale (afin de donner plus de poids au requêtes difficiles). Par conséquent, des classifieurs capables de faire des prédictions numériques ont dû être utilisés.

8.7. Résultats

Au niveau des attributs individuels on constate que plusieurs des attributs additionnels par rapport au chapitre précédent montrent une corrélation très significative avec la précision moyenne obtenue (aussi bien sans qu'avec BRF).

En particulier, le poid maximal d'un des termes de la requête initiale a une corrélation de rang positive de 0.62 (ce qui donne une *P-value* de $2.83E-6$) avec la précision moyenne sans BRF, ce qui est très fortement significatif. Plusieurs des autres attributs ajoutés montrent également une forte corrélation, ce qui les rendrait intéressant pour la prédiction de difficulté. Ils n'ont néanmoins pas été inclus dans le chapitre précédent, car ils n'étaient pas disponibles pour les systèmes traités.

La corrélation de rang des attributs avec l'effet de l'expansion est beaucoup moins nette, bien qu'un certain nombre d'attributs montrent une corrélation significative. Par exemple, le poid moyen des termes de la requête initiale a une corrélation de rang de -0.42, $p = 0.002$, alors que le seuil à partir duquel un résultat est considéré comme significatif est souvent défini comme $p = 0.05$.

Ceci permet d'avoir une première impression du potentiel des différents attributs, mais il ne nous semble pas utile d'accorder trop d'importance à ces mesures. Comme nous l'avons déjà exposé dans le chapitre précédent, la corrélation de rang permet de facilement quantifier la "valeur" d'un attribut, mais n'est pas toujours un bon indicateur du réel pouvoir de classification, surtout quand la corrélation n'est pas monotone.

Le tableau 8.4 résume les performances obtenues en terme de MAP et gMAP en appliquant l'expansion de manière sélective selon les prédictions faites par différents classifieurs.

	Classifieur	MAP	gMAP
	jamais	0.2399	0.1126
	toujours	0.2648	0.1470
nominal	NaiveBayes	0.2434	0.1363
	SMO	0.2549	0.1159
	J48	0.2522	0.1162
différence	LinearRegression	0.2574	0.1415
	SMOreg	0.2548	0.1388
	SimpleLinearRegression	0.2627	**0.1501**
	IBk	0.2544	**0.1509**
	M5	0.2648	0.1470
pourcentage	LinearRegression	0.2631	0.1448
	SMOreg	0.2571	0.1433
	SimpleLinearRegression	**0.2649**	**0.1471**
	IBk	0.2563	**0.1517**
	M5	0.2648	0.1470

TAB. 8.4.: Performances moyennes en terme de MAP et gMAP avec une application sélective de l'expansion basée sur la classification automatique, requêtes de taille moyenne (titre + descriptif)

On constate certaines améliorations par rapport à l'expansion systématique des requêtes, et plus encore par rapport à la version sans expansion. Par contre, certains paramétrages du BRF permettent d'obtenir de meilleurs résultats, ce qui remet en cause la décision d'utiliser les paramètres donnant le plus grand potentiel de performances supposant une décision parfaite. Étant donné l'impossibilité d'atteindre cet optimum, il est nécessaire de trouver un équilibre entre les bonnes performances de base et le potentiel d'amélioration par l'application sélective de l'enrichissement.

Les résultats sont néanmoins très encourageants et démontrent qu'une amélioration des performances est en principe possible par l'application sélective d'expansion de requêtes.

8.8. Sélection de modèle de recherche

Au delà du choix des paramètres pour un même système de recherche documentaire, il est possible d'utiliser nos méthodes pour sélectionner pour chaque requête le système susceptible de fournir les meilleurs résultats. Comme nous l'avons vu précédemment, les performances de différents systèmes de qualité comparable peuvent varier fortement selon les requêtes (voir graphique page 30).

Le tableau 8.5 montre le potentiel d'une sélection du modèle de recherche selon la requête à traiter pour une petit sous-ensemble des participants à TREC 8.

Malheureusement, nous n'arrivons pas actuellement, basé sur des prédictions réelles, à

MAP: décision oracle					
	ok8amxc	INQ603	ibms99a	Sab8A4	Flab8atd2
ok8amxc	*0.316872*				
INQ603	0.329528	*0.265888*			
ibms99a	0.34232	0.31694	*0.300508*		
Sab8A4	0.33289	0.297164	0.324226	*0.260798*	
Flab8atd2	**0.341444**	0.316798	**0.33384**	0.316922	*0.292986*

gMAP: décision oracle					
	ok8amxc	INQ603	ibms99a	Sab8A4	Flab8atd2
ok8amxc	*0.21339807*				
INQ603	0.23359835	*0.16872286*			
ibms99a	**0.2477065**	0.22593264	*0.20442724*		
Sab8A4	**0.2386158**	0.20399383	0.23798911	*0.16244688*	
Flab8atd2	0.2389081	0.2208621	0.23472848	0.22372333	*0.15481149*

TAB. 8.5.: Performances moyennes en terme de MAP et gMAP avec choix automatique (optimal) entre deux modèles disponibles pour chaque requête. La diagonale affiche les performances du système seul

améliorer les performances par rapport au meilleur système disponible.

9. EQUER : question/réponse en français

Sommaire

Une autre application est la tâche question/réponse, en particulier sur les données de la campagne Technolangue EQUER. Contrairement aux autres scénarios explorés, il s'agit ici d'un corpus en français.

D'autre part, le fait de se placer dans un contexte de question/réponse signifie que l'objectif et donc la notion de "difficulté" d'une question n'est pas le même que sur des application de recherche ad-hoc. Les informations disponibles sur lesquelles se base l'estimation de difficulté diffèrent également, en raison des différences de langue et du processus de recherche et réponse.

Il s'agit donc d'une évaluation complémentaire, permettant d'appréhender la robustesse et flexibilité de l'approche proposée. De plus, il s'agit d'un lien avec les travaux effectués par d'autres membres du LIA, qui permet de compléter le système question/réponse développé au sein du laboratoire.

9.1. Introduction

Les systèmes de questions-réponses (sQR) sont au cœur des préoccupations en recherche d'information. La campagne internationale TREC [1] inclut une tâche questions-réponses (QA) [2] depuis 1999, la campagne européenne CLEF [3] intègre la tâche QA@clef où les questions sont disponibles en plusieurs langues européennes, et enfin la campagne nationale Technolangue EVALDA [4] comporte le volet EQUER, auquel nous nous sommes plus particulièrement intéressés.

Les sQR fonctionnent habituellement selon une analyse modulaire : analyse de la question, recherche de documents pouvant contenir la réponse (moteur de recherche documentaire) puis analyse en profondeur des documents trouvés pour extraction de réponses. Les campagnes d'évaluation évoquées précédemment montrent que les systèmes ont encore beaucoup de mal à répondre à certaines questions. Hors de toute mesure quantitative liée à des campagnes d'évaluation, la prédiction de la capacité à répondre à une question posée (qu'elle soit liée au sQR lui même ou à l'absence de réponse dans le corpus) s'impose par un seuil d'admission de la question telle qu'elle est posée.

Dans le cas où on ne peut pas répondre, il faut entamer un processus de dialogue, dans la direction adaptée. Ceci peut comprendre un enrichissement (en cas d'ambiguité sémantique) ou une correction (il vaut mieux proposer différentes alternatives que proposer une réponse à une question automatiquement corrigée), ou encore une validation de la compréhension. Il y a alors deux problèmes qui se posent dont seul le second sera traité ici : d'une part quelle doit être la nature de ce dialogue et comment en exploiter la teneur et d'autre part à partir de quel critère initier ou non le dialogue ?

Il est donc particulièrement intéressant d'explorer la possibilité d'adapter les méthodes présentées dans les chapitres précédent à cette tâche.

9.2. Une méthode de prédiction appliquée à un sQR

9.2.1. Les scores de confiance en recherche documentaire

Nous avons adapté cette dernière méthode à la prédiction de la capacité à répondre dans le cadre de la campagne EQUER, avec le sQR développé au LIA [Gillard *et al.*, 2005].

Par rapport à l'évaluation de la qualité des réponses, la campagne TREC QA s'est intéressée en 2002 [Voorhees, 2002] à la capacité des systèmes à juger de la pertinence de leurs réponses. Pour cela, les participants n'avaient le droit de fournir qu'une seule réponse

[1] http://trec.nist.gov
[2] http://trec.nist.gov/data/qa.html
[3] http://www.clef-campaign.org
[4] http://www.elda.org/article118.html

courte par question, mais ces réponses devaient être ordonnées par confiance décroissante. Ainsi une mesure a complété le pourcentage de bonnes réponses et les classiques rappel/précision : *Confidence Weighted Score*. Elle calcule la moyenne du taux de bonnes réponses à tous les rangs dans le classement des réponses de chaque participant.

Pour répondre à cette problématique, la plupart des systèmes se basent sur un consensus entre plusieurs propositions de réponses faites par différentes parties du système, notamment lors de l'utilisation de ressources externes telles que des bases de connaissances [Chu-Carroll *et al.*, 2002] ou [Bellot *et al.*, 2002]. Notre système utilise également d'autres critères dans le cas de consensus, appris de manière empirique sur le comportement du système (critère sur le type de réponse attendue), ou utilisant directement des scores de recherche d'informations internes au sQR.

9.2.2. Les spécificités d'un sQR

On distingue deux grandes catégories de sQR : ceux qui reposent uniquement sur une base de règles et ceux reposant sur des méthodes stochastiques qui utilisent des règles pour l'étiquetage sémantique. Les premiers utilisent généralement des patrons de reformulation des questions afin de rechercher directement les réponses dans leur forme attendue. Dans ce cas là, la détermination de la capacité du système à répondre dépend principalement de sa capacité à reformuler, et de la présence des reformulations dans les documents. Nous nous sommes donc intéressés à la seconde catégorie de systèmes. La figure 9.1 illustre le fonctionnent général de ces sQR, et plus particulièrement de celui que nous utilisons [Gillard *et al.*, 2005], qui est modularisé comme la plupart des sQR aujourd'hui. Les données sur fond gris sont celles qui ont été transmises au système de prédiction pour l'apprentissage.

Les principales étapes de traitement sont l'analyse de la question, la recherche de documents pouvant contenir la réponse (moteur de recherche documentaire), puis une analyse en profondeur des documents trouvés pour extraction de réponses.

La plupart des sQR à base de méthodes stochastiques utilisent différents scores, notamment au niveau de l'appariement. De plus, tous utilisent une étape de focalisation à l'intérieur des documents pour cibler la ou les phrases contenant la réponse potentielle.

9.3. Expériences sur les données de la campagne EQUER avec différentes méthodes de prédiction

9.3.1. Le moteur LIA-QA et la campagne EQUER

Lors de la campagne EQUER, pour la partie générale, les participants disposaient des 500 questions classées selon des types généraux : 30 questions pour les catégories booléennes,

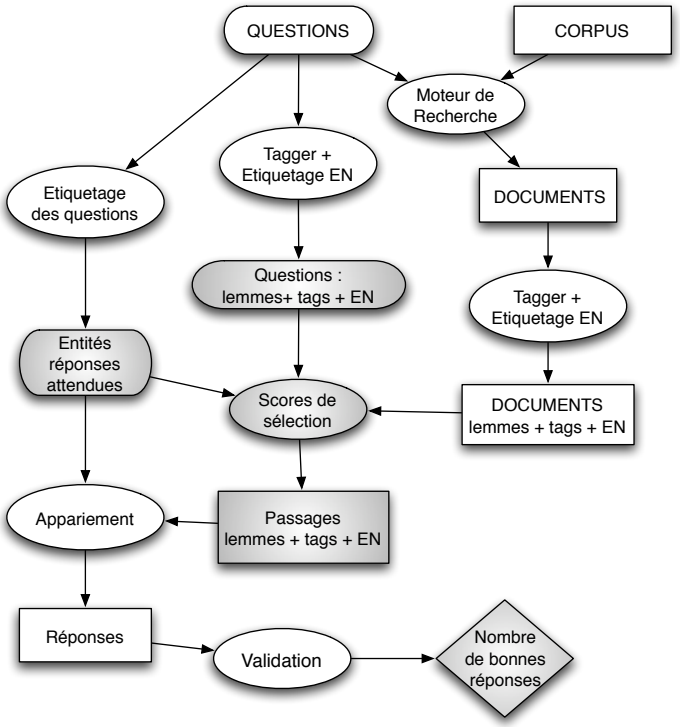

FIG. 9.1.: Fonctionnement général des sQR, en grisé les données utilisées par le système de prédiction de difficulté

listes, définitoire, et 407 questions factuelles.

Etant donné que l'objet de la campagne n'était pas la recherche documentaire, les résultats d'un moteur de recherche (Pertimm) étaient fournis aux candidats. Pour chaque question les participants disposaient des 100 premiers documents trouvés. Ces résultats étaient ordonnés mais les participants ne disposaient pas des scores.

Pour chaque question un système pouvait retourner jusqu'à 5 réponses. Les réponses ont été validées par un ou deux juges humains.

Au niveau des classifieurs du système de prédiction, les informations utilisées sont issues d'étapes de traitement du moteur LIA-QA, décrit dans [Gillard *et al.*, 2005], et dont la figure 9.1 illustre le principe global.

L'étiquetage morpho-syntaxique est effectué à l'aide du Tree Tagger [Schmid, 1994]. Celui en entités nommées s'appuie sur les étiquettes obtenues. Ces étiquetages sont appliqués à la fois aux questions et aux documents sélectionnés par le moteur de recherche. D'autre part, des automates permettent de déterminer pour chaque question quel type de réponse est attendue (cet étiquetage est décrit plus loin). Ensuite, au plus 1 000 extraits de 3 phrases issues des documents sont sélectionnés, en fonction d'un score de densité. Celui-ci prend en compte les mots de la question, les types d'entités nommées rencontrées et le type de

réponse attendu. Enfin, l'appariement est fait par le calcul d'un score de compacité, entre les étiquettes des termes contenues dans les extraits et celle attendue par la réponse.

9.3.2. Classification à partir des étiquettes de type de réponse attendue

Le type de réponse attendue a été utilisé par un certain nombre de systèmes tels que celui de [de Loupy et Bellot, 2000] lorsque les systèmes ont un fonctionnement dédié. Nous avons donc tenté dans cette section de déterminer si cette indication peut être un critère, sinon déterminatif, au moins prépondérant dans l'évaluation de la difficulté des questions.

Le composant d'étiquetage en types de réponses attendues de LIA-QA dédié à l'appariement est un étiquetage hiérarchique des questions. La hiérarchie utilisée a été inspirée par celle proposée par [Sekine *et al.*, 2002], dont elle est un sous-ensemble. Le choix de ce sous-ensemble, qui comprend 100 étiquettes, a été fait selon une observation de la fréquence des questions associées à une entrée de cette hiérarchie lors des précédentes campagnes d'évaluation QR de CLEF (2003 et 2004). Concrètement, cette phase d'étiquetage se déroule après une première étape d'uniformisation, à base de règles et de lexiques, permettant de réduire les différentes variantes à une même écriture et ainsi de diminuer le nombre de règles d'étiquetages à 172.

Afin de pouvoir utiliser nos résultats sur la classification à partir des étiquettes, nous avons fait une vérification manuelle de l'étiquetage en type de réponses attendues, dont voici l'analyse. Sur les 407 questions factuelles il y en a eu 52 non étiquetées dont :

- 13 étaient du type *procédé* , introduites par l'interrogatif *comment*
- 8 étaient du type *procédé* exprimées différemment (par quel procédé, de quelle façon, ...)
- 7 concernaient des événements
- 3 étaient des "pourquoi"
- 2 étaient floues en termes d'attente de réponse : "qu'en est il de ... ?" ou "que devient ... ?"

Les 19 autres questions se répartissent en 7 questions qui auraient dû être étiquetées par le système, et 12 dont les types n'étaient pas attendus même si ils se trouvent dans la hiérarchie des entités nommées de [Sekine *et al.*, 2002]. Ce sont des entités de type maladie, nourriture, animal, etc.

Parmi les 355 étiquettes de questions proposées, seulement 10 étaient erronées. Parmi elles 3 étaient des numérations d'une mauvaise cible, et 3 n'auraient pas dû être étiquetées car elles sont de type *procédé*.

Le tableau 9.1 recense pour certains types de réponses attendues le pourcentage de bonnes réponses fournies par le LIA-QA. On constate que, à part pour certains types très peu représentés, comme les films, les causes de decès ou les fonctions par exemple, la plupart des types conduisent à l'obtention d'une bonne réponse dans environ 57% des cas.

TYPE	N	OK	% R
Lieu	76	43	**56,6%**
nationalite	12	9	75%
ville	10	8	80%
Nombres	81	48	**59,3%**
argent	6	4	66,7%
longueur	10	6	60%
employés	4	1	25%
population	7	6	85,7%
personnes	10	7	70%
age	7	3	42,9%
Fonction	8	1	12,5%

TYPE	N	OK	% R
Personne	102	62	**60,8%**
president	9	6	66,7%
écrivain	7	5	71,4%
Organisation	18	8	**44,4%**
parti politique	8	7	87,5%
entreprise	2	1	50%
Date	41	24	**58,5%**
Journal	4	3	75%
Cause de décès	4	3	75%
Film	3	0	0%

TAB. 9.1.: Résultats des questions factuelles par type de réponse attendue : N est le nombre de questions étiquetées par le type, OK est le nombre de questions pour lesquelles il y a eu au moins une bonne réponse fournie, et %R le pourcentage de ces questions.

Cela montre qu'on ne peut pas utiliser uniquement le type de réponse attendue pour déterminer a priori si on sera capable de répondre à la question. On ne peut donc pas déterminer la difficulté d'une question, pour notre moteur en tous cas, en fonction de l'objet de la question. Cela dit, même si cette caractéristique des questions n'est pas suffisante comme seul prédicteur, elle peut être utilisée par les classifieurs dans les expériences suivantes, et elle peut tout à fait contribuer à une bonne classification. Il est à noter que pour toutes les questions où l'étiquette n'a pas été déteminée, le système n'a pas fourni de réponse. C'est donc le seul critère que nous avons pu dégager à l'aide de l'étiquetage en type de réponse attendue.

9.3.3. La prédiction par classification

Nous avons ensuite utilisé les méthodes présentées dans les chapitres précédents dans ce nouveaux contexte d'application.

Un certain nombre de caractéristiques de la question ont été utilisées. Les caractéristiques portant sur l'ambiguité des termes, dévoilées habituellement par le nombre de synonymes, le nombre de sens connus, ou les hyponymes, n'ont pas été utilisées dans ces premières expériences. Ces informations, issues de wordnet pour l'anglais, ne sont pas facilement disponibles pour la langue française.

Ceci a considérablement limité les caractéristiques utilisables pour cette application, mais cet effet est compensé en partie par d'autres informations additionnelles exploitables.

9.3.3.1. La question

Indépendamment de la réponse du système QR, la question peut être caractérisée par un certain nombre d'attributs, en partie issus du prétraitement de la question.

Hors la longueur de la question et différentes mesures basées sur la fréquence des termes de la question (IDF maximale, IDF moyenne, etc.), nous prenons également en compte l'objet de la question (le type de réponse attendu).

Étant donné que les questions dans une tâche QR montrent plus d'homogénéité que dans des applications ad-hoc plus générales, il est également intéressant d'exploiter les mots qui se retrouvent dans un grand nombre de questions et qui reflètent souvent la nature de la question. Dans nos expériences, parmi les mots déterminés par le processus d'apprentissage d'arbres de décision comme étant discriminants se trouvent par exemple "Combien" (en début de phrase) ou "quel" (pas en début de phrase, comme l'indique la minuscule). Nous n'avons pas pour l'instant utilisé de SCTs (l'intégration avec les autres attributs étant plus compliquée), mais ce type de classifieur devrait pouvoir livrer des résultats intéressants dans cette application.

Les étiquettes morpho-syntaxiques et les entités nommées sont utilisées comme indicateurs de la complexité structurelle de la question.

9.3.3.2. Similarité cosinus

Comme pour les autres applications présentées, nous avons utilisé la similarité cosine entre les résultats proposés par le système, caractéristique qui porte sur la cohésion lexicale des réponses. Une différence par rapport aux applications ad-hoc est que les résultats sont basés sur des extraits assez courts, plutôt que des documents entiers. Ceci a, bien évidemment, une grande influence sur des mesures de cohésion, par le simple fait qu'un nombre plus faible de mot dans chaque segment conduit à un recoupement du vocabulaire entre les segments très différent de ce qu'on constate sur des documents longs.

Nous avons considéré aussi bien la similarité entre les documents issus du moteurs de recherche d'où sont extraits les passages, que la cohésion entre les passages eux-mêmes. Cette idée part du principe qu'une forte variabilité linguistique est corrélée avec un plus grand risque qu'une partie des documents ou passages utilisés soient non pertinents par rapport à la question, et conduisent à des réponses fausses.

La similarité cosinus a été calculée pour chaque question entre les 5, 10, 15 ou 20 premiers passages de documents ou documents complets sélectionnés par le système pour extraire les réponses. La similarité est calculée pour chaque paire de documents ou d'extraits en fonction des termes présents dans chacun d'eux et d'un poids qui leur est attribué pour chaque terme Ti. Ce poids dépend du nombre d'occurrences du terme Ti dans l'extrait ou dans le document (tf_i), et de la fréquence des lemmes dans la totalité des $|D|$ documents

du corpus de la campagne (df_i). Le poids w_{ij} d'un terme Ti dans le document Dj est donné par :

$w_{ij} = tf_{ij} * log|D|/df_i$

L'autre mesure de cohésion utilisée dans le système de prédiction sur la recherche documentaire est l'entropie, qu'on peut calculer sur des ensembles de documents à l'aide d'un modèle de langage. Dans l'optique de pouvoir utiliser notre système en tant qu'aide à l'utilisateur, nous avons choisi de ne pas calculer cette caractéristique, très gourmande en calculs. C'est pour la même raison que nous avons limité le calcul des similarités moyennes entre les paires de documents aux 20 premiers document.

9.3.3.3. Scores des extraits

Sur des recherches ad-hoc, les scores de similarités entre les documents et la requête sont généralement utilisés pour le classement des documents, et non pas pour une évaluation dans l'absolu de leur pertinence. Différentes transformations ont été testées sur ces scores afin d'en déterminer une corrélation avec la capacité à répondre. Il s'agit par exemple de la différence entre le score du premier document et le score du nième, ou bien une moyenne des scores des n premiers documents retournés.

Pour l'adaptation de cette mesure au problème des questions-réponses, nous avions uniquement les scores de densité de chaque extrait de document par rapport à la question. Il serait utile d'avoir également des informations sur les documents entiers trouvés dans la première phase de la recherche, mais ces scores n'étaient pas disponibles.

9.3.4. Prédiction par arbres de classification et SVM

Les expériences ont été menées avec les arbres de classification et les SVM implémentés au sein de la boite à outils WEKA [Witten et Frank, 1999].

Les classifieurs ont utilisé principalement trois classes de *features* :

- les questions, ainsi que leurs lemmes filtrés ou non, les étiquettes morpho-syntaxiques et sémantiques des mots qu'elles contiennent, ainsi que le type de réponse attendue. Sur ces données sont utilisées des mesures numériques et qualitatives ;
- les documents issus de la recherche effectuée par le moteur Pertimm dans le cadre de la campagne EQUER et les passages de 3 phrases sur lesquelles s'appuie l'appariement, le tout au format lemmatisé et filtré. Les scores de similarités entre les 5, 10, 15 et 20 premiers sont calculés et utilisés comme attributs ;
- les scores de densité qui ont permis de sélectionner les passages. Des mesures de moyenne sur les 5 à 50 premiers, ou d'écart type, ou la valeur à un rang donné sont obtenues sur ces scores de densité.

Enfin, les classifieurs disposaient pour chaque question du nombre de réponses courtes exactes retournées par le système (entre 0 et 5), correspondant aux deux classes, un nombre de réponses correctes supérieur à 0 pour la classe "facile", ou un nombre de réponses correctes nul pour la classe "difficile".

Les travaux en recherche documentaire sur la prédiction des questions s'évaluent souvent à partir de la distance de Kendall's tau, lorsqu'on dispose de scores liés à la prédiction. On mesure alors la distance entre le classement des réponses selon leur score de difficulté par rapport au classement selon le score de précision qu'elles ont obtenu. D'autres travaux dans un domaine où on peut déterminer si le résultat est vrai ou faux, comme les questions-réponses, utilisent pour se positionner des mesures de type CWS, qui de la même manière se fondent sur un classement de scores de prédiction.

Nous ne nous sommes pas basés dans notre travail sur ce type de mesure, car nous avons cherché à déterminer de manière binaire la capacité d'un système à répondre à une question. Les résultats sont donc exprimés en pourcentage de bonne prédiction, sachant qu'une question est considérée "facile" si elle a eu au moins une réponse exacte parmi les cinq propositions.

Comme il y a un nombre relativement réduit d'exemples (de questions), l'évaluation des classifications par arbres de décision et SVM est faite par une validation croisée avec 10 plis (10-fold cross-validation). De plus, nous proposons ici les résultats sur les questions factuelles uniquement, car les autres étaient présentes en quantité trop peu représentative (30 items) pour fournir des résultats cohérents. Le tableau 9.2 montre les résultats des deux classifieurs, avec les différentes classes de *features* utilisées. La dernière ligne correspond à une prédiction uniquement basée sur la distribution, qui peut donc être considérée comme la prédiction de référence.

feature	SVM	Arbres de Décision
toutes	68,5%	62,4%
questions	**69%**	64,1%
documents	53,8%	49,9%
passages	52,3%	50,1%
aucune	**51,6%**	

TAB. 9.2.: Résultats de la classification automatique, en pourcentage de bonnes prédictions

Les résultats montrent que l'utilisation uniquement des données relatives aux questions, quelle que soit la méthode de classification, est très efficace et suffisante. De plus la classification avec les SVM donne de très bons résultats bien supérieurs à la référence. Une des raisons possibles au succès de la classification dans notre expérience est qu'il y a dans l'apprentissage à peu près autant d'exemples positifs que d'exemples négatifs, le pourcentage de bonnes réponses obtenues dans la catégorie des questions factuelles avoisinant les 50%.

Les arbres de décision ont fourni des résultats moins bons que les SVM, mais en observant leur composition nous nous sommes aperçus qu'il y avait beaucoup de surapprentissage. On pourra à l'avenir optimiser cette classification en réduisant le nombre d'attributs et la taille des feuilles. La figure 9.2 montre un exemple d'un tel arbre, avec 8 feuilles qui prédisent si la réponse sera bonne ou mauvaise. Entre crochets est indiqué le nombre de questions correspondant effectivement à la prédiction, puis le nombre de mauvaises prédictions. Le paramétrage qui a permis d'obtenir cet arbre aboutit à 63,15% de prédiction correcte sur les questions factuelles avec toutes les classes de features disponibles en 10-fold cross-validation, et 73,71% sur le corpus d'apprentissage. L'inconvénient majeur de ce type d'optimisations est que dans ce cas le système pourrait être trop adapté aux données de EQUER et aux résultats de LIA-QA.

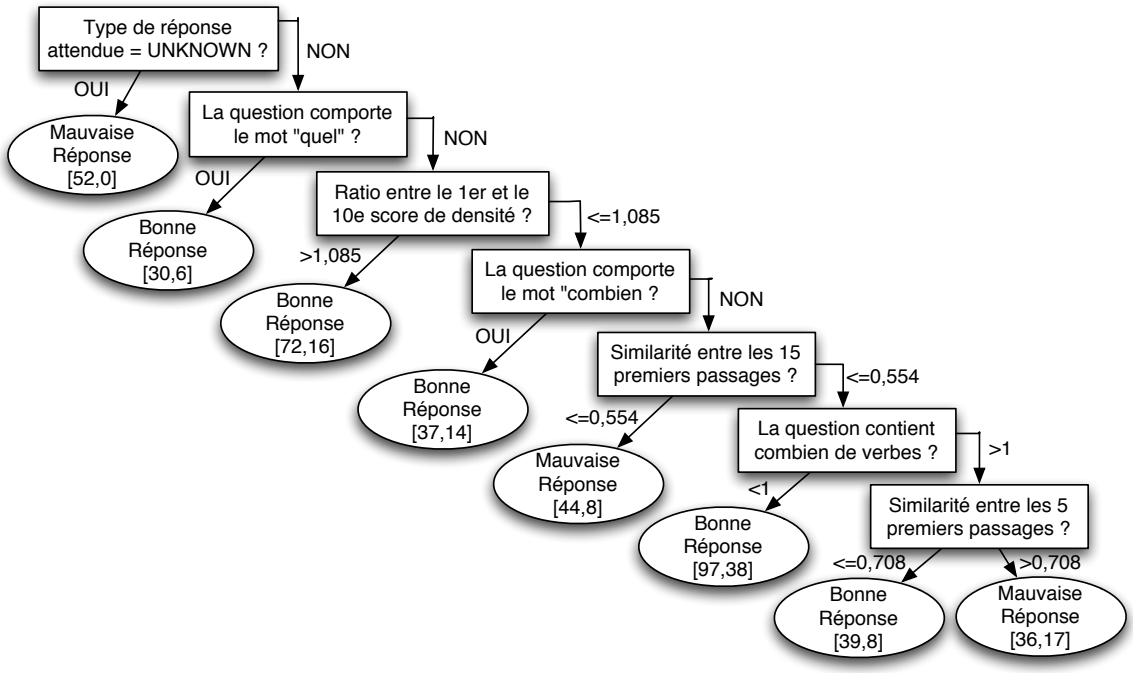

FIG. 9.2.: Un exemple d'arbre de décision optimisé. Les valeurs indiquées dans les feuilles sont respectivement le nombre de questions correctement classées par ce critère et le nombre de mauvaises prédictions.

9.4. Conclusion et perspectives

Les deux approches que nous avons mises en place pour la prédiction de la capacité d'un système à fournir une bonne réponse à une question factuelle ont montré que l'élément le plus significatif à prendre en compte était la question elle même, et non pas les documents

utilisés pour trouver la réponse ou encore le type de réponse attendue. Cela est plutôt une bonne chose car cela montre que avant même d'activer le moteur de questions-réponses on peut obtenir une très bonne indication sur la nécessité de la préciser ou de reformuler la question. De plus, les résultats obtenus avec les SVM sont très encourageants, et ils pourront probablement encore être améliorés par l'intégration de nouvelles ressources.

En effet nous avons très peu adapté la classification au problème particulier des questions-réponses, afin de vérifier dans un premier temps si les solutions pouvaient coïncider avec celles des systèmes de recherche d'informations ad-hoc. L'intégration du taux d'ambiguité dans les critères de prédiction à l'aide d'un réseau de collocations ou du réseau de connaissances sémantique EuroWordNet devra être la prochaine étape de cette étude. En effet les questions contiennent généralement peu de mots, ce qui amène un plus grand risque d'ambiguité des termes.

La difficulté des questions est un problème qui a été soulevé essentiellement du point de vue de la classification par types de réponses attendues. L'autre aspect qui peut rendre une question difficile est la nécessité d'un traitement de la négation ou de la voix passive [Lavenus et Lapalme, 2002]. L'intégration de ce type de critères syntaxiques devra à terme être prise en compte. En effet, même si ces problèmes là ne se rencontrent pas spécifiquement dans la campagne EQUER, ils correspondent à un besoin dans l'utilisation réelle de ces systèmes.

D'autre part, les cas d'utilisation réels amènent à s'interroger sur le cas des erreurs d'orthographe, notamment sur les noms propres (Hongrie –> Ongrie), ce qui en fait des mots absents du corpus et ne permet pas de retrouver les réponses sans traitement spécifique. Des erreurs sur les autres mots de la requête peuvent également avoir une influence importante sur la capacité à répondre. On pourrait alors imaginer un score orthographique qui selon le type d'erreurs pourraient prédire la capacité du système à passer outre.

10. Conclusions et Perspectives

Les résultats obtenus sur la problématique de la prédiction de difficulté des requêtes nous paraissent, malgré des imperfections évidentes, comparables voire supérieurs à ceux qu'on peut trouver dans les publications actuelles du domaine.

Notre travail se situe dans le contexte d'un intérêt grandissant pour ce problème et contribue au développement récent de cette thématique peu abordée au commencement de cette thèse. Le nombre croissant de publications, en particulier pendant ces deux dernières années, témoigne de l'importance de cette tâche qui est encore bien loin d'être pleinement résolue.

Dans le domaine de la décision automatique pour l'application sélective de techniques d'enrichissement, les résultats actuels sont très encourageants et présagent d'un grand potentiel pour cette approche.

La grande généricité de nos méthodes permet de les adapter facilement à différentes tâches et environnements, ce qui est démontré entre autres par nos expériences dans des applications de questions-réponses en français sur la base des campagnes EQUER.

Une des tâches que nous abordons actuellement est le classement par confiance des réponses à un ensemble de requêtes ou questions, comme cela est demandé dans les pistes *robust* et *QA* de TREC. Ces pistes comprennent de plus grandes collections de requêtes que le *adhoc* et peuvent donc être plus adaptées pour l'apprentissage de classifieurs donnant des prédictions numériques.

Il est relativement facile d'étendre notre méthode et d'y intégrer toutes sortes d'informations supplémentaires, selon la disponibilité dans l'application donnée. Nous profitons ainsi des informations spécifiques à l'expansion des requêtes dans le cadre du chapitre 8, en particulier les poids attribués au termes de la requête, alors que le système question-réponses utilisé dans le chapitre 9 nous fournit déjà une catégorisation en types de réponses attendus ainsi que des informations grammaticales sur les questions posées.

Il nous semble particulièrement intéressant de complémenter notre approche largement statistique par des informations linguistiques plus détaillées. En particulier, les analyses proposées dans [Lavenus *et al.*, 2004] et [Mothe et Tanguy, 2005] se prêtent à une intégration dans notre système, de même que d'autres méthodes fournissant des informations quantifiables de manière numérique ou en termes d'appartenance à des classes nommées.

Troisième partie .

Annexes

A. L'utilisation d'Oracle pour la recherche documentaire en langage naturel – Une approche à base de reformulation automatique des requêtes

Sommaire

Dans le cadre d'une bourse cofinancée par la Région PACA et un partenaire industriel, une partie du travail de cette thèse a été effectuée en collaboration avec l'entreprise Digitech SA. Ce chapitre présente une étude menée dans ce contexte et qui a été publiée à la conférence SIGIR 2005 [Grivolla, 2005] :

Dans des applications d'entreprise, souvent de grandes quantités de données sont stockées dans des systèmes de bases de données, tels Oracle. En dehors d'informations structurées, ceci peut inclure des documents texte qui ne peuvent pas être facilement retrouvés à l'aide de requêtes SQL traditionnelles.

Oracle fournit des moyens pour traiter le problème de la recherche "plein texte" sous le nom d'*Oracle Text*, dans la forme d'opérateurs particuliers permettant la recherche à l'intérieur de la partie texte d'une entrée dans la base. Nous avons exploré l'effet de ces différents opérateurs pour des requêtes dérivées de requêtes en langage naturel. Nous allons dans ce chapitre comparer les performances de recherche obtenues avec différentes reformulations à partir du langage naturel vers des requêtes SQL Oracle.

A.1. Introduction

Quand on s'intéresse à de grandes quantités de texte, archivées dans une base de donnée d'entreprise, il est souvent intéressant de pouvoir y accéder par une interface utilisant le langage naturel pour rechercher de l'information. Des requêtes structurées et rigides comme celles formulées en SQL ne sont pas particulièrement adaptées à la recherche dans des données non structurées, et de plus leur complexité peut être un obstacle important pour les utilisateurs potentiels.

Néanmoins, il n'est souvent pas praticable d'utiliser un système spécialement conçu pour la recherche en langage naturel, tel que ceux participant aux campagnes TREC, en plus du système de base de données structuré existant.

Oracle adresse ce besoin en incluant des méthodes de recherche plein texte dans leur produit de base de données. Le potentiel de ces méthodes a été démontré par la participation d'une équipe d'Oracle à la campagne d'évaluation TREC 8 ([Voorhees et Harman, 1999]) où ils ont pu obtenir de bonnes performances de recherche ([Mahesh *et al.*, 1999]). Malheureusement, ces résultats ont été obtenus avec des requêtes formulées manuellement sur la base des requêtes fournies dans le cadre de la campagne, et montrent donc la meilleure utilisation des opérateurs disponibles dans Oracle par une équipe d'utilisateurs experts.

Afin de rendre une partie de ce potentiel accessible à des utilisateurs non experts de la base de données, nous nous sommes penchés sur différents moyens de traduire de manière entièrement automatique des requêtes en langage naturel (anglais en l'occurrence) vers des expressions SQL interprétables par la base Oracle.

A.2. Motivations et approche

Pour la recherche plein texte, Oracle permet l'utilisation d'expressions SQL de la forme suivante :

```
select * from TABLE where contains(TEXTCOLUMN, <query>, 1) > 0 order by
score(1) desc ;
```

Ceci renvoie tous les documents (lignes de la table `TABLE`) pour lesquels le score obtenu en comparant la requête au contenu de la colonne `TEXTCOLUMN` est positif.

Un certain nombre d'opérateurs sont disponibles pour la construction de la requête, en particulier les suivants :

`ABOUT` recherche des *thèmes*, en utilisant la base de connaissances fournie

`ACCUMulate (,)` classe par score cumulé pour tous les termes

`AND (&)` : tous les termes doivent être présents

`OR (|)` : au moins un des termes doit être présent

`NEAR (;)` classe par proximité des termes

`stem ($)` utilise du stemming ou de la lemmatisation (cherche tous les termes avec la même racine)

La soumission de l'équipe Oracle à TREC 8 montre qu'il est possible d'obtenir des performances très intéressantes pour la recherche documentaire en langage naturel en utilisant les outils fournis par *Oracle InterMedia* ou *Oracle Text*.

En même temps, les exemples de requêtes reformulées donnés dans l'article montrent que ces résultats ont été obtenus en combinant plusieurs des opérateurs disponibles, choisissant soigneusement des termes et phrases appropriées pour la recherche, et en leur assignant manuellement des poids.

La requête 414 (Cuba, sugar, exports) a ainsi été transformée en :
`ABOUT(cuba) * 3, ABOUT(sugar) * 5, ABOUT(Russia) * 2, (ABOUT(export) or ABOUT(import))`

La requête utilisée pour le sujet 450 (King Hussein, peace) était encore plus complexe :
`(King Hussein=Hussayn=Husayn=Husyan * 5, ABOUT(peace talks) * 3, peace * 2, ABOUT(Israel)) MINUS (Iraq or ABOUT(Iraq))`

Nous avons utilisé le même corpus de requêtes et documents issu de TREC 8 et avons généré plusieurs ensembles de requêtes, en appliquant différentes transformations afin d'obtenir des expressions SQL valides.

A.2.1. Reformulations

Nous avons commencé en utilisant différents opérateurs dans leur forme de base, avant d'explorer des combinaisons de plusieurs opérateurs. Nous avons enfin rajouté une pondération des termes de la requête selon leur fréquence dans la collection de documents (extraite directement de la base de données), de manière similaire à la pondération $TF.IDF$ classique dans le modèle vectoriel [Salton, 1991].

Une requête telle que (requête 401),
`What language and cultural differences impede the integration of foreign minorities in Germany ?`
et ainsi convertie en différentes formes, par exemple :
`$What * 2.48, $language * 2.96, $cultural * 4.82, $differences * 4.79, $impede * 7.37, $integration * 5.28, $foreign * 2.83, $minorities * 6.07, $Germany * 3.87` (avec lemmatisation et pondération additionnelle des termes)

Oracle donne quelques exemples de transformations du langage naturel vers des requêtes *Oracle Text* pour leur application *Ultrasearch*, qui combinent en ordre décroissant d'importance la recherche de la phrase exacte, des requêtes `NEAR`, et finalement des requêtes `ACCUM`.

transformation type	mean avg. prec. medium	short
ACCUM	7%	16%
AND	1%	13%
NEAR	0%	14%
OR	1%	4%
stemmed	7%	17%
stemmed+weighting	12%	19%
about	16%	19%
about+weighting	13%	17%
about (separate terms)	10%	15%
about (expanded query)	7%	14%
about+stemmed	7%	20%
about+stemmed+weighting	13%	21%
ultrasearch	7%	17%

TAB. A.1.: Moyenne des précisions moyennes (MAP) obtenues pour des requêtes courtes (*titre*) et moyennes (*titre et descriptif*)

A.3. Résultats expérimentaux

Nous avons testé une sélection de méthodes de reformulation sur les différents ensembles de requêtes en langage naturel fournis pour la campagne TREC 8. Ceci comprend des requêtes très courtes (titre uniquement), qui consistent en quelques mot-clés, des requêtes de longueur moyenne sous forme de phrases entières (descriptif), des requêtes combinées titre et descriptif, ainsi que des requêtes longues des plusieurs phrases (titre, descriptif et narratif).

Le tableau A.1 montre la précision moyenne obtenue par ces différents ensembles de requêtes.

L'équipe Oracle avait obtenu une MAP d'environ 47% à leur participation à TREC 8, ce qui est largement supérieur aux performances même des meilleurs systèmes automatiques, et une bonne performance pour un système "manuel". Nos résultats devront plutôt être comparés à ceux obtenus par d'autres systèmes entièrement automatiques, qui atteignent généralement (pour de bons systèmes) environ 25 à 30% de moyenne de précision moyenne.

On peut constater que des requêtes `ABOUT` basiques procurent des résultats relativement acceptables, même si les performances restent en dessous de celles de systèmes développés spécifiquement pour des requêtes de type TREC. La simple accumulation des termes de la requête (avec ou sans lemmatisation) donne des résultats considérablement plus faibles, en particulier pour des requêtes plus longues.

Le rajout d'une pondération additionnelle des termes (Oracle utilise déjà en interne une forme de pondération basée sur les fréquences dans les documents) augmente considéra-

blement les performances pour des requêtes de type ACCUM.

Néanmoins, ceci n'est pas facilement reproduit pour des requêtes ABOUT. Ceci semble être dû à la dégradation massive des performances quand on applique l'opérateur ABOUT aux termes individuels de la requête plutôt qu'à la requête entière, chose inévitable afin d'ajouter des poids individuels au termes.

Nous avons cherché à obtenir plus d'informations sur l'expansion de thèmes effectuée par l'opérateur ABOUT, ce que l'on peut obtenir par l'intermédiaire de la procédure ctx_query.explain. Mais la requête étendue renvoyée par cette procédure n'obtient pas les performances de la requête initiale, ce qui pourrait être dû à une perte d'information sur les poids relatifs des termes et phrases individuelles dans la requête étendue.

Les expressions de type AND, NEAR et OR se sont avérées entièrement inadaptées pour des requêtes plus longues. Les deux premières sont beaucoup trop restrictives pour le cadre TREC et ne renvoient pratiquement pas de documents, alors que l'évaluation est basée sur une liste de (normalement) 1000 documents renvoyés pour chaque requête. L'opérateur OR d'autre part considère tous les documents contenant au moins un des termes de la requête comme réponse valable, et à valeur égale, ce qui mène à une précision très faible.

L'utilisation des ces opérateurs en combinaison avec d'autres moyens de reformuler la requête n'a ou bien pas d'effet, ou alors dégrade la précision, sauf quand ils sont appliqués manuellement pour des requêtes très spécifiques. La transformation décrite dans l'application *Ultrasearch* donne ainsi les mêmes résultats qu'une simple requête ACCUM (avec lemmatisation) pour les requêtes de TREC 8.

Pour des requêtes courtes à base de mot-clés précis, les opérateurs AND et NEAR peuvent fournir des résultats adéquats, sachant qu'ils renvoient considérablement moins de documents que d'autres reformulations (ce qui peut être positif dans certaines applications). Pour certaines requêtes ils ne renvoient pas de documents du tout. La MAP indiquée dans le tableau est calculée uniquement sur les requêtes qui ont des résultats, la moyenne sur toutes les requêtes serait donc plus faible.

A.4. Discussion et perspectives

Nous avons constaté qu'il est possible d'obtenir de bons résultats pour des requêtes courtes, en particulier en combinant la recherche de thèmes (ABOUT) et de termes (stem) avec une pondération supplémentaire.

Sur la base des gains obtenus en ajoutant des pondérations aux termes dans des requêtes stem, nous avions espéré pouvoir améliorer de manière similaire des requêtes plus longues, utilisant l'opérateur ABOUT. Malheureusement, une trop grande partie du traitement interne des requêtes semble être cachée à l'utilisateur, et il est très difficile (ou même impossible) d'ajuster la manière dont sont classés les documents.

Bien que les performances en utilisant simplement l'opérateur `ABOUT` sans travail additionnel sur la requête peuvent être acceptables, en particulier pour des requêtes courtes composées essentiellement de mot-clés choisis, d'autres systèmes qui sont optimisés pour la recherche documentaire en langage naturel sont bien plus performants. Il serait donc intéressant de pouvoir optimiser le processus de recherche pour le type de requêtes rencontré dans un contexte d'application particulier.

Nous avons constaté un problème potentiel dans le fait qu'*Oracle Text* n'assigne que des valeurs entières de 0 à 100 aux documents. Ceci amène à un grand nombre de documents avec le même score, dont l'ordre n'est par conséquent pas déterminé. Il ne semble néanmoins pas que ceci ait un impact important sur les performances obtenues.

Table des figures

Liste des tableaux

Bibliographie

[Amati *et al.*, 2004] Giambattista AMATI, Claudio CARPINETO et Giovanni ROMANO (2004). Query difficulty, robustness and selective application of query expansion. *In Proceedings of ECIR'04*, Sunderland, UK.

[Amitay *et al.*, 2003] E. AMITAY, D. CARMEL, A. DARLOW, M. HERSCOVICI, R. LEMPEL, A. SOFFER, R. KRAFT et J. ZIEN (2003). Juru at TREC 2003 - topic distillation using query-sensitive tuning and cohesiveness filtering. *In TREC-12 Proceedings*.

[Baeza-Yates *et al.*, 2005] General Chair-Ricardo BAEZA-YATES, General Chair-Nivio ZIVIANI, Program Chair-Gary MARCHIONINI, Program Chair-Alistair MOFFAT et Program Chair-John TAIT, éditeurs (2005). *SIGIR '05 : Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, New York, NY, USA. ACM Press.

[Baeza-Yates, 1999] Ricardo A. BAEZA-YATES (1999). *Modern Information Retrieval*. Addison Wesley Publishing Company.

[Banks *et al.*, 1998] David BANKS, Paul OVER et Nien-Fan ZHANG (1998). Blind men and elephants : Six approaches to trec data. *Information Retrieval*.

[Bellot *et al.*, 2002] Patrice BELLOT, Eric CRESTAN, Marc EL-BÈZE, Laurent GILLARD et Claude De LOUPY (2002). Coupling named entity recognition, vector-space model and knowledge bases for trec-11 question answering track. *In Actes de TREC 11*.

[Callan *et al.*, 1992] James P. CALLAN, W. Bruce CROFT et Stephen M. HARDING (1992). The INQUERY retrieval system. *In Proceedings of DEXA-92, 3rd International Conference on Database and Expert Systems Applications*, pages 78–83.

[Carmel *et al.*, 2005] David CARMEL, Ian SOBOROFF et Elad YOM-TOV, éditeurs (2005). *Predicting query difficulty – methods and applications, ACM SIGIR 2005 workshop*, Salvador, Brasil.

[Carpineto *et al.*, 2001] Claudio CARPINETO, Renato DE MORI, Giovanni ROMANO et Brigitte BIGI (2001). An information-theoretic approach to automatic query expansion. *ACM Trans. Inf. Syst.*, 19:1–27.

[Chu-Carroll *et al.*, 2002] Jennifer CHU-CARROLL, John PRAGER, Christopher WELTY, Krzysztof CZUBA et David FERRUCCI (2002). A multi-strategy and multi-source approach to question answering. *In Actes de TREC 11*.

[Cleverdon, 1962] C.W. CLEVERDON (1962). Report on the testing and analysis of an investigation into the comparative efficiency of indexing systems. Rapport technique, College of Aeronautics, Cranfield, England.

[Cleverdon *et al.*, 1966] C.W. CLEVERDON, J. MILLS et E.M. KEEN (1966). Factors determining the performance of indexing systems. Rapport technique Vol.] : Design, Vol.2 : Test Results. Aslib Cranfield Research Project, College of Aeronautics, Cranfield, England.

[Cooper, 1988] William S. COOPER (1988). Getting beyond boole. *In Information Processing and Management*, volume 24, pages 243–248.

[Cronen-Townsend *et al.*, 2002] Steve CRONEN-TOWNSEND, Yun ZHOU et W. Bruce CROFT (2002). Predicting query performance. *In Proceedings of SIGIR 2002*, pages 299–306. ACM Press.

[de Loupy, 2000] Claude DE LOUPY (2000). *Évaluation de l'Apport de Connaissances Linguistiques en Désambiguïsation Sémantique et Recherche Documentaire*. Thèse de doctorat, Laboratoire Informatique d'Avignon.

[de Loupy et Bellot, 2000] Claude DE LOUPY et Patrice BELLOT (2000). Evaluation of document retrieval systems and query difficulty. *In Proceedings of "Using Evaluation within HLT Programs : Results and Trends"*, pages 32–39, Athènes, Grèce.

[Fox, 1983] E. FOX (1983). Characteristics of two new experimental collections in computer and information science containing textual and bibliographic concepts. Rapport technique Technical Report TR 83-561, Cornell University Computing Science Department.

[Freund et Schapire, 1996] Yoav FREUND et Robert E. SCHAPIRE (1996). Experiments with a new boosting algorithm. *In Thirteenth International Conference on Machine Learning*, pages 148–156.

[Gauvain *et al.*, 1999] J.-L. GAUVAIN, Y. KERCADIO, L. LAMEL et G. ADDA (1999). The LIMSI SDR System for TREC-8. *In Proc. of the Text Retrieval Conference, TREC-8, notebook*, Gaithersburg.

[Gillard *et al.*, 2005] Laurent GILLARD, Patrice BELLOT et Marc EL-BÈZE (2005). Le lia à equer (campagne technolangue des systèmes questions-réponses). *In Actes de TALN'05*, Dourdan, France.

[Grivolla, 2005] Jens GRIVOLLA (2005). Using Oracle for natural language document retrieval – an automatic query reformulation approach. *In* [Baeza-Yates *et al.*, 2005], pages 605–606.

[Harman, 1992] Donna HARMAN (1992). Overview of the First Text REtrieval Conference (TREC-1). *In TREC-1 Proceedings*, pages 1–20.

[Harman, 2000] Donna HARMAN (2000). What we have learned, and not learned, from trec. *In 22nd Annual Colloquium on IR Research*, Sidney Sussex College, Cambridge, England.

[Harman et Buckley, 2004] Donna HARMAN et Chris BUCKLEY (2004). SIGIR 2004 workshop : RIA and "where can IR go from here ?". *SIGIR Forum*, 38(2):45–49.

[He et Ounis, 2004] Ben HE et Iadh OUNIS (2004). Inferring query performance using pre-retrieval predictors. *In Proceedings of SPIRE 2004*.

[Joachims, 1998a] T. JOACHIMS (1998a). Making large-scale support vector machine learning practical. *In Advances in Kernel Methods : Support Vector Machines*. MIT Press, Cambridge, MA.

[Joachims, 1998b] Thorsten JOACHIMS (1998b). Text categorization with support vector machines : learning with many relevant features. *In* Claire NÉDELLEC et Céline ROUVEIROL, éditeurs : *Proceedings of ECML-98, 10th European Conference on Machine Learning*, numéro 1398, pages 137–142, Chemnitz, DE. Springer Verlag, Heidelberg, DE.

[Jourlin *et al.*, 2000] Pierre JOURLIN, Sue E. JOHNSON, Karen SPÄRCK JONES et Philip C. WOODLAND (2000). Spoken document representations for probabilistic retrieval. *Speech Communication*.

[Kuhn et de Mori, 1995] R. KUHN et R. DE MORI (1995). The application of semantic classification trees to natural language understanding. 17(5):449–460.

[Kwok, 2005] K.L. KWOK (2005). An attempt to identify weakest and strongest queries. *In* [Carmel *et al.*, 2005].

[Lavenus *et al.*, 2004] Karine LAVENUS, Jens GRIVOLLA, Laurent GILLARD et Patrice BELLOT (2004). Question-answer matching : two complementary methods, 2004. *In actes de la 7è conférence RIAO*, Avignon, France.

[Lavenus et Lapalme, 2002] Karine LAVENUS et Guy LAPALME (2002). Evaluation des systèmes de question réponse. aspects méthodologiques. 43(3):181–208.

[Lu et Keefer, 1994] X. A. LU et R. B. KEEFER (1994). Query expansion/reduction and its impact on retrieval effectiveness. *In TREC-3 Proceedings*. http ://trec.nist.gov/pubs/trec3/t3_proceedings.html.

[Macdonald *et al.*, 2005] Craig MACDONALD, Ben HE et Iadh OUNIS (2005). Predicting query performance in intranet search. *In* [Carmel *et al.*, 2005].

[Mahesh *et al.*, 1999] Kavi MAHESH, Jacquelynn KUD et Paul DIXON (1999). Oracle at TREC 8 : A lexical approach. *In TREC-8 Proceedings*.

[Mizzaro, 1998] Stefano MIZZARO (1998). How many relevances in information retrieval ? *Interacting with Computers*, 10(3):303–320.

[Mothe et Tanguy, 2005] Josiane MOTHE et Ludovic TANGUY (2005). Linguistic features to predict query difficulty. *In* [Carmel *et al.*, 2005].

[Müller et Wysotzki, 1997] W. MÜLLER et F. WYSOTZKI (1997). The decision-tree algorithm cal5 based on a statistical approach to its splitting algorithm. *In Machine Learning and Statistics : The Interface*, pages 45–65. R. Nakeiazadeh and C.C. Taylor.

[Ponte et Croft, 1998] Jay M. PONTE et W. Bruce CROFT (1998). A language modeling approach to information retrieval. *In Research and Development in Information Retrieval*, pages 275–281.

[Robertson *et al.*, 1996] S. ROBERTSON, S. WALKER, K. SPÄRCK JONES, M.M. HANCOCK-BEAULIEU et M. GATFORD (1996). Okapi at TREC-3. *In Overview of the Third Text REtrieval Conference (TREC-3)*.

[Robertson et Spärck Jones, 1976] Stephen E. ROBERTSON et Karen SPÄRCK JONES (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27:129–46.

[Rocchio, 1971] J. ROCCHIO (1971). Relevance feedback in information retrieval. *In* [Salton, 1971].

[Rorvig, 1999] Mark RORVIG (1999). Retrieval performance and visual dispersion of query sets. *In TREC-8 Proceedings*. http ://trec.nist.gov/pubs/trec8/t8_proceedings.html.

[Salton, 1971] Gerard SALTON, éditeur (1971). *The SMART Retrieval System : Experiments in Automatic Document Processing*. Prentice Hall, Englewood Cliffs, NJ.

[Salton, 1989] Gerard SALTON (1989). *Automatic Text Processing – The Transformation, Analysis, and Retrieval of Information by Computer*. Addison–Wesley.

[Salton, 1991] Gerard SALTON (1991). Developments in automatic text retrieval. *In Science*.

[Schmid, 1994] Helmut SCHMID (1994). Probabilistic part-of-speech tagging using decision trees. *In Actes de International Conference on New Methods in Language Processing*, pages 44–49, Manchester, UK.

[Schulmeister et Wysotzki, 1997] B. SCHULMEISTER et F. WYSOTZKI (1997). Dipol-a hybrid piecewise linear classifier. *In Machine Learning and Statistics : The Interface*, pages 133–151. R. Nakeiazadeh and C.C. Taylor.

[Sehgal et Srinivasan, 2005] Aditya SEHGAL et Padmini SRINIVASAN (2005). Predicting performance for gene queries. *In* [Carmel *et al.*, 2005].

[Sekine *et al.*, 2002] Satoshi SEKINE, Kiyoshi SUDO et Chikashi NOBATA (2002). Extended named entity hierarchy. *In Actes de LREC 2002*, Las Palmas, Canary Islands, Spain.

[Spärck Jones, 1999] Karen SPÄRCK JONES (1999). Summary performance comparisons TREC-2 through TREC-8. *In TREC-8 Proceedings, Appendix B*.

[Spärck Jones *et al.*, 2001] K. SPÄRCK JONES, P. JOURLIN, S.E. JOHNSON et P.C. WOODLAND (2001). The cambridge multimedia document retrieval (mdr) project : Summary of experiments.

[Spärck Jones *et al.*, 1998] Karen SPÄRCK JONES, Steve WALKER et Stephen E. ROBERTSON (1998). A probabilistic model of information retrieval : development and status. *Technical Report 446*.

[Spärck Jones et Webster, 1979] K. SPÄRCK JONES et C.A. WEBSTER (1979). Research in relevance weighting. Rapport technique British Library Research and Development Report 5553, Computer Laboratory, University of Cambridge.

[Swanson, 1988] D.R. SWANSON (1988). Historical note : information retrieval and the future of an illusion. *Journal of the American Society for Information Science*, 39(2):92–98.

[Tombros et Crestani, 1999] Tassos TOMBROS et Fabio CRESTANI (1999). A study of users' perception of relevance of spoken documents. Rapport technique TR-99-013, Berkeley, CA.

[Vapnik, 1982] Vladimir N. VAPNIK (1982). *Estimation of Dependences Based on Empirical Data*. Springer-Verlag.

[Vapnik, 1995] Vladimir N. VAPNIK (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc.

[Voorhees, 2002] E. M. VOORHEES (2002). Overview of the trec 2002 question answering track. *In Actes de TREC 11*.

[Voorhees, 2003] Ellen M. VOORHEES (2003). Overview of the trec 2003 robust retrieval track. *In TREC-12 Proceedings*.

[Voorhees, 2005] Ellen M. VOORHEES (2005). The trec robust retrieval track. *SIGIR Forum*, 39(1):11–20.

[Voorhees et Harman, 1996] Ellen M. VOORHEES et Donna HARMAN (1996). Overview of the fifth text retrieval conference. *In TREC-5 Proceedings*. http ://trec.nist.gov/pubs/trec5/t5_proceedings.html.

[Voorhees et Harman, 1997] Ellen M. VOORHEES et Donna HARMAN (1997). Overview of the sixth text retrieval conference. *In TREC-6 Proceedings*. http ://trec.nist.gov/pubs/trec6/t6_proceedings.html.

[Voorhees et Harman, 1999] Ellen M. VOORHEES et Donna HARMAN (1999). Overview of the eighth text retrieval conference. *In TREC-8 Proceedings*.

[Walker et de Vere, 1990] S. WALKER et R. DE VERE (1990). Improving subject retrieval in online catalogues : 2. relevance feedback and query expansion. *British Library Research Paper 72*.

[Wheatley, 1879] H. B. WHEATLEY (1879). *What is an Index ? A Few Notes on Indexes and Indexers*, page 28. London, UK : Longmans, Green & Co.

[Witten et Frank, 1999] Ian H. WITTEN et Eibe FRANK (1999). *Data Mining : Practical machine learning tools with Java implementations*. Morgan Kaufmann, San Francisco.

[Xu et Croft, 1996] Jinxi XU et W. Bruce CROFT (1996). Query expansion using local and global document analysis. *In SIGIR '96 : Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 4–11, New York, NY, USA. ACM Press.

[Yom-Tov *et al.*, 2005a] Elad YOM-TOV, Shai FINE, David CARMEL et Adam DARLOW (2005a). Learning to estimate query difficulty : including applications to missing content detection and distributed information retrieval. *In* [Baeza-Yates *et al.*, 2005], pages 512–519.

[Yom-Tov *et al.*, 2005b] Elad YOM-TOV, Shai FINE, David CARMEL et Adam DARLOW (2005b). Metasearch and federation using query difficulty prediction. *In* [Carmel *et al.*, 2005].

[Zobel, 1998] J. ZOBEL (1998). How reliable are the results of large-scale information retrieval experiments ? *In* W. B. CROFT, A. MOFFAT, C. J. VAN RIJSBERGEN, R. WILKINSON et J. ZOBEL, éditeurs : *Proceedings of the ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 307–314, Melbourne, Australia.

Publications

[Grivolla, 2004] Jens GRIVOLLA (2004). Méthodes statistiques et apprentissage automatique pour l'évaluation de requêtes en recherche documentaire. *In actes de Récital*, Fès, Maroc.

[Grivolla, 2005a] Jens GRIVOLLA (2005a). Une méthode pour l'évaluation automatique de la "difficulté" d'une requête. *In Actes de CORIA 2005*, Grenoble, France.

[Grivolla, 2005b] Jens GRIVOLLA (2005b). Using Oracle for natural language document retrieval – an automatic query reformulation approach. *In* General Chair-Ricardo BAEZA-YATES, General Chair-Nivio ZIVIANI, Program Chair-Gary MARCHIONINI, Program Chair-Alistair MOFFAT et Program Chair-John TAIT, éditeurs : *SIGIR '05 : Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 605–606, New York, NY, USA. ACM Press.

[Grivolla *et al.*, 2005] Jens GRIVOLLA, Pierre JOURLIN et Renato DE MORI (2005). Automatic classification of queries by expected retrieval performance. *In* David CARMEL, Ian SOBOROFF et Elad YOM-TOV, éditeurs : *Predicting query difficulty – methods and applications, ACM SIGIR 2005 workshop*, Salvador, Brasil.

[Lavenus et Grivolla, 2003] Karine LAVENUS et Jens GRIVOLLA (2003). Systèmes de question/réponse : Approches linguistiques et statistiques pour l'évaluation de requêtes. *In Atelier Question/Réponse de l'Atala*, Paris.

[Lavenus *et al.*, 2004a] Karine LAVENUS, Jens GRIVOLLA, Laurent GILLARD et Patrice BELLOT (2004a). Deux pistes complémentaires pour améliorer l'appariement question réponse. *In actes de l'atelier "Questions-réponses" de la conférence TALN*, Fès, Maroc.

[Lavenus *et al.*, 2004b] Karine LAVENUS, Jens GRIVOLLA, Laurent GILLARD et Patrice BELLOT (2004b). Question-answer matching : two complementary methods, 2004. *In actes de la 7è conférence RIAO*, Avignon, France.