Université de Montréal

Expansion de requête dans la recherche d'information- comparaison des ressources et des méthodes

par

Fuman Jin

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Mémoire présenté à la Faculté des études supérieures

en vue de l'obtention du grade de

Maître ès Sciences

en informatique

Décembre 2002

Université de Montréal

Faculté des études supérieures

Ce mémoire intitulé :

Expansion de requête dans la recherche d'information

Comparaison des ressources et des méthodes

présenté par :

Fuman Jin

a été évalué par un jury composé des personnes suivantes :

Victor Ostromoukhov	(président-rapporteur)
Jian-Yun Nie	(directeur de recherche)
Esma Aïmeur	(Co-directrice de recherche)
Philippe Langlais	(membre du jury)

Mémoire accepté le : _____

Sommaire

L'expansion de requête est une méthode simple et efficace pour élargir la couverture de recherche et permet ainsi de trouver plus de documents pertinents. Elle consiste simplement à ajouter dans la requête initiale des termes supplémentaires reliés.

Dans les études précédentes sur l'expansion de requête, avec le modèle vectoriel, les termes d'expansion sont ajoutés directement dans le vecteur original. Cependant, l'ajout direct des termes d'expansion peut biaiser la requête initiale en ajoutant de l'emphase au terme original : plus un terme est étendu, plus l'aspect que ce terme représente est renforcé.

Au lieu de l'ajout direct des termes d'expansion dans la requête originale, nous proposons de combiner le terme et ses termes d'expansion au moyen de l'opérateur logique OU. Notre idée de base est qu'un terme d'expansion représente une expression alternative du terme original, plutôt qu'une expression additionnelle. De cette façon, nous ne renforçons pas un terme auquel beaucoup de termes sont reliés. L'équilibre globale entre les termes dans une requête est gardé.

Dans cette étude, nous implantons cette nouvelle méthode d'expansion dans le système de recherche d'information SMART. Puis, nous évaluons l'expansion de requête avec les trois différentes ressources suivantes :

- WordNet, un thésaurus construit manuellement
- l'information mutuelle, une relation statistique calculée selon les occurrences des mots dans les documents
- et la pseudo-rétroaction de la pertinence ("pseudo relevance feedback"), une relation statistique obtenue des résultats d'une première recherche

Nos expériences dans une collection de TREC montrent que cette approche est plus approprié que le simple ajout des termes dans la requête étendue.

Mots-clés : opérateur logique, modèle vectoriel, expansion de requête, information mutuelle, WordNet

Abstract

Query expansion is a simple and effective method to extend the coverage of retrieval and thus makes it possible to find more relevant documents. Basically, it consists in adding related terms in the initial query.

In the previous studies on query expansion with the vector space model, expansion terms are added directly in the original vector. However, direct addition of expansion terms can change the focus of the initial query by adding emphasis to original terms : the more a term is expanded, the more the aspect it represents is reinforced.

Instead of the direct addition of expansion terms in the original query, we propose to combine the term and its expansion terms by means of the logical operator OR. Our basic idea is that an expansion term represents an alternative expression of the original term, rather than an additional expression. In this way, the focus of the whole query remains unchanged.

In this study, we implement this new expansion method in the information retrieval system SMART. Then, we evaluate query expansion with the following three resources :

- WordNet, a manually constructed thesaurus
- Mutual information, a statistical relation calculated according to the occurrences of terms in the documents
- and Pseudo-relevance feedback, a statistical relation derived from first-round retrieval

Our experiments on a TREC collection show that our approach is more appropriate than the simple addition approach.

Key words : Logical operator, vector space model, query expansion, mutual information, WordNet

Table des matières

Table des figures

Liste des tableaux

Remerciements

Je remercie mon directeur de recherche Jian-Yun Nie pour m'avoir guidé, aidé et encouragé tout au long de cette rercherche. Je remercie aussi ma co-directrice Esma Aïmeur pour m'avoir encouragé.

Merci aussi à ma mère, à mon père, à ma soeur pour leur support, et à tout le monde du RALI pour les gâteaux, les bonbons et les boissons.

Chapitre 1

Introduction

Dans ce mémoire, nous décrivons notre travail sur l'expansion de requête dans la recherche d'information. Nous proposons une nouvelle méthode d'expansion en considérant les termes d'expansion comme des alternatives aux termes originaux.

Selon Salton [SM83], la recherche d'information étudie la représentation, le stockage, l'organisation, et l'accès de l'information. Dans le passé, la recherche d'information était un domaine restreint auquel se sont intéressés surtout les bibliothécaires et les gens en sciences de l'information. Mais grâce à la popularité et à l'essor de l'Internet, la recherche d'information connaît un regain d'intérêt. En effet, à partir des années 1990, avec l'ouverture de l'accès d'Internet au grand public, l'Internet croît de façon fulgurante. Le nombre d'utilisateurs d'Internet augmente exponentiellement, et le volume d'information disponible sur l'Internet ne cesse de grossir. L'une des raisons

de cette augmentation est reliée au succès du World Wide Web qui permet à tout le monde de créer ses propres documents multimédia et de les faire pointer sur d'autres pages Web.

Une quantité extraordinaire d'information sous forme électronique est maintenant disponible et accessible en tout temps. Actuellement en 2002, le moteur de recherche Google [Goo] répertorie 3 083 324 652 pages Web. Le Web est devenu maintenant une source majeure d'information pour les gens du monde entier.

Cependant, même aujourd'hui en utilisant les meilleurs moteurs de recherche, ces systèmes de recherche n'arrivent pas à trouver les informations qui satisfont entièrement les besoins des utilisateurs [Har92]. Il y a plusieurs raisons à ce phénomène. L'un des problèmes dans la recherche d'information est que les requêtes formulées par les utilisateurs sont souvent une spécification partielle de leur besoin d'information. En effet, ils ne se donnent pas la peine de construire de longues requêtes, ils soumettent des requêtes très courtes et succintes. C'est le cas notamment sur le Web où la longueur moyenne des requêtes est environ de 2 mots [ASS01, CCW95]. En fait, la performance d'un système de recherche d'information est proportionnelle à la longueur de la requête [QF93]. Souvent, un terme n'est pas suffisant pour exprimer ou décrire un concept intégralement. Il est donc nécessaire d'ajouter d'autres termes pour le compléter.

Par ailleurs, il existe aussi un problème de vocabulaire. Les termes formulés par l'utilisateur ne correspondent pas nécessairement aux termes utilisés

dans les documents pour un même concept [BM90]. La plupart des systèmes de recherche d'information se basent sur la correspondance des termes pour trouver les documents pertinents. Les systèmes de recherche d'information identifient les documents qui contiennent les termes de la requête de l'usager. Mais comme très souvent et surtout dans une grande collection, les utilisateurs et les auteurs des documents de la collection n'utilisent pas les mêmes termes pour décrire un même concept, un très grand nombre de documents pertinents sont ignorés par le système. Ceci est dû au fait que la comparaison entre une requête et un document se limite aux termes en commun qui appartiennent à la fois au document et à la requête.

L'expansion de requête [Fox80] est une solution possible à ces problèmes. L'expansion de requête consiste à compléter la requête initiale avec des termes supplémentaires, élargissant ainsi la couverture de recherche. En enrichissant les requêtes avec des termes fortement reliés aux termes initiaux, l'expansion de requête règle le problème de requête courte et le problème de vocabulaire, permettant ainsi d'aller chercher plus de documents pertinents.

Dans les études précédentes sur l'expansion de requête dans le modèle vectoriel, les chercheurs se sont concentrés sur la sélection des termes d'expansion et sur la pondération de ces nouveaux termes. Ils ont obtenu des résultats variés. Toutefois, dans toutes ces études, on présume que les termes d'expansion doivent être ajoutés directement dans la requête originale. Par exemple, si un terme A est relié au terme B alors une requête qui contient le terme B peut être étendue en additionnant A dans la requête.

Dans ce mémoire, nous allons comparer la performance de l'expansion de requête en utilisant trois différentes ressources et évaluer une nouvelle méthode pour intégrer les termes d'expansion dans le vecteur de la requête étendue. Notre idée de base est qu'un terme d'expansion représente une expression alternative du terme original, plutôt qu'une expression additionnelle. Donc il doit être combiné avec le terme original en utilisant une relation de OU.

Dans cette étude, nous allons implanter cette idée dans le modèle vectoriel. Nous allons utiliser le système de recherche SMART pour faire nos expériences. Nous utiliserons les trois ressources suivantes :

- WordNet
- l'information mutuelle
- et la pseudo-rétroaction de la pertinence ("pseudo relevance feedback")

Nous ferons nos évalutions sur la collection de TREC AP en utilisant des requêtes courtes et longues.

Dans le prochain chapitre, nous présenterons une revue de littérature sur l'expansion de requête, afin de mettre en évidence la diversité des approches et des résultats obtenus. Également, nous introduirons le concept de l'information mutuelle que nous utilisons.

Le chapitre 3 aura pour objet la description de notre principe de l'expansion de requête. En outre, nous présenterons les problèmes reliés à l'ajout direct et nous proposerons une autre façon d'intégrer les termes d'expansion dans la requête étendue en expliquant le bien fondé de notre approche.

Le chapitre 4 couvrira les détails de notre implantation de l'expansion de requête dans le système de recherche SMART.

Le chapitre 5 présentera les différents résultats obtenus ainsi que leurs interprétations.

Et finalement, dans le chapitre 6, nous conclurons ce mémoire avec une discussion sur les résultats obtenus et nous formulerons quelques conclusions.

1.1 Glossaire [Wei]

- Besoin d'information : ce que l'utilisateur veut vraiment savoir. Une requête est une approximation du besoin d'information.
- Collection : groupe de documents dans lequel un utilisateur souhaite obtenir de l'information.
- Document : une information que l'utilisateur peut vouloir retrouver. Cette information peut être un fichier de textes, une page web, une image, ou une phrase d'un livre.
- Expansion de requête : tout processus qui construit une nouvelle requête à partir de la requête initiale.
- Fichier inverse : Un index qui, pour chaque terme apparaissant dans la collection, donne une liste de documents où ce terme est présent.
- Hyperonyme : une relation sémantique entre deux concepts, il représente la catégorie supérieure. Par exemple, l'arbre est l'hyperonyme de l'érable.

- Hyponyme : une relation sémantique entre deux concepts, il représente la catégorie inférieure. Par exemple, l'érable est un hyponyme de l'arbre.
- IDF ("Inverse Document Frequency") : IDF est une mesure de la distribution d'un terme particulier dans tous les documents de la collection. Elle est habituellement définie comme log(taille de la collection/nombre de document contenant le terme).
- Indexation : Processus de conversion d'une collection en une forme appropriée à la recherche.
- Pertinence : une mesure abstraite qui indique si un document satisfait au besoin d'information de l'utilisateur.
- Précision : une mesure standard de la performance de recherche d'information, la précision est définie par le nombre de documents pertinents trouvés, divisé par le nombre total de documents trouvés. Par exemple, supposons que dans la collection il y a 80 documents qui ont rapport avec le besoin d'information. Le système retourne 60 documents, dont 40 d'entre eux ont rapport avec le besoin d'information. Alors la précision de ce système est de $40/60 = 67\%$.
- Rappel : une mesure standard de la performance de recherche d'information, le rappel est défini par le nombre de documents pertinents trouvés divisé par le nombre total de documents pertinents dans la collection. Par exemple, supposons que dans la collection il y a 80 documents qui ont rapport avec le besoin d'information. Le système retourne 60 documents, dont 40 d'entre eux ont rapport avec le besoin d'information. Alors le rappel de ce système est de $40/80 = 50\%$.

- Rétroaction de la pertinence ("Relevance feedback") : une forme d'expansion de requête où l'utilisateur indique au système quels documents retournés par le système sont pertinents à sa requête. Le système trouve les termes communs à ces documents pertinents, les ajoute à la vieille requête et lance une nouvelle recherche avec cette requête, dans le but d'améliorer la performance de recherche.
- Requête : une liste de termes qui caractérisent l'information que l'utilisateur recherche.
- Similarité : une mesure qui indique à quel point une requête ressemble à un document. Dans le modèle vectoriel, une méthode populaire est de calculer le cosinus de l'angle des deux vecteurs.
- Terme : un simple mot, ou un terme complexe (composé) identifié comme indexe.
- TF ("Term Frequency") : le nombre de fois qu'un terme particulier se retrouve dans un document.
- TREC : "Text REtrieval Conference" une conférence sur la recherche d'information dans laquelle une collection de tests et des requêtes sont fournies.

Chapitre 2

Revue de littérature

Dans cette revue de littérature, nous allons d'abord introduire le modèle vectoriel et le système de recherche d'information SMART. Ensuite, nous allons présenter différents travaux de recherche importants effectués sur l'expansion de requête dans le domaine de la recherche d'information. Nous pouvons distinguer deux grandes voies dans l'expansion de requête. La première est l'expansion de requête qui n'a pas besoin d'une recherche préliminaire. Et la deuxième est l'expansion de requête qui se base sur les résultats d'une recherche préliminaire.

2.1 Modèle vectoriel

Dans la recherche d'information, le modèle joue un rôle très important. Le modèle accomplit les deux fonctions suivantes :

1. Il permet de créer une représentation interne pour le document et pour

la requête à partir des termes pondérés issus de l'indexation.

2. Il définit une méthode de comparaison entre une représentation de document et une représentation de requête pour calculer leur similarité.

Le modèle vectoriel [SM83] est souvent utilisé dans la recherche d'information. L'espace vectoriel est formé par tous les termes issus de l'indexation. Les documents et les requêtes sont représentés par un vecteur de poids. Un poids indique l'importance d'un terme dans le document ou dans la requête. Les poids des termes des documents et des requêtes peuvent être calculés en utilisant le schéma de pondération $tf \times idf$ [SM83]. Dans ce schéma, le poids attribué à un terme dépend de sa fréquence dans le document, le tf ("term frequency"), et de la fréquence de ce terme dans tous les documents de la collection, le idf ("inverted document frequency").

$$\begin{array}{c} \\ Doc_1 \\ Doc_2 \\ \vdots \\ Doc_m \end{array} \begin{array}{c} \begin{array}{cccc} Terme_1 & Terme_2 & \cdots & Terme_n \end{array} \\ \begin{pmatrix} poids_{11} & poids_{12} & \cdots & poids_{1n} \\ poids_{21} & poids_{22} & \cdots & poids_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ poids_{m1} & poids_{m2} & \cdots & poids_{mn} \end{pmatrix} \end{array}$$

FIG. 2.1 – Représentation d'une collection dans le modèle vectoriel

Dans le modèle vectoriel, une collection de documents est représentée par une matrice, comme l'illustre la figure 2.1. Chaque rangée de la matrice est un

document. Le $poids_{ij}$ représente le poids du terme $Terme_j$ dans le document Doc_i. Il y a autant de dimensions que de nombre de termes dans la collection et il y a autant de rangées que de nombre de document dans la collection.

Il y a plusieurs façons de calculer la similarité entre un vecteur document et un vecteur requête. Soit un document D, représenté par le vecteur $[d_1, \quad d_2, \quad \cdots, \quad d_n]$ et une requête Q, représentée par le vecteur $[q_1, \quad q_2, \quad \cdots, \quad q_n]$. Les deux formules les plus souvent utilisées pour le calcul de similarité sont le produit interne et la formule du cosinus, qui sont respectivement :

$$Sim(D, Q) = \sum_{i=1}^{n} (d_i \times q_i) \tag{2.1}$$

$$Sim(D, Q) = \cos(D, Q) = \frac{\sum_{i=1}^{n} (d_i \times q_i)}{\sqrt{\sum_{i=1}^{n} (d_i)^2 \times \sum_{i=1}^{n} (q_i)^2}} \tag{2.2}$$

La formule du cosinus donne des valeurs dans l'intervalle [0, 1], elle est donc normalisée. Mais la formule du produit interne ne l'est pas. Le degré de similarité entre un document et une requête dépend du poids des termes qu'ils ont en commun. Plus ils partagent des termes de poids élevés, plus la similarité entre eux est grande.

2.2 SMART

SMART [Buc85] est un système de recherche d'information expérimental développé à l'université de Cornell. Il implante le modèle vectoriel proposé par Salton dans les années 1960. Le but principal de SMART est de fournir

une structure dans laquelle on peut faire de l'indexation, de la recherche d'information, et de l'évaluation.

Les documents et les requêtes peuvent être indexés automatiquement par SMART qui crée des vecteurs de poids pour les représenter. Le poids indique l'importance d'un terme dans le document comme décrit en haut. Plusieurs schémas de pondération sont disponibles pour le calcul de poids des termes dans les documents et les requêtes. Nous utilisons le schéma de pondération *ltc* qu nous décrirons plus loin. Les vecteurs sont sauvegardés sur le disque et un fichier inversé est crée pour les documents. Lors de l'évaluation, le vecteur de la requête est d'abord comparé aux vecteurs des documents pour calculer une valeur de similarité. Une liste de documents triés dans l'ordre décroissant est ensuite retournée.

Nous avons modifié SMART pour qu'il puisse étendre les requêtes. Nous utilisons SMART pour indexer la collection des documents et pour évaluer les requêtes.

2.3 Expansion de requête

La plupart du temps, l'expansion de requête emploie des thésauri pour connaître les termes reliés. Un thésaurus est une structure de donnée qui décrit des concepts et les relations sémantiques qui existent entre eux.

L'expansion de requête consiste à choisir des termes dans le thésaurus qui sont fortement reliés aux termes originaux de la requête en utilisant les rela-

tions définies dans le thésaurus. Un poids est ensuite assigné aux nouveaux termes qui sont ajoutés dans la requête originale pour former une nouvelle requête. Par exemple, si le terme "voiture" est dans la requête originale, en consultant ce terme dans le thésaurus, on trouve qu'il a un lien avec le terme "automobile", ils sont reliés par une relation de synonyme. Donc le terme "automobile" est ajouté dans la requête originale et un poids lui est calculé, ce qui a pour effet d'élargir la couverture de la requête et ainsi augmenter le rappel.

De nombreuses études ont été publiées sur l'expansion de requête en utilisant des thésauri. Dans la recherche d'information, deux sortes de thésaurus sont utilisés, il y a les thésauri manuels et les thésauri statistiques.

2.3.1 Thésaurus manuel

Les thésauri manuels, comme son nom l'indique, sont construits à la main par des êtres humains qui établissent des relations et des liens d'hiérarchie entre les termes. Dans un thésaurus manuel, on y trouve donc des concepts et des relations sémantiques qui les relient. Ces relations peuvent être des relations de synonymie, d'hyperonymie, d'hyponymie, etc.

2.3.2 WordNet

WordNet [Mil95] est un thésaurus électronique anglais qui a été conçu en s'inspirant des théories psycholinguistiques courantes de la mémoire lexicale des humains. WordNet distingue quatre groupes grammaticaux, soient :

les noms, les ajectifs, les verbes et les adverbes. Ceux-ci sont organisés sous forme d'ensembles de synonymes appelés "synset". Les ensembles de synonymes sont reliés entre eux par différents types relations, par exemple : synonymie, antonymie, hyponymie, hyperonymie, etc. WordNet a été développé par le laboratoire de science cognitive de l'université de Princeton sous la direction du professeur George A. Miller.

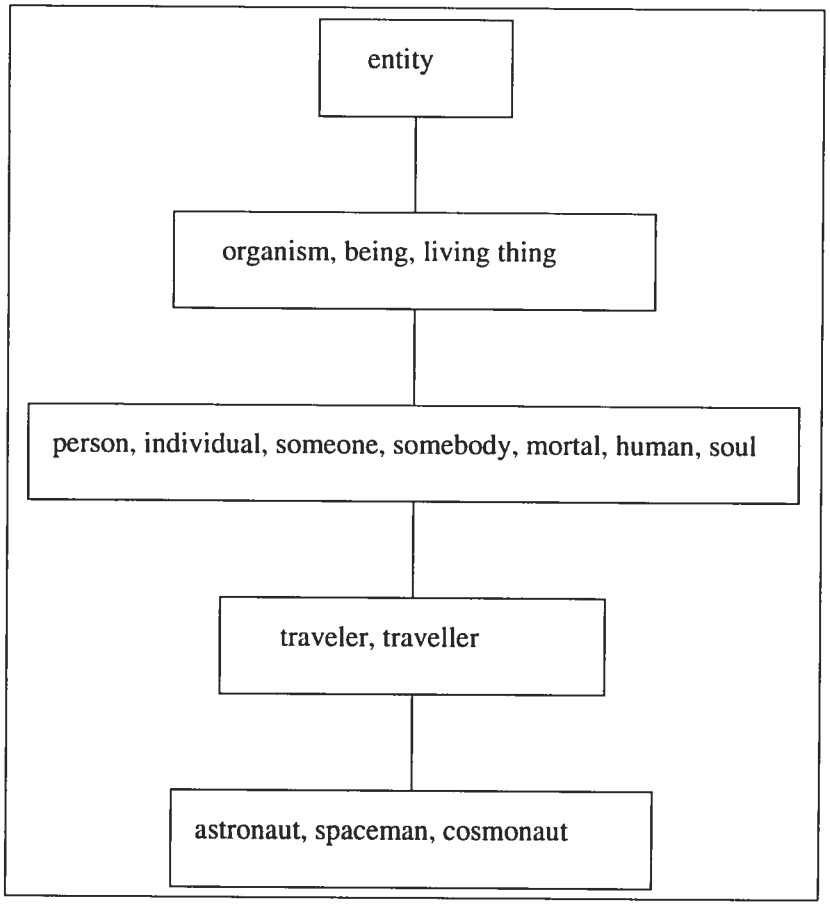

FIG. 2.2 – Les hyperonymes du nom "astronaut" dans WordNet

La figure 2.2 montre le "synset" correspondant à l'astronaute (en anglais "astronaut") ainsi que les "synsets" qui lui sont reliés par la relation d'hyperonymie.

L'utilisation de thésausi manuels dans l'expansion de requête a donné des résultats variés.

Voorhees [Voo94] a testé l'expansion de requête sur la collection de TREC en utilisant le thésaurus WordNet [Mil95]. Dans ses expériences, elle a choisi manuellement des synonymes, des hyperonymes et des hyponymes dans WordNet et les a ajoutés dans la requête originale. Elle a attribué un poids de 1 aux termes originaux de la requête et des poids de 0.1, 0.3, 0.5, 1, et 2 pour les termes ajoutés. Elle a utilisé le système de recherche d'information SMART pour faire les évaluations. Dans ses expériences, elle a réussi à améliorer seulement la performance des requêtes courtes. Cependant, pour les requêtes longues, la performance a tendance à se dégrader. Elle a aussi utilisé WordNet pour désambiguïser le sens des mots pour la recherche d'information [Voo93], mais elle n'a pas obtenu de bons résultats.

Mandala [MTT98, MTT99] a cherché à comprendre la raison pour laquelle l'utilisation de WordNet dans l'expansion de requête n'a pas réussi à améliorer l'efficacité de la recherche. Il a montré qu'il manque dans WordNet la plupart des relations entre les termes qui sont utiles pour la recherche d'information. Par exemple, dans WordNet, il n'y a pas de nom propre et il n'existe pas de relation entre des termes qui proviennent de groupes grammaticaux ("parts of speech") différents. Ces résultats ont indiqué aussi que

l'expansion de requête peut augmenter le rappel mais dégrade la précision. Pour pallier ces faiblesses, il proposa donc d'enrichir WordNet avec d'autres thesauri. Il a combiné WordNet avec un thésaurus statistique basé sur l'information mutuelle et avec un thésaurus basé sur la dépendance syntaxique. L'idée derrière cette combinaison est que chaque type de thésaurus possède des caractéristiques différentes et, donc en les combinant, on obtient un thésaurus plus adéquat et plus complet pour l'expansion de requête. Il réussit ainsi à améliorer fortement l'efficatité de recherche de cette manière.

2.3.3 Thésaurus statistique

Les thésauri statistiques sont construits à partir de la collection de documents en exploitant les co-occurrences de termes dans la collection. L'idée générale derrière l'utilisation des statistiques sur les co-occurrences de termes pour construire des thésauri est la suivante : les termes qui ont tendance à apparaître ensemble dans des documents ont des chances d'être semblables ou reliés. Les données de co-occurrence fournissent donc une mesure statistique pour identifier automatiquement des relations entre des termes qui normalement sont contenues dans un thésaurus construit à la main.

Selon Qiu et Frei [QF93], il est possible d'obtenir une amélioration consistante et significative du rappel et de la précision en utilisant des thésauri construits automatiquement avec des données statistiques. Il faut cependant accorder un certain poids aux nouveaux termes en considérant leurs similarités avec tous les termes de la requête originale. Ils ont obtenu une augmen-

tation de performance d'environ de 20% à 30%.

Il existe de multiples façons de calculer la relation statistique entre des termes. La plupart d'entre elles sont "ad hoc". Nous ne les présentons pas ici. Mais nous nous concentrons sur le calcul de l'information mutuelle.

2.3.4 Information mutuelle

Généralement les gens répondent plus rapidement au mot "infirmière" si celui-ci suit un mot qui lui est fortement associé comme celui de "docteur". L'information mutuelle [CH90] est une mesure objective basée sur la théorie de l'information qui permet d'estimer ce genre d'association entre les mots dans un corpus de documents. L'information mutuelle est calculée à partir de la distribution des termes dans un corpus.

L'information mutuelle I(x,y) d'un terme x avec un autre terme y se défininit comme suit :

$$I(x,y) = P(x,y) \times \log_2 \frac{P(x,y)}{P(x)P(y)} \tag{2.3}$$

où x et y sont des termes, $P(x,y)$ est la probabilité d'observer x et y ensemble dans la collection. $P(x)$ et $P(y)$ sont les probabilités d'observer x et y dans la collection. L'information mutuelle compare la probabilité d'observer x et y ensemble $P(x,y)$, avec la probabilité d'observer x et y indépendamment $P(x) \times P(y)$.

Dans notre implantation, $P(x)$ et $P(y)$ sont estimées en comptant le

nombre d'observations de x et de y dans le corpus et en le normalisant par N, la taille du corpus (comptée en mots). $P(x, y)$ est estimée en comptant le nombre fois où x et y apparaissent ensemble dans une fenêtre de 20 termes et en le normalisant par N.

Une autre formule de l'information mutuelle, appelée information mutuelle ponctuelle est la suivante :

$$I(x, y) = \log_2 \frac{P(x, y)}{P(x)P(y)} \tag{2.4}$$

La seule différence avec la formule précédente est qu'elle n'est pas multipliée par $P(x, y)$.

Les valeurs de l'information mutuelle varient beaucoup d'un terme à l'autre. Pour les rendre plus comparables, nous normalisons les valeurs de la façon suivante :

$$NI(x, y) = \frac{I(x, y)}{max_{yi}(I(x, y_i))}$$

où y_i est un terme d'expansion de x.

2.4 Expansion de requête basée sur les résultats d'une recherche

Une autre méthode d'expansion est d'utiliser des termes provenant des documents qui ont été jugés pertinents. Dans ces méthodes, il faut d'abord lancer une recherche préliminaire car la source des termes d'expansion est

fournie par les documents pertinents. L'expansion de requête consiste à choisir des termes dans ces documents pertinents, puis à les ajouter dans la requête initiale et à repondérer les poids des termes de la requête. Il existe deux méthodes qui exploitent des termes issus des documents pertinents, le "relevance feedback" et le "pseudo relevance feedback".

2.4.1 Rétroaction de pertinence ("Relevance feedback")

La rétroaction de pertinence est une méthode efficace pour améliorer la précision et le rappel. Elle fonctionne de la manière suivante : par exemple, l'utilisateur formule la requête initiale, "voiture Batman". Il la soumet au système qui lui retourne un ensemble de documents qui constitue une liste. L'utilisateur sélectionne à partir de cette liste un document pertinent et il indique au système que le document choisi est pertinent. Le système utilise alors ce document pour repondérer la requête, et ajouter des nouveaux termes afin de formuler une nouvelle requête. Si le document que l'utilisateur a jugé pertinent contient le terme "Batmobile" alors le système peut l'ajouter dans la requête. La requête est modifiée de manière à ce qu'elle ressemble aux documents que l'utilisateur souhaite obtenir. Le système peut lui trouver ainsi des documents qui répondent mieux aux besoins d'information de l'utilisateur, ainsi, le rappel et la précision peuvent être augmentés.

De la même façon, on peut aussi diminuer le poids des termes qui se trouvent dans les documents que l'utilisateur considère non pertinents, ou les enveler directement. La formule de reformation "relevance feedback" la

plus souvent utilisée est la formule de Roccio [JJR71] :

$$Q_{new} = \alpha * Q_{old} + \beta * R - \gamma * NR$$

où Q_{old} représente l'ancienne requête, R est un vecteur formé d'un ensemble de termes selectionnés parmi les documents pertinents, et NR est un autre vecteur formé de termes appartenant aux documents non pertinents. α, β et γ sont des paramètres qui permettent de règler l'importance relative de ces trois facteurs dans la construction de la nouvelle requête Q_{new}. La requête subséquente Q_{new} constitue alors une meilleure approximation de la requête initiale Q_{old}. Et par conséquent, elle produit une meilleure recherche.

La requête originale de l'utilisateur permet d'identifier une région de la collection qui doit contenir des documents pertinents. En identifiant des documents pertinents et des documents non pertinents à l'intérieur de cette région, l'utilisateur fournit au système suffisamment d'information pour construire une requête modifiée qui sera centrée autour des documents pertinents et qui s'éloignera le plus possible des documents non pertients.

En additionnant de 300 à 530 termes provenant des documents pertinents et en repondérant la nouvelle requête avec la formule de Roccio, Buckley [BSAS94] a réussi à obtenir une augmentation de performance de 25% sur une précision moyenne initiale de 29.85.

2.4.2 "Pseudo relevance feedback"

Le problème de "relevance feedback" réside dans le fait qu'il exige que les jugements de pertinence portés par l'utilisateur soient connus. Ce qui n'est souvent pas possible. Pour résoudre ce problème, la technique de pseudo relevance feedback est proposée.

Le "pseudo relevance feedback" fonctionne comme ceci : comme les documents pertinents ne sont pas connus, les n premiers documents, trouvés avec la requête initiale, sont considérés pertinents. Le système utilise alors les termes de ces n premiers documents pour étendre la requête.

Cependant, cette méthode possède un problème : si une grande partie des n documents considérés pertinents ne le sont pas, les termes ajoutés à la requête (provenant de ces documents) ne sont probablement pas reliés à la requête initiale. Ainsi, la qualité des documents retournés par la requête étendue sera probablement plus mauvaise. L'efficacité de cette méthode dépend surtout de la qualité des n premiers documents.

En utilisant l'expansion de requête avec la méthode "pseudo feedback" se basant sur les premiers documents retrouvés, Buckley [BSAS94] réussi à améliorer la performance du système de 16% à 20%.

2.5 Synthèse

Dans ce chapitre, nous avons introduit le modèle vectoriel et SMART. Ensuite, nous avons présenté les différents approches de l'expansion de requête, ainsi que les améliorations obtenus.

Dans tous ces études, nous pouvons remarquer que les termes d'expansion sont ajoutés directement dans la requête étendue, soit comme des dimensions additionnelles, soit pour augmenter le poids des dimensions existantes. Dans le prochain chapitre, nous nous interrogeons sur cette façon d'étendre la requête, et nous proposons une autre méthode.

Chapitre 3

Principe de l'expansion de requête

Nous allons présenter dans ce chapitre l'expansion de requête. Les requêtes courtes ne contiennent pas assez de termes reliés au sujet en question. Par conséquent, de telles requêtes retournent un nombre limité de documents pertinents. Une solution consiste à ajouter dans la requête initiale de nouveaux termes qui sont reliés aux termes initiaux. En ajoutant de nouveaux termes, la requête originale est élargie de sorte que la couverture de recherche est élargie aussi. On peut donc s'attendre à trouver un nombre plus élevé de documents pertinents et augmenter ainsi le taux de rappel.

Les questions importantes qui doivent être considérées dans l'expansion de requête sont : Quels sont les mots à ajouter ? Comment les nouveaux termes doivent-ils être intégrés dans la nouvelle requête ? La précision est-elle affectée ?

Nous allons voir que dans le modèle vectoriel, la requête étendue construite

par ajout direct des termes d'expansion dans la requête originale peut engendrer des problèmes. Au lieu de l'ajout direct, nous proposons de combiner les termes d'expansion avec le terme original, en les connectant avec l'opérateur logique OU. Nous allons expliquer le bien fondé de notre approche, et montrer comment cette nouvelle requête étendue est construite et comment elle est évaluée.

3.1 Comment déterminer les termes à ajouter ?

Généralement, l'expansion de requête utilise un thésaurus comme source de termes d'expansion. Le thésaurus contient les relations entre les termes. Durant le processus d'expansion de requête, le système de recherche demande au thésaurus de lui fournir des termes qui sont reliés aux termes de la requête. Comme nous avons mentionné, les thésauri utilisés dans l'expansion de requête peuvent être séparés en deux catégories :

1. Il y a les thésauri génériques construits manuellement (Miller [Mil95]). Dans ces thésauri, on y trouve les relations de sysnonymie, d'hyperonymie, d'hyponymie etc...
2. Il y a les thésauri statistiques construits automatiquement à partir des données extraites de la collection (Rijsbergen [vR77]). Dans ces thésauri, on établit des relations entre les termes qui apparaissent très souvent ensemble.

3.2 Comment les termes d'expansion sont intégrés dans la requête ?

Après avoir obtenu les termes d'expansion, une nouvelle requête doit être construite à partir de ces termes d'expansion et des termes originaux de la requête. Les termes d'expansion sont intégrés différemment selon le modèle. Regardons comment les termes d'expansion sont intégrés dans le modèle booléen et dans le modèle vectoriel.

3.2.1 Modèle booléen

Dans le modèle booléen, les termes d'expansion sont mis en disjonction avec les termes de la requête originale. Par exemple, si un terme A est relié au terme B alors A est mis en disjonction avec B dans la requête étendue. Ainsi, dans la nouvelle requête, B va être remplacé par (B OU A). Le terme d'expansion est considéré comme une alternative au terme originale. Cette approche semble raisonnable. Mais en général, peu d'utilisateur formule leurs requêtes en forme booléenne.

Cependant, dans le modèle vectoriel, il n'y a pas cette possibilité d'ajouter des termes en disjonction.

3.2.2 Modèle vectoriel

Dans le modèle vectoriel, la plupart du temps, la requête étendue est construite avec l'ajout direct qui consiste à ajouter directement les nouveaux

termes dans la requête étendue. Par exemple si le terme A est relié au terme B, alors une requête qui contient le terme B peut être étendue en ajoutant directement le terme A dans le vecteur de la requête. Le terme d'expansion est considéré comme un supplément au terme original et non comme une alternative.

3.3 Problèmes dans la méthode d'ajout direct

Regardons ce qui se passe si un terme important ne se trouve pas dans une requête. Le niveau de similarité calculée entre une telle requête et les documents est bas, car ce terme important n'est pas là pour apporter sa contribution dans le calcul de similarité. De même, ce terme important ne prend pas part à la discrimination des documents. Par conséquent, le taux de rappel est bas. C'est pourquoi il faut ajouter des termes reliés à la requête initiale.

Dans cette section nous allons examiner la méthode d'ajout direct et la requête étendue construite à partir de celle-ci. De plus, nous allons montrer les différents problèmes qu'elle peut engendrer.

3.3.1 Ajout direct

Comme nous venons de le voir, dans le modèle booléen, les termes d'expansion sont mis en disjonction dans la requête étendue, alors que, dans le modèle vectoriel, il n'est pas possible d'ajouter des opérateurs logiques.

Dans le modèle vectoriel, les termes d'expansion sont ajoutés directement dans la requête étendue, ce qui a pour effet que les termes d'expansion sont considérés comme des suppléments aux termes originaux et non des alternatives comme dans le modèle booléen.

L'ajout direct est simple et facile à implanter. Mais comme les termes d'expansion sont considérés comme des suppléments aux termes originaux, l'ajout direct peut déséquilibrer les poids des termes de la requête étendue et ainsi biaiser le besoin d'information de l'utilisateur.

3.3.2 Biais du besoin d'information original

Dans cette section nous allons montrer qu'à cause de l'ajout direct des termes d'expansion dans la requête originale, le besoin d'information original peut être biaisé.

Regardons un exemple d'expansion de requête pour illustrer ce phénomène. Supposons que nous avons une requête initiale qui est composée de deux termes, soient les termes A_0 et B_0, représentant deux aspects importants de la requête.

$$[A_0, \quad B_0]$$

Nous allons étendre notre requête. Nous obtenons alors des termes fortement liés à A_0 à l'aide d'une ressource, par exemple un thésaurus manuel ou un thésaurus statistique. Ces termes sont dénotés par A_1, A_2, A_3 et A_4. Nous voulons aussi des termes d'expansion pour B_0, mais ce même thésaurus ne

trouve rien pour le terme B_0. Nous avons donc ici une expansion de requête où un terme est plus étendu que l'autre, le terme A_0 est étendu avec 4 termes alors que l'autre terme B_0, n'est pas étendu du tout.

Ensuite, nous construisons la requête d'expansion en ajoutant directement les termes d'expansion obtenus dans la requête originale. Nous obtenons alors la requête étendue suivante (sans considérer les poids, pour raison de simplicité) :

$$[A_0, \quad B_0, \quad A_1, \quad A_2, \quad A_3, \quad A_4]$$

Nous pouvons observer que la requête étendue a en tout 6 dimensions dont 5 dimensions concernent l'aspect A, mais seulement une dimension qui concerne l'aspect B.

Maintenant, regardons ce qui se passe lors du calcul de la similarité de cette requête étendue avec les documents. Deux cas peuvent se présenter :

1. Les termes A_0, A_1, A_2, A_3 et A_4 ne peuvent pas apparaître dans un même document.

   Si ces termes sont des vrais synonymes, alors il est peu probable qu'ils se retrouvent dans un même document. Par exemple, les deux termes "cosmonaute" et "astronaute" sont des vrais synonymes. Il est peu probable qu'un auteur choisisse d'utiliser ces deux termes dans le même document, c'est-à-dire qu'il n'emploiera pas tantôt "cosmonaute" et tantôt "astronaute" pour se référer à la même personne. Pour être cohérent et clair, l'auteur va choisir d'utiliser le même terme tout au long du

document. Les vrais synonymes apparaissent rarement ensemble dans le même document. Donc dans ce cas-là, nous pouvons considérer qu'il n'y a aucun document qui contient plus qu'un de ces termes. Alors la requête étendue est équivalente à un ensemble d'alternatives formées par les vecteurs suivants : $[A_0, \quad B_0]$, $[A_1, \quad B_0]$, $[A_2, \quad B_0]$, $[A_3, \quad B_0]$, $[A_4, \quad B_0]$.

Dans ce cas particulier, quand les termes d'expansion ne peuvent pas apparaître ensemble dans un même document, l'ajout direct des termes d'expansion dans la requête originale ne pose pas de problème.

2. Les termes A_0, A_1, A_2, A_3 et A_4 peuvent apparaître dans un même document. En fait, les vrais synonymes sont rares.

   Comme nous avons vu dans la revue de littérature, très souvent, les relations statistiques calculées à partir de la co-occurrence des termes dans les documents, sont utilisées pour trouver les termes d'expansion. Pour notre exemple, cela veut dire que les termes A_1, A_2, A_3 et A_4 sont des termes qui apparaissent très souvent avec le terme A_0 dans les documents. Par conséquent, il y a dans la collection des documents qui contiennent plusieurs des termes A_0, A_1, A_2, A_3 et A_4.

   Lors de l'évaluation, tous ces termes sont considérés, la similarité est calculée en additionnant le poids de tous ces termes. Ainsi pour un document qui contient plusieurs aspects de A, tous les aspects sont additionnés de sorte que pour ce document l'aspect A prendra une importance beaucoup plus grande que l'aspect B. C'est pourquoi dans

une requête étendue, les termes les plus étendus vont implicitement obtenir une importance plus grande par rapport aux autres termes qui sont moins étendus, parce que le poids des termes d'expansion est additionné.

Cette mise en valeur gratuite dans la requête étendue créée par un nombre de termes d'expansion non balancés peut potentiellement biaiser le besoin d'information original. Par exemple, supposons que la requête originale contient deux termes qui sont d'une importance égale : avec A_0 = "voiture" et B_0 = "électrique" et supposons que les termes d'expansion sont A_1 = "char", A_2 = "automobile", A_3= "auto" et A_4= "bagnole". La requête étendue contient alors les termes suivants :

$$[voiture,\ électrique,\ char,\ automobile,\ auto,\ bagnole]$$

Nous pouvons clairement voir que le concept de "voiture" est mis en valeur dans la requête étendue.

L'ajout direct des termes d'expansion dans le vecteur de la requête peut biaiser le besoin d'information en donnant plus de poids aux termes qui ont plus de termes d'expansion.

3.3.3 Inconsistance de l'ajout direct

En ajoutant directement les termes d'expansion dans la requête initiale, les termes d'expansion deviennent indépendants des termes originaux. Cette indépendance est le problème fondamental de l'ajout direct.

Le but de l'expansion de requête est d'étendre la couverture de recherche pour aller chercher un plus grand nombre de documents pertinents. Donc il est évident qu'il faut que les termes d'expansion soient fortement reliés aux termes originaux de la requête. Par conséquent, ces termes d'expansion sont dépendants des termes originaux, parce qu'ils sont obtenus à partir des termes originaux. Cependant, quand la requête étendue est évaluée, toutes les dimensions sont considérées indépendantes ; ainsi lors de l'évaluation, il n'y a plus de relation entre les termes originaux et les termes d'expansion, et le poids de tous les termes est additionné pour le calcul de similarité.

À cause de l'indépendance des termes lors de l'évaluation, une requête étendue comme celle-ci :

$$[\textit{voiture},\ \textit{électrique},\ \textit{automobile},\ \textit{auto},\ \textit{char},\ \textit{bagnole}]$$

peut trouver des documents qui concernent seulement "voiture", mais pas "électrique". Et plus on ajoute de termes pour compléter "voiture", mais rien pour "électrique", plus cet effet de biais est accentué, parce que dans cette requête, il y a beaucoup plus de termes qui sont reliés au le concept de la "voiture" plutôt qu'à celui de "l'électrique", il y a un déséquilibre.

Dans l'ajout direct, nous observons une contradiction dans le pricipe : d'un côté, lors de l'expansion, les termes d'expansion sont dépendants de termes originaux, de l'autre côté lors de l'évaluation, les termes d'expansion sont considérés indépendants. À cause de cette inconsistance, le système retourne des documents qui ne satisfont que partiellement le besoin d'information.

Les études précédentes ont obtenu des résultats variés avec l'expansion de requête. Voorhees [Voo94] a testé l'expansion de requête dans la collection de TREC 2 en utilisant le thésaurus WordNet [Mil95]. Elle a réussi seulement à améliorer la performance des requêtes courtes de façon marginale. Mais pour les requêtes longues, l'expansion des requêtes ne produit pas de résultats significatifs. En enrichissant WordNet avec des thesauri statistiques et des dépendances syntaxiques, Mandala [MTT98] a montré que l'expansion de requête peut améliorer grandement l'efficacité de la recherche d'informantion.

Dans notre étude, nous nous sommes demandés si l'ajout direct était raisonnable et si nous pouvions trouver une autre façon d'intégrer les termes d'expansion dans le modèle vectoriel.

3.4 Solution proposée

Nous venons de voir que le problème de biais vient d'un déséquilibre dans le nombre de termes d'expansion. Et comme lors de l'évaluation, ces termes sont considérés indépendants, les concepts qui ont plus de termes d'expansion sont mis en valeur.

Pour éliminer ce biais, il faut éliminer le déséquilibre. Une solution possible est d'étendre chaque terme initial par un nombre constant de termes d'expansion. Le problème de cette solution est que les termes sont très différents les uns des autres. Il y a des termes qui ont beaucoup de termes d'expansion, alors qu'il y a d'autres qui n'ont pas ou peu de termes d'expan-

sion. Donc cette solution ne résout pas complètement le problème.

Une autre solution possible est de combiner un terme et ses termes d'expansion dans une expression qui représente ensemble une facette. En outre, nous considérons que chaque terme dans la requête initiale correspond à une facette différente. De cette manière, l'expansion d'une requête va renforcer l'expression de chaque facette, mais le nombre de facettes ne sera pas changé. Il est alors possible de conserver l'équilibre entre les différentes facettes, et de ne pas biaiser le rapport entre ces facettes dans la requête initiale. Nous allons voir comment faire ceci plus tard.

Discutons d'abord de notre hypothèse que chaque mot dans une requête représente une facette différente. Cette hypothèse n'est pas toujours vraie comme dans la requête "voiture et bagnole". Mais ce genre de requête avec des concepts répétés est relativement peu fréquent. Dans la plupart des cas un concept n'est décrit qu'une seule fois, surtout quand la requête est courte comme celle utilisée sur le Web. Donc, notre hypothèse est raisonnable.

Pour combiner un terme et ses termes d'expansion dans une expression, il faut établir une relation appropriée entre ces termes. Nous allons d'abord étudier le genre de relation qui existe entre le terme original et ses termes d'expansion. Ensuite, nous allons proposer une nouvelle façon d'intégrer les termes d'expansion dans la requête étendue au lieu de l'ajout direct.

3.4.1 Relation entre terme original et termes d'expansion

Dans cette section nous cherchons à identifier quelle est la relation qui existe entre un terme original et ses termes d'expansion. Intuitivement, nous pensons que les termes d'expansion sont des alternatives du terme original pour décrire la même facette.

Dans notre exemple précédent, les termes d'expansion du terme "voiture" sont "char", "automobile", "auto", "bagnole". En observant ces termes d'expansion, nous pouvons remarquer que ces termes d'expansion sont des équivalents du terme original, c'est-à-dire que le terme original peut être substitué par ces termes d'expansion. À la place du terme "voiture", nous pouvons utiliser le terme "automobile". La requête [*automobile*, *électrique*] est équivalente à la requête [*voiture*, *électrique*]. Les termes d'expansion sont donc des alternatives du terme original. Par conséquent, l'opérateur logique OU est tout indiqué pour connecter un terme original avec ses termes d'expansion.

Pour notre exemple, le terme "voiture" et ses termes d'expansion peuvent être mis ensemble pour former une expression comme celle-ci :

$$(voiture \vee char \vee automobile \vee auto \vee bagnole)$$

Les cinq termes sont placés ensemble dans une expression logique. L'expression représente une seule facette, le concept de voiture. Ce concept peut être exprimé de cinq façons possibles. L'opérateur logique OU permet d'exprimer les alternatives dans l'expression.

3.4.2 Intégration de l'opérateur logique OU dans la requête étendue

Dans cette section, nous montrons comment notre requête étendue est construite.

Pour construire la requête étendue, au lieu de l'ajout direct, nous allons combiner le terme original et ses termes d'expansion dans une expression, en les connectant avec l'opérateur logique OU de la façon décrite dans la section précédente. Le terme original et ses termes d'expansion sont mis dans la même dimension. Ainsi nous n'ajoutons pas d'autres facettes. Pour l'exemple précédent, la structure de la requête étendue devient :

$$[\,(\textit{voiture} \vee \textit{char} \vee \textit{automobile} \vee \textit{auto} \vee \textit{bagnole}), \textit{électrique}\,]$$

La requête étendue possède deux facettes comme dans la requête ogirinale. L'expression composée des termes reliés par OU représente un seul aspect de la requête. L'expression connectée par OU indique que l'aspect "voiture" peut être exprimé par l'un des cinq termes. L'importance relative entre les deux aspects, "voiture" et "électrique" est conservée. Le changement que nous apportons au vecteur est qu'il est maintenant possible de grouper plusieurs termes reliés entre eux dans une expression et de la mettre dans une seule facette du vecteur. Ainsi les termes d'expansion sont mis dans la même facette que le terme original, ce qui permet de conserver l'importance relative entre les différentes facettes. Dans nos expériences, les synonymes qui sont dans la même facette que le terme original proviennent de WordNet ou du thésaurus

construit à partir l'information mutuelle. Nous donnerons plus de détails au chapitre 4.

3.5 Évaluation des opérateurs logiques

Dans cette section nous allons décrire comment nous évaluons la requête étendue qui intégre l'opérateur logique OU. Il y a deux problèmes à résoudre :

1. Comment évaluer la relation OU ?
2. Comment intégrer cet opérateur logique dans le modèle vectoriel ?

Nous allons répondre à ces deux questions dans les deux sections suivantes.

3.5.1 Évaluation de la relation OU

La relation OU peut être évaluée en utilisant les probabilités ou en utilisant la théorie des ensembles flous.

En probabilité, la probabilité de A OU B est déterminée comme suit :

$$P(A \vee B) = P(A) + P(B) - P(A \wedge B) \tag{3.1}$$

En considérant que A et B sont indépendants, dans ce cas là, nous avons :

$$P(A \vee B) = P(A) + P(B) - P(A) \times P(B) \tag{3.2}$$

Cette formule correspond à la première façon possible d'évaluer une facette en forme de disjonction.

Une autre façon est d'utiliser la théorie des ensembles flous [Zad65]. Dans cette théorie la relation OU est souvent évaluée de cette façon :

$$P(A \vee B) = Max(P(A), P(B)) \tag{3.3}$$

En comparant les formules 3.2 et 3.3 entre elles, nous pouvons remarquer que la première formule 3.2, est plus dépendante de A et de B à la fois que la deuxième. En effet, elle tient compte de la contribution des deux éléments dans la formule. Par contre, la deuxième formule 3.3, considère seulement le plus important des deux éléments, la contribution de l'autre élément est ignorée.

Il est difficile de choisir d'utiliser une formule plutôt que l'autre. C'est pourquoi, nous avons testé les deux formules dans nos expériences.

3.5.2 Intégration de l'opérateur OU dans le modèle vectoriel

Dans cette section, nous allons montrer comment intégrer l'évaluation de la relation OU dans le calcul de similarité du modèle vectoriel. Le changement apporté au modèle vectoriel est que la dimension de la requête étendue peut maintenant contenir une expression logique composée de termes connectés entre eux par OU. Nous pensons qu'il y a deux façons possibles pour évaluer cette expression.

1. La requête étendue peut être transformée en une combinaision logique de vecteurs connectés par OU. Par exemple, la requête étendue (A OU

B, C) est équivalente à (A, C) OU (B, C). Cette solution n'est pas efficace. Comme nous pouvons le voir dans l'exemple précédent, C se retrouve dans les deux élements de la disjonction ; ce qui entraîne que C doit être évalué plusieurs fois et le nombre de vecteurs créés ainsi peut être considérable si la requête de départ est longue.

2. Nous pouvons aussi évaluer directement chaque dimension de la requête étendue. Quand la dimension est une expression logique, nous calculons une seule valeur de similarité pour cette dimension en utilisant une des formules qui permettent d'évaluer la relation OU et en tenant compte de chaque terme de l'expression. Par exemple, supposons que nous utilisons la formule 3.3 pour l'évaluation de la relation OU. Pour la requête étendue suivante, $[(A \vee B), \quad C]$, le poids pour la première facette est déterminé par $Max(P(A), P(B))$ et le poids pour la deuxième dimension par $P(C)$. Ensuite, le produit interne utilise ces deux valeurs pour le calcul de similarité : $Max(P(A), P(B)) + P(C)$. De cette façon, chaque terme dans la requête étendue ne sera évalué qu'une seule fois.

Nous avons choisi d'implanter la deuxième solution en utilisant le fichier inversé car elle est plus efficace. Nous utilisons l'algorithme suivant pour obtenir un ensemble de documents pondérés pour une requête étendue.

Dans cet algorithme, la fonction `Union()` désigne l'union de deux listes de documents pondérés en additionnant le poids des éléments en commun des deux listes. La fonction `Combiner()` désigne la combinaison de deux listes

```
EvalReq(Req)
{
    liste = null;
    Pour chaque dimension Dim dans Req
        {
            docliste = EvalDimension(Dim);
            liste = Union( liste, docliste);
        }
    return liste;
}

EvalDimension(Dim)
{
    liste = null;
    Pour chaque terme Trm dans Dim
        {
            listeTrm = liste des documents qui contiennent Trm;
            liste = Combiner(liste, listeTrm);
        }
    return liste;
}
```

FIG. 3.1 – Algorithme pour obtenir un ensemble de documents pondérés

de documents pondérés en combinant le poids des éléments en communs en appliquant l'une des deux formules 3.2 ou 3.3 pour évaluer l'opérateur logique OU.

3.5.3 Discussions

Il est intéressant de remarquer que l'algorithme décrit en haut est une généralisation de l'évaluation classique du produit interne du modèle vectoriel. En effet, pour reproduire l'évaluation classique, il suffit d'utiliser la formule suivante pour évaluer la relation OU :

$$P(A \vee B) = P(A) + P(B)$$

Le modèle vectoriel a un grand désavantage par rapport au modèle booléen. En effet, dans le modèle vectoriel, il n'est pas possible de formuler des requêtes en incluant des opérateurs booléens. Par exemple, dans le modèle vectoriel, les requêtes "guerre ET mondiale" et "vélo OU bicyclette" ne peuvent pas être représentées par des vecteurs de poids.

En 1983, Gerard Salton, Edward A. Fox et Harry Wu [SFW83] ont introduit le modèle p-norme qui permet de générer le modèle vectoriel et le modèle booléen comme deux extrêmités. Le modèle p-norme possède les caractéristiques des deux modèles, il permet de formuler des requêtes avec des opérateurs logiques et de pondérer les termes des documents ainsi que ceux des requêtes. Donc le modèle p-norme est un modèle plus général. Ce modèle est intéressant car il permet d'unifier le modèle vectoriel avec le modèle booléen.

Le modèle que nous proposons présente une autre façon d'intéger le modèle booléen et le modèle vectoriel. Cette intégration nous permet d'obtenir les avantages de ces deux modèles : d'un côté, l'expression disjonctive nous permet d'exprimer la relation d'alternative entre les termes d'expansion et le terme initial. D'un autre côté, nous bénificions d'une évaluation vectorielle globale nous permettant de bien classer les documents dans la réponse. Par rapport à l'expansion de la requête par l'ajout direct, notre approche permet d'éviter le biais créé durant l'expansion.

3.6 Expansion incertaine

Jusqu'ici, nous avons omis les pondérations de termes dans l'expansion de la requête, nous nous sommes plutôt concentrés sur le principe. Mais la pondération est un aspect important de l'approche proposée De façon générale, nous avons deux pondérations :

- la pondération pour décrire l'importance d'un terme dans la requête et dans le document.
- la pondération pour indiquer dans quelle mesure un autre terme est relié à un terme donné.

Par exemple, soit B un terme de la requête originale, P_B le poids accordé au terme B et A un terme relié à B. Le terme A est un alternatif à B à un degré α, où $\alpha \leq 1$. Dans ce cas, nous considérons que le poids du terme alternatif A est $P_B \times \alpha$.

L'algorithme pour évaluer une dimension devient alors :

```
EvalDimension(Dim)
{
    liste = null;
    Pour chaque terme Trm d'incertitude Alpha dans Dim
        {
            listeTrm = liste des documents qui contiennent Trm;
            chaque document de listeTrm est multiplié par Alpha;
            liste = Combiner(liste,  listeTrm);
        }
    return liste;
}
```

FIG. 3.2 – La procedure EvalDimension en intégrant l'incertitude alpha

Chapitre 4

Implantation de l'expansion de requête

Ce chapitre présente et discute des détails de l'implantation de l'expansion de requête en intégrant l'opérateur logique OU, dont le principe a été conçu dans le chapitre précédent.

L'expansion de requête est implantée dans SMART avec le langage C sous le système d'exploitation Linux. Nous allons d'abord présenter le processus d'indexation et d'évaluation de SMART. Puis nous allons expliquer notre implantation d'expansion de requête avec WordNet, avec l'information mutuelle et le "relevance feedback".

Puis nous allons discuter plus en détail de chacune de ses composantes.

4.1 Architecture du système de recherche d'information SMART

SMART est conçu pour être flexible et facile à modifier. Il permet de sélectionner des procédures pendant l'exécution du programme au moyen d'un fichier de spécification. Par exemple, il est possible d'écrire plusieurs procédures pour faire la lemmatisation. Il suffit d'indiquer dans le fichier de spécification laquelle des procédures de lemmatisation nous voulons utiliser. Nous modifions la version 11 de SMART pour ajouter l'expansion de requête. Cette version de SMART a été conçue et implementée par C. Buckley [Buc85]. Elle contient environ 44 000 lignes de code C. Nous faisons rouler SMART sous Linux avec 2GB de mémoire vive et 66GB de disque dur.

4.1.1 Indexation et évaluation de requête

Avant de pouvoir étendre les requêtes et de les évaluer dans SMART, il est nécessaire d'indexer les documents. Le but de l'indexation est de créer un représentation interne des documents et de déterminer pour chaque document les concepts les plus importants.

La figure 4.1 montre le processus d'indexation et d'évaluation dans SMART. Durant l'indexation des documents et des requêtes, SMART crée des vecteurs de poids pour les représenter. Ensuite, à partir de ces vecteurs de documents, SMART construit un fichier inversé qui correspond à un index, pour chaque terme apparaissant dans la collection, il liste les documents contenant ce

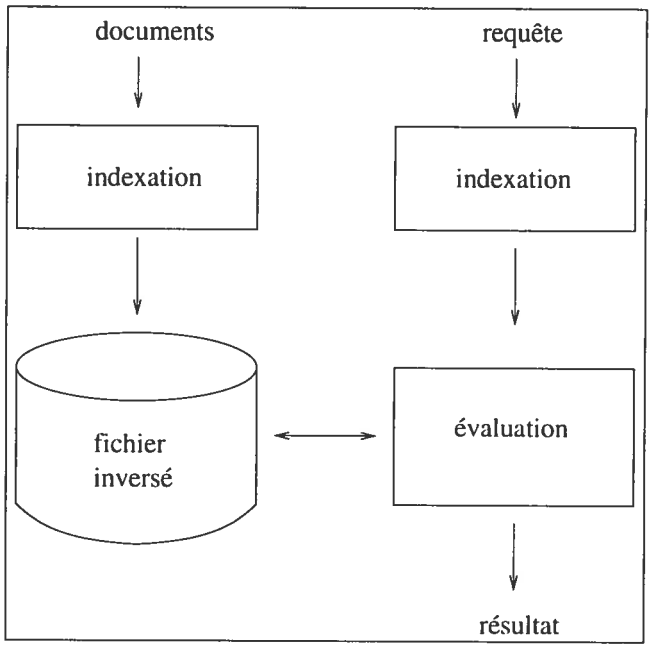

FIG. 4.1 – Indexation et évaluation

terme et leurs pondérations.

Après avoir créé le fichier inversé, découlant des vecteurs des documents et les vecteurs des requêtes, SMART peut commencer à évaluer les requêtes. Durant l'évaluation d'une requête, le fichier inversé est consulté pour obtenir une liste de documents pondérés qui contiennent les termes de la requête. Ensuite, le vecteur de la requête est comparé avec les vecteurs de ces documents pour calculer une valeur de similarité pour chacun de ces documents. Comme, un même document peut obtenir plusieurs valeurs de similarité, puisqu'il contient plusieurs termes de la requête, ses valeurs doivent être additionnées (selon le calcul de similarité par le produit interne). Enfin, ces

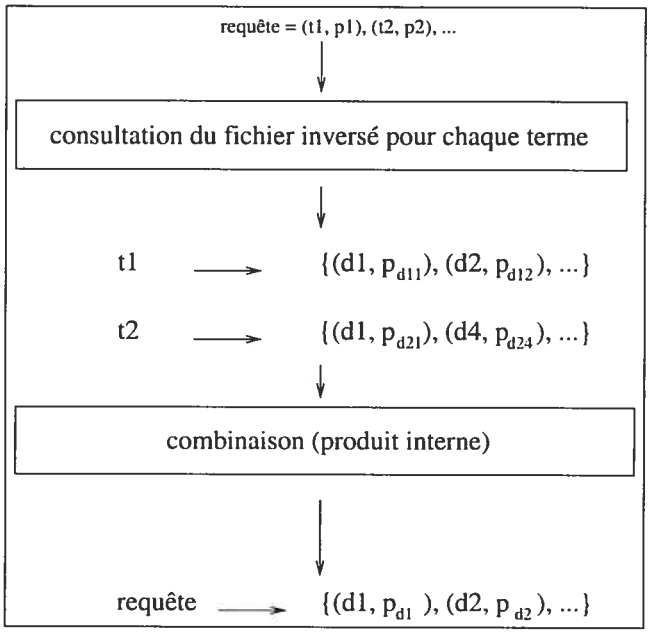

FIG. 4.2 – Évaluation d'un requête dans SMART

documents sont triés dans l'ordre décroissant en utilisant cette valeur. La figure 4.2 illustre l'évaluation d'une requête dans SMART avec de plus amples détails.

L'expansion de requête intervient lors de l'évaluation de la requête. Deux nouvelles composantes ont été ajoutées dans l'évalutation, soit la composante "expansion" et la composante "fusion avec OU" (voir figure 4.3).

La composante "expansion" intercepte d'abord la requête originale. Ensuite, à partir de cette requête, elle sélectionne des termes d'expansion, calcule un poids pour ces termes d'expansion et crée la requête étendue. Puis, cette requête étendue est donnée au fichier inversé pour trouver les documents qui

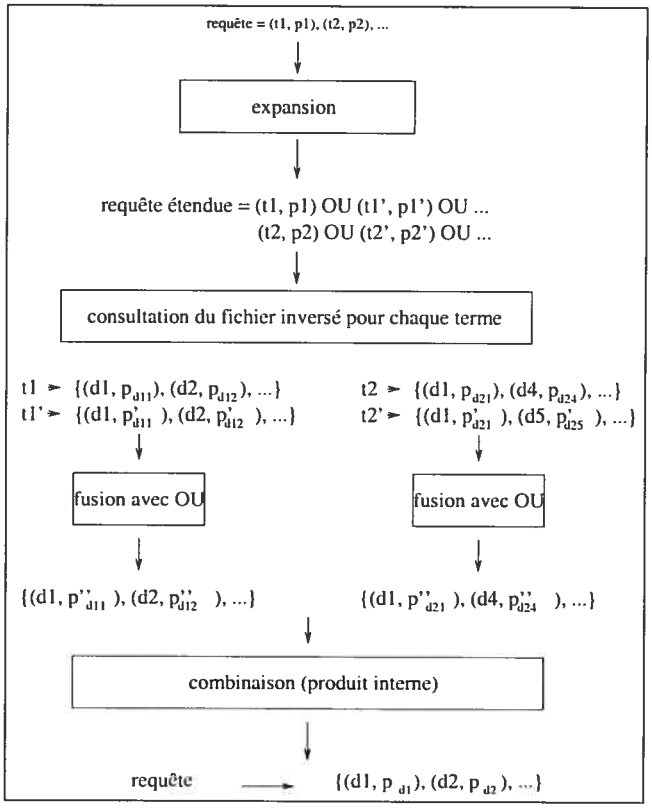

FIG. 4.3 – Évaluation d'une requête avec expansion

contiennent les termes de la requête étendue.

La composante "fusion avec OU" intercepte la liste des documents et la fusionne avec la liste obtenue à partir des termes précédents de la disjonction en utilisant l'une des formules d'évaluation de OU vues dans le chapitre précédent.

4.2 Implantation de l'expansion de requête

La procédure génénale qui implante la composante d'expansion de requête a pour paramètre un terme pondéré et retourne un ensemble de termes pondérés.

Dans SMART un terme pondéré de la requête est une structure `CON_WT`.

```
typedef struct {
    long con;                    /* Actual concept number */
    float wt;                    /* and its weight */
} CON_WT;
```

où `con` est le numéro d'identification du terme, et `wt` son poids. Une requête est représentée par une suite de `CON_WT`, triée dans l'ordre de `con`. C'est une forme de représentation pour le vecteur.

L'algorithme de l'expansion est le suivant :

1. sélectionner des termes d'expansion pour chaque terme original de la requête.
2. calculer un poids pour les termes d'expansion
3. créer une requête étendue en mettant tous ces termes dans un vecteur.

L' expansion utilise trois différentes ressources pour obtenir les termes d'expansion. Les trois ressources que nous avons utilisées sont WordNet, le "pseudo relevance feedback" et un thésaurus statistique basé sur l'information mutuelle. Nous allons présenter ces trois implantations.

4.2.1 WordNet

Le thésaurus WordNet dispose d'un ensemble de fonctions permettant d'accéder aux données. Dans WordNet, la fonction `findtheinfo_ds` permet d'obtenir les termes d'expansion. Cette fonction est assez complexe parce qu'elle a beaucoup de paramètres. Voici le prototype de la fonction :

```
SynsetPtr findtheinfo_ds(char *searchstr, int pos, int ptr_type,
int sense_num ) ;
```

Le premier argument "`searchstr`" est le terme pour lequel nous voulons chercher des termes d'expansion. Le deuxième argument "`pos`" (part of speech) est le groupe grammatical dans lequel nous voulons les termes d'expansion. Les quatre groupes grammaticaux possibles sont nom, verbe, adjectif, et adverbe. Nous utilisons seulement les noms. Le troisième argument "`prt_type`" est le type de recherche que nous voulons effectuer, par exemple les synonymes ou les hyperonymes. Et le dernier argument "`sense_sum`" indique lequel des sens du terme "`searchstr`" nous désirons obtenir des termes d'expansion. Nous utilisons tous les sens (ALLSENSE) dans notre cas car nous n'avons pas de moyen de sélectionner un sens particulier. Cette fonction retourne une liste chaînée d'une structure de donnée représentant un "synset". Ce dernier est la structure de base de WordNet, il représente un ensemble de synonymes. Une fois, une telle liste de "synsets" est obtenue, nous parcourons la liste pour obtenir les termes de tous les "synsets".

4.3 Extraction de l'information mutuelle

Pour pouvoir utiliser l'information mutuelle, il a fallu créer un thésaurus statistique. Nous avons donc construit un thésaurus statistique en se basant sur l'information mutuelle. Les difficultés rencontrées dans la construction d'un tel thésaurus sont souvent reliées à la taille colossale de la collection.

Pour construire le thésaurus statistique basé sur l'information mutuelle, nous avons besoin de l'information sur les co-occurrences de termes. Or dans SMART, quand les documents sont indexés, la position des termes n'est pas gardée. Ce qui ne nous permet pas d'exploiter les co-occurrences dans une fenêtre de 20 mots. C'est pour cette raison que nous avons dû utiliser Lemur [Pro] qui, lui, sauvegarde cette information.

Lemur est un ensemble d'outils conçus pour faciliter l'étude de la recherche d'information. Ces outils peuvent indexer de très grandes collections de textes, rechercher de l'information et en faire l'évaluation. Lemur est un système de recherche d'information écrit en langage C et C++, pour les systèmes d'exploitation Unix.

Lemur ne calcule pas l'information mutuelle, mais nous utilisons sa librairie de fonctions pour extraire les informations nécessaires de la collection afin de calculer l'information mutuelle entre deux termes quelconques . La collection est d'abord indexée avec Lemur, et les informations sont extraites par un programme dédié (écrit en C++). La méthode `termInfoListSeq()` de la classe `InvFPIndex` est utilisée pour obtenir les termes d'un document

dans l'ordre qu'ils apparaissent dans le document.

Les données extraites sont écrites dans un fichier.

La figure 4.4 expose le contenu du fichier. Le fichier est divisé en trois

```
61528413
273218
62292400
report  1       175113
saigon  2       467
official        3       174555
release 4       57967
education       5       20511
camp    6       15525
150     7       11632
officer 8       43190
overthrow       9       3567
south   10      65839
vietnamese      11      6203
government      12      200966
13      13      33351
year    14      412913
detention       15      2973
...
38586   4000    2
38586   4004    1
38586   4034    1
38586   4054    1
38586   4056    8
38586   4073    3
38586   4074    1
...
```

FIG. 4.4 – Informations extraites du corpus pour le calcul de l'information mutuelle

parties :

1. La première partie est composée de trois nombres : la taille du corpus selon le nombre de termes qu'il contient, le nombre de termes uniques dans le corpus et le nombre de co-occurrences créées.
2. La deuxième partie liste chaque terme unique. Pour chaque terme, trois

renseignements sont gardés, le terme, son identificateur numérique et le nombre de fois qu'il apparaît dans le corpus.

3. La troisième partie concerne les co-occurrences. Chaque ligne contient le premier terme, le deuxième terme et le nombre de fois que cette co-occurence est observée dans le corpus.

Nous nous sommes intéressés à la co-occurrence des termes dans une fenêtre de vingt termes. Voici l'algorithme utilisé par le programme pour créer les co-occurrences.

```
pour chaque fenêtre
    identifier les différentes co-occurences dans cette fenêtre
    pour chaque co-occurrence
        si la co-occurence est déjà rencontrée
            incrémenter le nombre d'observation de 1
        sinon
            créer cette co-occurence
            le nombre d'observation est égale à 1
avance la fenêtre d'un terme pour créer un autre fenêtre
```

FIG. 4.5 – Algorithme utilisé pour créer les co-occurrences

4.4 "Pseudo relevance feedback"

Dans le "pseudo relevance feedback", les termes d'expansion proviennent des premiers documents trouvés par le système après une première recherche.

Dans SMART, après avoir fait une recherche, la fonction `get_seen_docs` est d'abord appelée pour obtenir les premiers documents qui ont été trouvés.

Puis la fonction `did_vec` est appelée pour chacun de ces documents afin de recueillir les termes de ces documents.

Enfin, les termes identiques sont combinés en additionnant leur poids, les termes qui ont les poids les plus importants deviennent les termes d'expansion.

4.5 Synthèse

Dans ce chapitre, nous avons examiné en détail l'implantation de l'expansion de requête avec les différents ressources. Le prochain chapitre présente les résultats obtenus avec cette implantation.

Chapitre 5

Expérimentations et résultats

Dans ce chapitre, nous présentons et analysons nos expérimentations et les résultats obtenus avec l'expansion de requête à partir des trois différentes ressources et la nouvelle méthode d'intégration des termes d'expansion dans le vecteur de la requête étendue.

Nous allons d'abord décrire le corpus, puis les évaluations. Enfin, nous décrirons en détail les résultats obtenus avec chaque évaluation.

5.1 Description du corpus de test

Pour nos évaluations, nous utilisons la collection AP de TREC 7. Cette collection de 525M contient 242 918 documents en anglais. Elle contient aussi 28 requêtes ainsi que la liste des documents pertinents qui les correspondent. Les documents dans cette collection sont des articles de journaux qui proviennent de l'agence de presse américaine "Associated Press". Ces articles ont

été recueillis par les laboratoires de Bell AT&T entre les années 1988 et 1990. Les documents se présentent sous le format standard SGML.

Un document commence par le marqueur <DOC> et termine par le marqueur </DOC>. Le titre d'un article se trouve entre les marqueurs <HEAD> et </HEAD>. Le corps de l'article est délimité par les marqueurs <TEXT> et </TEXT>. Il y a d'autres marqueurs dans l'entête du document, mais pour nos expériences , nous utilisons seulement le titre et le texte de l'article (voir figure 5.1).

```
<DOC>
<DOCNO> AP880212-0004 </DOCNO>
<FILEID>AP-NR-02-12-88 1637EST</FILEID>
<FIRST>r w AM-PeanutSupports     02-12 0155</FIRST>
<SECOND>AM-Peanut Supports,150</SECOND>
<HEAD>Peanut Price Supports Will Go Higher This Year</HEAD>
<DATELINE>WASHINGTON (AP) </DATELINE>
<TEXT>
   Price supports for peanuts grown under 1988
quotas will be $615.27 per ton, an increase of $7.80 from last
year, the Agriculture Department said Friday.
   Deputy Secretary Peter C. Myers said the increase was required
by a formula in the law which takes rising production costs into
consideration.
   The annual quota is set at a level equal to the estimated
quantity of peanuts that will be needed for domestic edible uses,
seed and related purposes.
   Production of non-quota peanuts, which can be grown for peanut
oil and meal, and for export, will be supported at $149.75 per ton,
unchanged from last year, Myers said.
   In setting the support for non-quota peanuts, officials are
required to consider certain factors, including the demand for oil
and meal, the expected prices for other vegetable oils and meals,
and the foreign demand for peanuts.
</TEXT>
</DOC>
```

FIG. 5.1 – Un exemple de document de la collection AP

Les requêtes sont identifiées par un numéro qui suit le marqueur <num>

et elles sont séparées par le marqueur <top> (voir figure 5.2). Chaque requête contient un titre, une description et une narration. Nous utilisons deux types de requête. La requête courte qui contient seulement le champ titre, et la requête longue qui contient les champs titre et description.

```
<top>
<num> Number: 33
<E-title>
Genetic Engineering
<E-description>
What are the latest development in the agricultural
application of genetic engineering?
<E-narr>
...
```

FIG. 5.2 – La requête numéro 33

Les documents et les requêtes sont indexés avec le système SMART [Buc85] en utilisant un schéma de pondération ltc. Dans ce schéma, le poids w_{ik} d'un terme T_k dans un document D_i est calculé avec cette formule [BSAS94] :

$$w_{ik} = \frac{(log(f_{ik}) + 1.0) \times log(N/n_k)}{\sqrt{\sum_{j=1}^{t}[(log(f_{ij}) + 1.0) \times log(N/n_j)]^2}} \tag{5.1}$$

où f_{ik} est le nombre de fois que le terme T_k apparaît dans le document D_i, N est la taille de la collection et n_k est le nombre de documents qui contiennent le terme T_k.

Le schéma ltc est une forme de $tf * idf$. Nous pouvons remarquer que $log(f_{ik})+1.0$ correspond à tf, $log(N/n_k)$ correspond à idf et le dénominateur

permet de normaliser le poids, ce qui permet de reproduire la formule de similarité cosinus en utilisant le produit interne.

5.2 Évaluation

Dans cette section, nous allons décrire les évaluations ainsi que les objectifs de chaque évaluation.

5.2.1 Description

Dans toutes nos évaluations, la collection et les requêtes sont indexées et pondérées de la même façon. Les seuls éléments qui varient lors de l'expansion de requête sont la source des termes d'expansion, le poids de ces termes, la longueur des requêtes utilisées, et la façon dont les termes d'expansion sont intégrés dans la requête.

La précision moyenne est souvent utilisée pour mesurer la performance d'un système en recherche d'information. Cette précision décrit bien la performance du système. Nous utilisons la précision moyenne sur les 11 points de rappel pour mesurer la performance de l'expansion de requête. Pour chaque requête, nous utilisons les 1 000 premiers documents trouvés par le système pour calculer la précision moyenne sur les 11 points de rappel. C'est la façon standard utilisée dans le domaine de recherche d'information. La précision moyenne est calculée comme suit : Pour chacune des valeurs de rappel (0.00, 0.10, 0.20 ...1.00), une précision est calculée à l'aide de la liste des documents

pertinents et de la liste des documents retournés par le système. La précision moyenne est calculée en faisant la moyenne de ces 11 valeurs de précision (voir tableau 5.1).

Tous les résultats des expansions sont comparés avec les résultats du test de référence pour lequel les requêtes ne sont pas étendues. L'amélioration ou la réduction de la performance de chaque évaluation sera calculée en pourcentage.

5.2.2 Objectifs

Le but de nos expériences consiste à évaluer l'effet de l'expansion de requête par différentes méthodes. Pour ce faire, nous commençons par déterminer la performance du système sans l'expansion de requête : le test de référence.

Ensuite, nous allons évaluer l'expansion de requête avec WordNet. Nous voulons connaître l'impact des différentes relations sémantiques de WordNet dans l'expansion de requête. Nous voulons aussi connaître l'effet de la longueur des requêtes sur l'expansion de requête dans WordNet, ainsi que le poids des termes d'expansion.

Puis nous allons examiner l'expansion de requête avec l'information mutuelle. Ici aussi, nous voulons connaître l'influence de la longueur des requêtes sur la performance de l'expansion de requête, ainsi que le poids des termes d'expansion. Nous allons examiner l'impact du nombre de termes d'expan-

sion dans l'expansion de requête. Pour cela, nous allons limiter le nombre de termes d'expansion de deux façons.

- Nous utilisons un nombre fixe de termes d'expansion.
- Nous fixons un seuil sur le poids des termes d'expansion.

Comme nous avons déjà discuté au chapitre 2, il y a deux formules pour calculer l'information mutuelle. Nous désirons aussi comparer la perfomance de ces deux formules dans l'expansion de requête.

Par ailleurs , nous voulons connaître l'impact de notre nouvelle méthode d'intégrer les termes d'expansion en comparant cette méthode avec la méthode d'ajout direct. Pour ce faire, nous allons tester cette méthode avec WordNet et avec l'information mutuelle.

Finalement, nous allons évaluer la performance de l'expansion de requête avec le "pseudo-relevance feedback".

5.3 Test de référence

Pour le test de référence, les requêtes sont directement évaluées par SMART sans expansion.

Le tableau 5.1 montre les précisions obtenues avec les requêtes courtes et les requêtes longues. Pour chacun des 11 points de rappel (0.00, 0.10 ..., 1.00), une précision est calculée. La précision moyenne est obtenue en faisant la moyenne des ces 11 précisions. Nous pouvons remarquer que les requêtes

Rappels	Requête courte Précisions (%)	Requête longue Précisions (%)
0.00	60.56	66.62
0.10	47.00	49.95
0.20	40.58	44.09
0.30	36.87	39.93
0.40	34.09	37.01
0.50	30.12	33.09
0.60	24.68	27.33
0.70	22.07	25.04
0.80	18.95	22.01
0.90	13.00	15.90
1.00	7.17	10.00
Préc. moy.	30.46	33.72

TAB. 5.1 – Test de référence de la collection AP

longues produisent de meilleurs résultats que les requêtes courtes. La précision moyenne des requêtes longues est supérieure à la précision moyenne des requêtes courtes d'environ 10 %. En effet, les requêtes courtes ont une précision moyenne de 30.46 % tandis que les requêtes longues ont une précision moyenne de 33.72 %. Ce sont les performances de référence avec lesquelles nous allons comparer.

5.4 Expansion avec WordNet

Dans ces expérimentations, le thésaurus WordNet fournit les termes d'expansion pour étendre la requête originale. Nous avons testé plusieurs relations sémantiques dans WordNet : la relation de synonymie, la relation d'hyperonymie et la relation d'hyponymie.

Nous étendons seulement les noms communs des requêtes. Les verbes, les adjectifs et les adverbes sont ignorés. Car la pluspart des requêtes ne sont constituées que de noms communs. Tous les sens d'un nom sont considérés. À partir d'un nom de la requête, nous allons chercher ses "synsets" dans WordNet. À partir de ces "synsets", nous obtenons les synonymes pour tous les sens de ce terme. Puis, en suivant les pointeurs de ces "synsets", nous pouvons obtenir les hyperonymes et les hyponymes. Dans le cas où un nom ne se trouve pas dans WordNet, alors une liste vide est retournée. Les termes d'expansion fournis par WordNet sont pondérés de façon uniforme. Plusieurs valeurs de poids ont été testées (voir tableau 5.2).

Type de requête	Poids des termes ajoutés	Préc. moy. (amél. %) syn. et hype.	Préc. moy. (amél. %) syn. et hypo.
Courte	0.05	**31.43 (3.2)**	31.86 (4.6)
	0.10	31.07 (2.0)	**31.95 (4.9)**
	0.15	30.31 (-0.5)	30.67 (0.7)
	0.20	29.58 (-2.9)	31.13 (2.2)
	0.25	28.24 (-7.3)	30.76 (1.0)
Longue	0.05	33.85 (0.4)	**34.43 (2.1)**
	0.10	33.15 (-1.7)	33.89 (0.5)
	0.15	31.73 (-5.9)	33.15 (-1.7)
	0.20	28.46 (-15.6)	32.34 (-4.1)
	0.25	24.92 (-26.1)	31.36 (-7.0)

TAB. 5.2 – Précisions moyennes avec l'expansion de requête utilisant WordNet

Les résultats obtenus indiquent que la performance se dégrade avec l'augmentation du poids des termes d'expansion. Comme nous pouvons le constater dans le tableau 5.2, avec un poids de 0.25 une dégradation de -26.1 % est observée pour les requêtes longues qui utilisent les hyperonymes, par contre avec un poids plus faible, par exemple de 0.10, la dégradation est moins important, elle est alors de -1.7 %. Une explication possible à ce fait est que les termes d'expansion provenant de WordNet sont ambigus. Si on accorde trop d'importance à ces termes, on introduit beaucoup de bruit dans la requête. Par conséquent, un poids fort ne fait qu'amplifier ce bruit et diminuer la précision moyenne.

L'expansion de requête courtes produit de meilleurs résultats que l'expansion des requêtes longues. En effet, dans le tableau 5.2, tous les résultats obtenus par les requêtes courtes sont supérieurs aux résultats des requêtes longues. De même, la dégradation des résultats provenant des requêtes longues est plus importante que les requêtes courtes quand le poids augmente. Cela peut être expliqué par le fait qu'une requête courte a plus besoin d'être étendue qu'une requête longue.

Le meilleur résultat observé avec les requêtes longues est de 2.1 %, obtenu en utilisant les synonymes et les hyponymes avec un poids de 0.05. Par contre pour les requêtes courtes, le meilleur résultat est de 4.9 % en utilisant les synonymes et les hyponymes avec un poids de 0.10.

Dans les requêtes longues, il y a plus de termes qui sont étendus. Donc un nombre plus important de termes d'expansion est ajouté dans ces requêtes.

Comme les termes d'expansion de WordNet sont ambigus, l'amplitude des bruits ajoutés dans les requêtes longues est plus élevée que dans les requêtes courtes. C'est pourquoi les requêtes courtes obtiennent de meilleurs résultats que les requêtes longues et les requêtes longues sont plus sensibles au poids.

L'expansion de requête avec les hyponymes donne de meilleurs résultats qu'avec les hyperonymes. En effet, tous les résultats utilisant les hyponymes dépassent ceux qui utilisent les hyperonymes. Cette observation est tout à fait intuitive. En effet, quand on cherche le terme "ordinateur", le terme "PC" (un hyponyme) peut être satisfaisant. Dans le sens inverse, le terme "ordinateur" est moins satisfaisant pour une requête "PC".

Globalement, nous n'avons pas observé un grand impact en faisant l'expansion de requête avec WordNet. Ces résultats sont comparables aux résultats obtenus par d'autre chercheurs [Voo94, MTT98].

5.5 Expansion avec l'information mutuelle

Dans ces tests, l'information mutuelle est utilisée pour déterminer les termes d'expansion. Elle fournit les termes qui apparaissent le plus souvent avec le terme en question dans la collection. Nous étendons les requêtes avec ces termes. Regardons deux exemples d'expansion avec l'information mutuelle, pour les requêtes "Famine in Sudan" et "Art Theft" (voir les tableaux 5.3 et 5.4). Nous pouvons constater que les termes retournés par l'information mutuelle sont des termes qui sont fortement liés avec le terme original.

Famine	**Sudan**
relief	*rebel*
sudan	*southern*
million	*government*
people	*relief*
war	*people*
ethiopia	*sudan*
food	*liberation*
die	*khartoum*
civil	*army*
drought	*sudanese*
year	*food*
sudanese	*war*
government	*ethiopia*
country	*famine*
thousand	*mahdi*
aid	*million*

TAB. 5.3 – Termes d'expansion pour la requête "Famine in Sudan"

Art	**Theft**
museum	*charge*
art	*police*
gallery	*steal*
work	*conspiracy*
artist	*count*
entertainment	*arrest*
weekend	*burglary*
exhibit	*car*
painting	*museum*
collection	*crime*
show	*property*
center	*theft*
fine	*art*
contemporary	*stolen*
director	*robbery*
school	*sentence*
exhibition	*auto*

TAB. 5.4 – Termes d'expansion pour la requête "Art Theft"

Information mutuelle	Nombre de terme d'expansion					
	20	25	30	50	60	90
Formule (2.3)	36.86	37.00	36.92	37.50	37.60	35.95
	(9.3)	(9.7)	(9.5)	(11.2)	(11.5)	(6.6)
Formule (2.4)	33.96	33.85	34.06	34.06	34.02	34.06
	0.7	0.4	1.0	1.0	0.9	1.0

TAB. 5.5 – Précisions moyennes obtenues avec les deux formules d'information mutuelle

Le tableau 5.5 montre les résultats obtenus par les deux différentes formules de l'information mutuelle dans laquelle un poids de 0.15 a été attribué aux requêtes longues. La formule d'information mutuelle ponctuelle produit de moins bons résultats, car cette formule accorde une plus grande importance aux termes rares. Par exemple, dans le corpus nous observons le terme "oil" 54282 fois, "price" 89975 fois et APPEC ("Asia Pacific Petroleum Conference") 2 fois, "oil" apparaît avec "price" 16325 fois et "oil" apparaît avec "APPEC" 2 fois. La formule d'information mutuelle ponctuelle nous donne :

$$I(\text{“}oil\text{”}, \text{“}price\text{”}) = 7.684122$$

$$I(\text{“}oil\text{”}, \text{“}APPEC\text{”}) = 10.146563$$

Par contre avec l'autre formule :

$$I(\text{“}oil\text{”}, \text{“}price\text{”}) = 0.002039$$

$$I(\text{“}oil\text{”}, \text{“}APPEC\text{”}) = 0.00000033$$

Nous pouvons constater qu'avec la formule ponctuelle, la relation entre "oil" et "APPEC" est plus forte que la relation entre "oil" et "price". Par contre, avec l'autre formule c'est le contraire, car elle tient compte du nombre de co-occurrences. Comme "oil" apparaît avec "APPEC" 2 fois seulement, la valeur obtenue est très faible. En regardant les termes d'expansion de "oil" qui produisent les plus grandes valeurs avec l'information mutuelle ponctuelle, nous constatons qu'ils ont tous les caractéristiques de "APPEC", ce sont tous des termes qui apparaissent une ou deux fois dans le corpus, et apparaissent avec le terme "oil" une ou deux fois. Ces termes ne sont pas intéressants pour l'expansion de requête. Par conséquent nous utiliserons la formule 2.3 dans le reste de nos tests.

Nous limitons le nombre de termes d'expansion de deux façons :

- par un nombre fixe de termes d'expansion
- par un seuil sur le poids des termes d'expansion

5.5.1 Nombre de termes d'expansion limité par un nombre fixe

Nous pouvons observer que la précision moyenne augmente avec le nombre de termes d'expansion utilisé (voir tableau 5.6).

En ce qui concerne le poids, pour les requêtes courtes, on peut observer que la précision augmente avec le poids jusqu'à 0.20. Par contre, pour les requêtes longues, c'est seulement vrai quand le nombre de termes d'expansion

Type de requête	Poids	Nombre de terme d'expansion						
		10	20	50	100	200	400	600
Courte	0.10	32.31	32.53	33.48	33.84	33.93	34.24	34.48
		(6.1)	(6.8)	(9.9)	(11.1)	(11.4)	(12.4)	(13.2)
	0.15	32.84	33.02	33.45	34.05	34.66	34.85	**35.00**
		(7.8)	(8.4)	(9.8)	(11.8)	(13.8)	(14.4)	**(14.9)**
	0.20	32.99	32.90	33.66	34.48	34.72	34.75	**35.00**
		(8.3)	(8.0)	(10.5)	(13.2)	(14.0)	(14.1)	**(14.9)**
	0.25	32.77	32.71	33.75	34.33	34.45	34.69	34.91
		(7.6)	(7.4)	(10.8)	(12.7)	(13.1)	(13.9)	(14.6)
	0.30	32.81	32.47	33.08	33.84	34.24	34.42	34.45
		(7.7)	(6.6)	(8.6)	(11.1)	(12.4)	(13.0)	(13.1)
Longue	0.10	36.18	36.42	36.92	37.43	37.53	35.74	**37.80**
		(7.3)	(8.0)	(9.5)	(11.0)	(11.3)	(6.0)	**(12.1)**
	0.15	36.75	36.86	37.50	35.98	35.98	35.98	35.98
		(9.0)	(9.3)	(11.2)	(6.7)	(6.7)	(6.7)	(6.7)
	0.20	36.79	37.06	35.57	35.74	35.00	34.39	34.36
		(9.1)	(9.9)	(5.5)	(6.0)	(3.8)	(2.0)	(1.9)
	0.25	36.87	34.90	35.34	34.80	33.75	32.98	32.71
		(9.3)	(3.5)	(4.8)	(3.2)	(0.1)	(-2.2)	(-3.0)
	0.30	36.79	34.70	33.96	33.69	32.88	32.00	31.73
		(9.1)	(2.9)	(0.7)	(-0.1)	(-2.5)	(-5.1)	(-5.9)

TAB. 5.6 – Précisions moyennes avec IM. en variant le nombre de terme d'expansion et le poids des termes d'expansion

est petit. Contrairement à WordNet où l'augmentation du poids détériore la précision moyenne, avec l'information mutuelle la précision moyenne augmente avec le poids. Ceci implique que les termes d'expansion de l'information mutuelle sont plus reliés à la requête originale que les termes d'expansion obtenus avec WordNet. C'est pourquoi il est possible d'améliorer la précision moyenne en utilisant un poids plus important.

Dans les requêtes courtes, les plus grandes augmentations sont de 14.9%, obtenues en utilisant 600 termes d'expansion et un poids de 0.15 ou 0.20. Du côté des requêtes longues, la plus grande augmentation enregistrée est de 12.1%, obtenue en utilisant un poids de 0.10 et un nombre de termes d'expansion égal à 600.

Là encore, les résultats des requêtes courtes sont meilleurs que ceux des requêtes longues. Ceci s'explique par le fait que les requêtes courtes ont plus de potentiel à être améliorées car elles contiennent moins d'information. De même, plus le nombre de termes d'expansion introduits dans les requêtes longues est élevé plus il y a du bruit qui s'ajoute. Pour cette raison, pour les requêtes longues, lorsque le poids est important et que le nombre de termes d'expansion est grand, les résultats sont négatifs. Par contre, les requêtes courtes sont moins affectées.

5.5.2 Nombre de termes d'expansion limité par un seuil sur le poids

Au lieu d'utiliser un nombre fixe pour limiter le nombre de termes d'expansion, nous avons aussi limité le nombre de termes d'expansion en le filtrant par un seuil sur la valeur de l'information mutuelle normalisée calculée pour des termes d'expansion. Par exemple, dans le tableau 5.7, un seuil de 0.90 signifie que seuls les termes d'expansion qui ont une valeur d'information mutuelle normalisée supérieure ou égale à 0.90 sont retenus pour l'expansion de requête.

Les résultats obtenus ressemblent beaucoup aux résultats de la section précédente. Dans les requêtes courtes, on peut observer que la précision augmente avec le poids jusqu'à 0.20. Par contre dans les requêtes longues, cette valeur est plus petite. Elle est de 0.10. Dans les requêtes courtes, les plus grandes augmentations sont de 15.0% et 15.1%, obtenues en utilisant un seuil de 0.01 et des poids respectifs de 0.15 et 0.20. En ce qui concerne les requêtes longues, les plus grandes augmentations sont de 12.1% et 12.3%, obtenues en utilisant un poids de 0.10 et des seuils respectifs de 0.02 et 0.01. L'expansion de requête produit de meilleurs résultats avec la diminution du seuil parce que ainsi un plus grand nombre de termes d'expansion sont ajoutés. Cela rejoint la tendance du test précédent.

Type de requête	Poids	Seuil									
		0.90	0.70	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01
Courte	0.10	30.92	31.10	31.47	32.35	32.35	33.14	33.78	33.93	34.48	34.57
		(1.5)	(2.1)	(3.3)	(6.2)	(6.2)	(8.8)	(10.9)	(11.4)	(13.2)	(13.5)
	0.15	30.55	31.28	31.28	32.56	32.71	33.41	34.02	34.39	34.97	**35.03**
		(0.3)	(2.7)	(2.7)	(6.9)	(7.4)	(9.7)	(11.7)	(12.9)	(14.8)	**(15.0)**
	0.20	30.43	31.25	31.25	32.47	32.74	33.23	34.18	34.42	34.85	**35.06**
		(-0.1)	(2.6)	(2.6)	(6.6)	(7.5)	(9.1)	(12.2)	(13.0)	(14.4)	**(15.1)**
	0.25	31.01	31.13	30.98	32.07	32.56	33.08	33.81	34.24	34.85	34.94
		(1.8)	(2.2)	(1.7)	(5.3)	(6.9)	(8.6)	(11.0)	(12.4)	(14.4)	(14.7)
	0.30	30.80	30.95	30.83	31.95	32.32	32.59	33.35	34.12	34.33	34.45
		(1.1)	(1.6)	(1.2)	(4.9)	(6.1)	(7.0)	(9.5)	(12.0)	(12.7)	(13.1)
Longue	0.10	34.16	34.46	34.93	35.78	36.11	36.52	37.33	37.46	37.80	**37.87**
		(1.3)	(2.2)	(3.6)	(6.1)	(7.1)	(8.3)	(10.7)	(11.1)	(12.1)	**(12.3)**
	0.15	34.02	34.53	35.27	36.25	36.52	36.89	35.74	35.81	35.78	35.74
		(0.9)	(2.4)	(4.6)	(7.5)	(8.3)	(9.4)	(6.0)	(6.2)	(6.1)	(6.0)
	0.20	33.85	34.56	35.41	36.28	36.52	35.17	35.41	34.70	33.99	33.75
		(0.4)	(2.5)	(5.0)	(7.6)	(8.3)	(4.3)	(5.0)	(2.9)	(0.8)	(0.1)
	0.25	33.82	34.50	35.17	35.95	34.56	34.73	33.85	32.74	32.44	32.30
		(0.3)	(2.3)	(4.3)	(6.6)	(2.5)	(3.0)	(0.4)	(-2.9)	(-3.8)	(-4.2)
	0.30	33.82	34.36	34.93	33.72	33.75	33.38	32.51	32.03	31.49	31.39
		(0.3)	(1.9)	(3.6)	(0.0)	(0.1)	(-1.0)	(-3.6)	(-5.0)	(-6.6)	(-6.9)

TAB. 5.7 – Précisions moyennes avec IM. en utilisant un seuil pour la selection des termes d'expansion

5.6 Intégration des termes d'expansion avec l'opérateur logique OU

Dans cette section, nous testons notre méthode d'expansion de requête en combinant les termes d'expansion avec l'opérateur logique OU au lieu de la méthode d'ajout direct. Nous présentons ici, les résultats obtenus avec WordNet et l'information mutuelle.

Nous remarquons que, quand la formule de la théorie de l'ensemble flou est utilisée pour évaluer l'opérateur OU, les résultats sont moins bons. Cela montre que la formule de MAX n'est pas appropriée pour l'expansion de requête. Par conséquent, nous montrons seulement les résultats obtenus avec la formule des probabilités.

Dans WordNet, avec les requêtes courtes ou longues, l'amélioration obtenue en combinant les termes d'expansion avec l'opérateur OU est insignificative par rapport à l'ajout direct (voir tableaux 5.8 et 5.9). Ceci peut s'expliquer par le fait que WordNet fournit de vrais synonymes, et ces derniers se trouvent rarement dans un même document, et donc il n'existe pas de biais.

En utilisant l'information mutuelle (tableaux 5.10 et 5.11), nous remarquons une amélioration plus importante qu'avec WordNet. Et nous pouvons constater que l'amélioration devient plus importante quand le poids des termes d'expansion est plus grand. Dans les tableaux 5.10 et 5.11, nous

Type d'expansion	Poids	Synonyme et hyperonyme	Synonyme et hyponyme
Ajout direct	0.05	31.43 (3.2)	31.86 (4.6)
	0.10	31.07 (2.0)	**31.95 (4.9)**
	0.15	30.31 (-0.5)	30.67 (0.7)
	0.20	29.58 (-2.9)	31.13 (2.2)
	0.25	28.24 (-7.3)	30.76 (1.0)
Avec OU	0.05	31.43 (3.2)	31.86 (4.6)
	0.10	31.10 (2.1)	**32.10 (5.4)**
	0.15	30.46 (0.0)	31.28 (2.7)
	0.20	29.97 (-1.6)	31.37 (3.0)
	0.25	28.88 (-5.2)	31.10 (2.1)

TAB. 5.8 – Précisions moyennes pour les requêtes courtes, évaluées avec OU en utilisant des termes d'expansion qui sont issus de WordNet

Type d'expansion	Poids	Synonyme et hyperonyme	Synonyme et hyponyme
Ajout direct	0.05	33.85 (0.4)	**34.43 (2.1)**
	0.10	33.15 (-1.7)	33.89 (0.5)
	0.15	31.73 (-5.9)	33.15 (-1.7)
	0.20	28.46 (-15.6)	32.34 (-4.1)
	0.25	24.92 (-26.1)	31.36 (-7.0)
Avec OU	0.05	33.85 (0.4)	**34.46 (2.2)**
	0.10	33.25 (-1.4)	34.02 (0.9)
	0.15	31.87 (-5.5)	33.45 (-0.8)
	0.20	30.69 (-9.0)	32.67 (-3.1)
	0.25	25.80 (-23.5)	31.87 (-5.5)

TAB. 5.9 – Précisions moyennes pour les requêtes longues, évaluées avec OU en utilisant des termes d'expansion qui sont issus de WordNet

pouvons observer que, quand le poids est de 0.10, alors il n'existe presque pas de différence entre les résultats de l'ajout direct et de la méthode avec le OU. Par contre, pour un poids de 0.35 les différences deviennent un peu plus importantes.

Il semble que l'intégration avec l'opérateur OU est plus stable. En effet, dans le tableau 5.11, pour un poids 0.10 et un nombre de 400 termes d'expansion, le résultat de l'ajout direct connaît soudainement une baisse, mais le résultat de la méthode avec OU n'est pas affecté. Ceci montre que la méthode avec l'opérateur OU est plus robuste, et moins sensible aux variations de poids des termes d'expansion.

De même, nous avons utilisé l'information mutuelle avec un seuil (voir tableaux 5.12 et 5.13). Les résultats obtenus sont similaires aux résultats de l'expansion avec un nombre termes d'expansion fixe, une légère amélioration est obtenue avec OU.

5.7 Expansion avec pseudo-relevance feedback

Dans l'expérimentation avec le "pseudo relevance feedback", après une première recherche, les 5 premiers documents obtenus sont présumés pertinents. Les termes de ces documents sont triés par ordre décroissants selon leur poids. Les n premiers termes sont utilisés pour étendre la requête.

Les requêtes sont étendues avec l'ajout direct, de 100 à 420 termes d'expansion ont été ajoutés dans les requêtes. Le tableau 5.14 montre les résultats

Type d'expansion	Poids	Nombre de terme d'expansion						
		10	20	50	100	200	400	600
Ajout direct	0.10	32.32	32.53	33.48	33.84	33.93	34.24	34.48
		(6.1)	(6.8)	(9.9)	(11.1)	(11.4)	(12.4)	(13.2)
	0.15	32.84	33.02	33.45	34.05	34.66	34.85	**35.00**
		(7.8)	(8.4)	(9.8)	(11.8)	(13.8)	(14.4)	**(14.9)**
	0.20	32.99	32.90	33.66	34.48	34.72	34.75	**35.00**
		(8.3)	(8.0)	(10.5)	(13.2)	(14.0)	(14.1)	**(14.9)**
	0.25	32.77	32.71	33.75	34.33	34.45	34.69	34.91
		(7.6)	(7.4)	(10.8)	(12.7)	(13.1)	(13.9)	(14.6)
	0.30	32.81	32.47	33.08	33.84	34.24	34.42	34.45
		(7.7)	(6.6)	(8.6)	(11.1)	(12.4)	(13.0)	(13.1)
	0.35	32.41	32.32	32.77	33.48	33.69	33.75	33.81
		(6.4)	(6.1)	(7.6)	(9.9)	(10.6)	(10.8)	(11.0)
Avec OU	0.10	32.65	32.87	33.75	34.05	34.05	34.48	34.85
		(7.2)	(7.9)	(10.8)	(11.8)	(11.8)	(13.2)	(14.4)
	0.15	33.11	33.45	33.93	34.63	35.06	35.36	35.55
		(8.7)	(9.8)	(11.4)	(13.7)	(15.1)	(16.1)	(16.7)
	0.20	33.29	33.51	34.30	35.12	35.30	35.39	35.52
		(9.3)	(10.0)	(12.6)	(15.3)	(15.9)	(16.2)	(16.6)
	0.25	33.35	33.90	34.39	35.06	35.27	35.36	**35.73**
		(9.5)	(10.3)	(12.9)	(15.1)	(15.8)	(16.1)	**(17.3)**
	0.30	33.57	33.35	34.05	34.75	34.94	35.15	35.18
		(10.2)	(9.5)	(11.8)	(14.1)	(14.7)	(15.4)	(15.5)
	0.35	33.14	32.99	33.72	34.33	34.57	34.69	34.91
		(8.8)	(8.3)	(10.7)	(12.7)	(13.5)	(13.9)	(14.6)

TAB. 5.10 – Précisions moyennes pour les requêtes courtes, évaluées avec OU en utilisant des termes d'expansion qui sont issus de l'information mutuelle

Type d'expansion	Poids	Nombre de terme d'expansion						
		10	20	50	100	200	400	600
Ajout direct	0.10	36.18	36.42	36.92	37.43	37.53	35.74	**37.80**
		(7.3)	(8.0)	(9.5)	(11.0)	(11.3)	(6.0)	**(12.1)**
	0.15	36.75	36.86	37.50	35.98	35.98	35.98	35.98
		(9.0)	(9.3)	(11.2)	(6.7)	(6.7)	(6.7)	(6.7)
	0.20	36.79	37.06	35.57	35.74	35.00	34.39	34.36
		(9.1)	(9.9)	(5.5)	(6.0)	(3.8)	(2.0)	(1.9)
	0.25	36.86	34.90	35.34	34.80	33.75	32.98	32.71
		(9.3)	(3.5)	(4.8)	(3.2)	(0.1)	(-2.2)	(-3.0)
	0.30	36.79	34.70	33.96	33.69	32.88	32.00	31.73
		(9.1)	(2.9)	(0.7)	(-0.1)	(-2.5)	(-5.1)	(-5.9)
	0.35	34.73	34.36	33.35	32.74	31.90	31.22	30.95
		(3.0)	(1.9)	(-1.1)	(-2.9)	(-5.4)	(-7.4)	(-8.2)
Avec OU	0.10	36.32	36.52	37.06	37.53	37.70	37.94	**37.97**
		(7.7)	(8.3)	(9.9)	(11.3)	(11.8)	(12.5)	**(12.6)**
	0.15	36.92	37.09	37.53	36.25	36.01	36.11	36.11
		(9.5)	(10.0)	(11.3)	(7.5)	(6.8)	(7.1)	(7.1)
	0.20	37.29	37.36	35.88	36.08	36.01	35.10	34.93
		(10.6)	(10.8)	(6.4)	(7.0)	(6.8)	(4.1)	(3.6)
	0.25	37.23	35.24	35.74	36.15	34.36	33.52	33.38
		(10.4)	(4.5)	(6.0)	(7.2)	(1.9)	(-0.6)	(-1.0)
	0.30	37.13	35.17	34.66	34.60	33.25	32.40	32.17
		(10.1)	(4.3)	(2.8)	(2.6)	(-1.4)	(-3.9)	(-4.6)
	0.35	37.02	35.00	34.09	33.45	32.24	31.56	31.39
		(9.8)	(3.8)	(1.1)	(-0.8)	(-4.4)	(-6.4)	(-6.9)

TAB. 5.11 – Précisions moyennes pour les requêtes longues, évaluées avec OU en utilisant des termes d'expansion qui sont issus de l'information mutuelle

Type d'expansion	Poids	Seuil utilisé									
		0.90	0.70	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01
Direct	0.10	30.92	31.10	31.47	32.35	32.35	33.14	33.78	33.93	34.48	34.57
		(1.5)	(2.1)	(3.3)	(6.2)	(6.2)	(8.8)	(10.9)	(11.4)	(13.2)	(13.5)
	0.15	30.55	31.28	31.28	32.56	32.71	33.41	34.02	34.39	34.97	35.03
		(0.3)	(2.7)	(2.7)	(6.9)	(7.4)	(9.7)	(11.7)	(12.9)	(14.8)	(15.0)
	0.20	30.43	31.25	31.25	32.47	32.74	33.23	34.18	34.42	34.85	**35.06**
		(-0.1)	(2.6)	(2.6)	(6.6)	(7.5)	(9.1)	(12.2)	(13.0)	(14.4)	**(15.1)**
	0.25	31.01	31.13	30.98	32.07	32.56	33.08	33.81	34.24	34.85	34.94
		(1.8)	(2.2)	(1.7)	(5.3)	(6.9)	(8.6)	(11.0)	(12.4)	(14.4)	(14.7)
	0.30	30.80	30.95	30.83	31.95	32.32	32.59	33.35	34.12	34.33	34.45
		(1.1)	(1.6)	(1.2)	(4.9)	(6.1)	(7.0)	(9.5)	(12.0)	(12.7)	(13.1)
	0.35	30.70	30.83	30.83	31.77	31.92	32.29	33.20	33.60	33.69	33.81
		(0.8)	(1.2)	(1.2)	(4.3)	(4.8)	(6.0)	(9.0)	(10.3)	(10.6)	(11.0)
OU	0.10	30.86	31.19	31.71	32.53	32.53	33.60	33.81	34.15	34.63	34.94
		(1.3)	(2.4)	(4.1)	(6.8)	(6.8)	(10.3)	(11.0)	(12.1)	(13.7)	(14.7)
	0.15	30.89	31.50	31.92	33.05	32.99	33.90	34.27	34.88	35.52	**35.67**
		(1.4)	(3.4)	(4.8)	(8.5)	(8.3)	(11.3)	(12.5)	(14.5)	(16.6)	**(17.1)**
	0.20	30.61	31.47	31.80	32.99	33.08	33.84	34.79	35.09	35.42	35.58
		(0.5)	(3.3)	(4.4)	(8.3)	(8.6)	(11.1)	(14.2)	(15.2)	(16.3)	(16.8)
	0.25	30.70	31.37	31.68	32.71	33.05	33.81	34.48	35.00	35.61	**35.70**
		(0.8)	(3.0)	(4.0)	(7.4)	(8.5)	(11.0)	(13.2)	(14.9)	(16.9)	**(17.2)**
	0.30	31.07	31.22	31.40	32.35	32.93	33.54	34.24	34.94	35.15	35.12
		(2.0)	(2.5)	(3.1)	(6.2)	(8.1)	(10.1)	(12.4)	(14.7)	(15.4)	(15.3)
	0.35	30.76	31.10	31.59	32.56	32.56	33.17	33.96	34.54	34.69	34.82
		(1.0)	(2.1)	(3.7)	(6.9)	(6.9)	(8.9)	(11.5)	(13.4)	(13.9)	(14.3)

TAB. 5.12 – Précisions moyennes pour les requêtes courtes, évaluées avec OU en utilisant des termes d'expansion qui sont issus de l'information mutuelle et un seuil pour la selection des termes d'expansion

Type d'expansion	Poids	Seuil utilisé									
		0.90	0.70	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01
Direct	0.10	34.16	34.46	34.93	35.78	36.11	36.52	37.33	37.46	37.80	**37.87**
		(1.3)	(2.2)	(3.6)	(6.1)	(7.1)	(8.3)	(10.7)	(11.1)	(12.1)	**(12.3)**
	0.15	34.02	34.53	35.27	36.25	36.52	36.89	35.74	35.81	35.78	35.74
		(0.9)	(2.4)	(4.6)	(7.5)	(8.3)	(9.4)	(6.0)	(6.2)	(6.1)	(6.0)
	0.20	33.85	34.56	35.41	36.28	36.52	35.17	35.41	34.70	33.99	33.75
		(0.4)	(2.5)	(5.0)	(7.6)	(8.3)	(4.3)	(5.0)	(2.9)	(0.8)	(0.1)
	0.25	33.82	34.50	35.17	35.95	34.56	34.73	33.85	32.74	32.44	32.30
		(0.3)	(2.3)	(4.3)	(6.6)	(2.5)	(3.0)	(0.4)	(-2.9)	(-3.8)	(-4.2)
	0.30	33.82	34.36	34.93	33.72	33.75	33.38	32.51	32.03	31.49	31.39
		(0.3)	(1.9)	(3.6)	(0.0)	(0.1)	(-1.0)	(-3.6)	(-5.0)	(-6.6)	(-6.9)
	0.35	33.69	34.16	32.57	33.42	32.78	32.54	31.76	31.09	30.75	30.48
		(-0.1)	(1.3)	(-3.4)	(-0.9)	(-2.8)	(-3.5)	(-5.8)	(-7.8)	(-8.8)	(-9.6)
OU	0.10	34.09	34.50	35.07	35.84	36.22	36.65	37.46	37.73	37.97	**38.04**
		(1.1)	(2.3)	(4.0)	(6.3)	(7.4)	(8.7)	(11.1)	(11.9)	(12.6)	**(12.8)**
	0.15	34.09	34.73	35.37	36.22	36.45	37.06	35.88	35.95	36.08	35.88
		(1.1)	(3.0)	(4.9)	(7.4)	(8.1)	(9.9)	(6.4)	(6.6)	(7.0)	(6.4)
	0.20	34.02	34.73	35.71	36.52	36.79	35.31	35.74	34.97	34.43	34.29
		(0.9)	(3.0)	(5.9)	(8.3)	(9.1)	(4.7)	(6.0)	(3.7)	(2.1)	(1.7)
	0.25	33.92	34.70	35.47	36.25	35.00	35.20	34.53	33.69	33.08	32.78
		(0.6)	(2.9)	(5.2)	(7.5)	(3.8)	(4.4)	(2.4)	(-0.1)	(-1.9)	(-2.8)
	0.30	33.99	34.63	35.30	36.05	34.77	34.16	33.35	32.64	31.90	31.83
		(0.8)	(2.7)	(4.7)	(6.9)	(3.1)	(1.3)	(-1.1)	(-3.2)	(-5.4)	(-5.6)
	0.35	33.92	34.36	34.97	33.75	33.79	33.48	32.44	31.56	31.12	30.75
		(0.6)	(1.9)	(3.7)	(0.1)	(0.2)	(-0.7)	(-3.8)	(-6.4)	(-7.7)	(-8.8)

TAB. 5.13 – Précisions moyennes pour les requêtes longues, évaluées avec OU en utilisant des termes d'expansion qui sont issus de l'information mutuelle et un seuil pour la selection des termes d'expansion

Type de Requête	poids	Nombre de terme d'expansion 100	220	300	420
Courte	0.70	39.08	39.75	**39.96**	39.78
		(28.3)	(30.5)	**(31.2)**	(30.6)
	0.90	39.29	39.66	39.93	39.69
		(29.0)	30.2)	31.1)	30.3)
Longue	0.70	40.90	40.97	**41.17**	41.14
		(21.3)	(21.5)	**(22.1)**	(22.0)
	0.90	40.73	40.80	**41.17**	41.07
		(20.8)	(21.0)	**(22.1)**	(21.8)

TAB. 5.14 – Précisions moyennes avec "relevance feedback"

obtenus.

Nous constatons que l'expansion avec le "relevance feedback" donne les meilleurs résultats. Comparé à WordNet et à l'information mutuelle, il faut utiliser un poids beaucoup plus important pour les termes d'expansion, ce qui signifie que les termes d'expansion obtenus du "relevance feedback" sont des termes qui sont fortement reliés à la requête initiale.

5.8 Synthèse

Les résultats présentés plus haut indiquent que les meilleurs termes d'expansion sont les termes des documents pertinents qui proviennent du "pseudo

relevance feedback", ensuite viennent les termes de l'information mutuelle et enfin les termes de WordNet.

Les résultats de nos tests suggèrent que notre nouvelle méthode d'expansion est appropriée et peut améliorer la performance pour les thésauri statistiques.

Cependant, notre méthode d'expansion avec OU n'a pas besoin d'une recherche préliminaire comme dans le cas de l'expansion avec "pseudo-relevance feedback". Il est possible de combiner les deux méthodes, nous utilisons d'abord l'expansion avec OU, puis les premiers documents obtenus servent à la méthode de "pseudo-relevance feedback" comme source. Ceci fera l'objet d'une étude future. Les résultats de recherche pourraient s'améliorer.

Chapitre 6

Discussions et conclusion

L'expansion de requête est une méthode efficace pour élargir la couverture de recherche, permettant ainsi de trouver plus de documents pertinents. Dans les études précédentes sur l'expansion de requête dans le modèle vectoriel, les termes d'expansion sont habituellement ajoutés directement dans la requête originale comme étant des termes supplémentaires. Parce que les dimensions sont indépendantes, l'ajout direct peut biaiser le sens la requête initiale : L'aspect qui est renforcé correspond au terme qui est le plus étendu. Le focus de la requête étendue dépend donc de l'ampleur de l'expansion des différents termes. Ce processus peut modifier le focus de la requête originale.

Nous proposons une nouvelle méthode d'expansion qui relie les termes d'expansion au terme original avec l'opérateur logique OU. Notre idée de base veut qu'un terme d'expansion représente une expression alternative du terme original, plutôt qu'une expression additionnelle. Donc il doit être combiné

avec le terme original en utilisant une relation de OU.

Nous avons fait nos évalutions dans la collection de TREC AP en utilisant des requêtes courtes et des requêtes longues. Nous avons utilisé le système de recherche SMART pour effectuer nos expériences. Le système de recherche SMART a été modifié pour pouvoir étendre les requêtes avec WordNet, l'information mutuelle et le "pseudo relevance feedback", et pour intégrer les termes d'expansion dans la requête étendue avec l'opérateur logique OU.

Voici les conclusions que nous pouvons dégager de nos expériences :

- Les meilleurs résultats sont obtenus avec la méthode de "pseudo relevance feedback", où l'expansion des requêtes courtes a produit 31.2% d'amélioration et les requêtes longues 22.1 %. Puis vient l'information mutuelle qui se classe deuxième, une amélioration de 17.3 % est obtenue avec les requêtes courtes et de 12.6 % avec les requêtes longues en utilisant l'opérateur OU. Enfin, WordNet se positionne à la dernière place, sur les requêtes courtes, WordNet a apporté 5.4 % d'amélioration et seulement 2.2 % sur les requêtes longues.
- Les résultats des expériences semblent indiquer que l'intégration des termes d'expansion avec l'opérateur logique OU est plus appropriée que l'ajout direct. Nous avons pu observer une amélioration constante (même si elle est légère) sur la méthode par ajout direct.
- L'expansion de requête permet d'apporter une plus forte amélioration dans le cas des requêtes courtes que celui des requêtes longues.
- Le poids qu'il faut assigner aux termes d'expansion dépend de la res-

source utilisée. Il faut un poids faible (0.05 ou 0.10) pour les termes ambigus comme ceux de WordNet alors qu'il faut un poids plus élevé (0.80 ou 0.90) pour les termes qui produisent moins de bruit comme dans le cas du "pseudo relevance feedback".

6.1 Travaux futurs

Dans nos études futures, nous nous concentrerons sur une autre formule d'évaluation pour la relation OU. Actuellement, une hypothèse d'indépendance est faite entre les termes reliés par OU. Ceci est contradictoire, puisque l'idée de départ est que ces termes sont reliés. Nous testerons d'autres tailles de fenêtre pour le calcul de l'information mutuelle. Nous combinerons l'expansion avec OU dans la méthode de "pseudo-relevance feedback" pour faire la recherche préliminaire.

Bibliographie

[ASS01] BJ Jansen A Spink, D Wolfram and T Saracevic. Searching the web : The public and their queries. *Journal of the American Society for Information Science*, 53(3) :226–234, 2001.

[BM90] David C. Blair and M. E. Maron. Full-text information retrieval : further analysis and clarification. *Information Processing and Management : an International Journal*, 26(3) :437–447, 1990.

[BSAS94] Chris Buckley, Gerard Salton, James Allan, and Amit Singhal. Automatic query expansion using SMART : TREC 3. In *Text REtrieval Conference*, pages 0–, 1994.

[Buc85] C. Buckley. Implementation of the smart information retrieval system. *Technical Report 85-686*, 1985.

[CCW95] W. Bruce Croft, Robert Cook, and Dean Wilder. Providing government information on the internet : Experiences with THOMAS. In *Proceedings of the Second Annual Conference on the Theory and Practice of Digital Libraries (DL '95)*, pages 19–24, Austin, TX, June 1995.

[CH90] Kenneth Ward Church and Patrick Hanks. Word association norms, mutual information an lexicography. *Computational Linguistics*, 16(1) :22–29, 1990.

[Fox80] Edward A. Fox. Lexical relations : Enhancing effectiveness of information retrieval systems. *SIGIR Forum*, 15(3) :5–36, 1980.

[Goo] Google. http ://www.google.ca.

[Har92] Donna Harman. Relevance feedback revisited. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 1–10. ACM Press, 1992.

[JJR71] JR. J. J. Rocchio. Relevance feedback in information retrieval. *Communications of the ACM*, 38(11) :313–323, 1971.

[Mil95] George A. Miller. Wordnet : a lexical database for english. *Communications of the ACM*, 38(11) :39–41, 1995.

[MTT98] Rila Mandala, Takenobu Tokunaga, and Hozumi Tanaka. The use of wordnet in information retrieval. In *Proceedings of the COLING-ACL workshop on Usage of Wordnet in Natural Language Processing*, pages 31–37, 1998.

[MTT99] Rila Mandala, Takenobu Tokunaga, and Hozumi Tanaka. Combining multiple evidence from different types of thesaurus for query expansion. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 191–197. ACM Press, 1999.

[Pro] Lemur Project. The lemur toolkit for language modeling and information retrieval. http ://www-2.cs.cmu.edu/ lemur/.

[QF93] Yonggang Qiu and Hans-Peter Frei. Concept based query expansion. In *Proceedings of the sixteenth annual international ACM SIGIR conference on Research and Development in Information Retrieval*, pages 160–169. ACM Press, 1993.

[SFW83] Gerard Salton, Edward A. Fox, and Harry Wu. Extended boolean information retrieval. *Communications of the ACM*, 26(11) :1022–1036, 1983.

[SM83] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[Voo93] Ellen M. Voorhees. Using wordnet to disambiguate word senses for text retrieval. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 171–180. ACM Press, 1993.

[Voo94] Ellen M. Voorhees. Query expansion using lexical-semantic relations. In *Proceedings of the seventeenth annual international ACM-SIGIR conference on Research and development in information retrieval*, pages 61–69. Springer-Verlag New York, Inc., 1994.

[vR77] C.J. van Rijsbergen. A theoretical basis for the use of co-occurrence data in information retrieval. *Journal of Documentation*, 33(2) :106–119, 1977.

[Wei] Scott Weiss. Glossary for information retrieval. http ://www.cs.jhu.edu/ weiss/glossary.html.

[Zad65] Zadeh. Fuzzy sets. *Information and Control*, 8 :338–353, 1965.