Université de Montréal

Extracting and Exploiting Word Relationships for Information Retrieval

par

Guihong Cao

Département d'informatique et de recherche opérationnelle

Faculté des arts et des sciences

Thèse présentée à la Faculté des études supérieures

en vue de l'obtention du grade de

Doctorat ès sciences (Ph.D.)

en Informatique

Octobre 2008

Université de Montréal

Faculté des études supérieures

Cette thèse intitulée:

Extracting and Exploiting Word Relationships for Information Retrieval

Présentée par:

Guihong Cao

a été évaluée par un jury composé des personnes suivantes:

Philippe Langlais, président-rapporteur

Jian-Yun Nie, directeur de recherche

Pascal Vincent, membre du jury

Bruce W. Croft, examinateur externe

Alain Polguere, représentant du doyen de la FES

Résumé

L'explosion d'information sur le Web rend indispensable les systèmes de recherche d'information ou des engins de recherche pour aider à retrouver des informations pertinentes pour les utilisateurs. Les systèmes de recherche d'information traditionnels utilisent l'hypothèse suivante pour simplifier l'implantation : différents termes dans des documents et des requêtes sont supposés d'être indépendants. Par conséquent, les documents retrouvés doivent contenir exactement les mêmes termes que la requête. Or, les documents contenant des termes différents mais reliés peuvent aussi être pertinents.

Dans cette thèse, nous essayons de relaxer l'hypothèse d'indépendance en exploitant des relations entre termes, comme dans le cas où « algorithme » est relié à « informatique ». Le but de cette étude est de construire de meilleures représentations pour des documents et des requêtes. Dans cadre de modèle de langue statistique, pour estimer les modèles du document et de la requête, au lieu de nous fier uniquement à la distribution de termes, nous intégrons aussi des relations entre termes. Ainsi, les modèles construits ne sont plus basés seulement sur des termes qui apparaissent dans le document et dans la requête, mais aussi sur les termes reliés, qui fournissent des représentations alternatives aux contenus sémantiques du document et de la requête.

Nus avons exploré différentes façons de construire de meilleurs modèles afin de résoudre différents problèmes. En particulier, nous proposons les approches pour traiter les problèmes suivants :

- L'expansion du document, qui vise à étendre la représentation du document afin de couvrir des termes reliés. Ceci permet de relaxer l'approche de « match exact ». Les relations entre termes proviennent soit d'un thésaurus manuel, soit des

statistiques du corpus (basés sur des co-occurrences). Ces relations sont intégrées dans le processus de lissage du modèle de document de telle manière que les termes reliés sont attribués une probabilité plus forte qu'un terme non relié.

- L'expansion de la requête, qui vise à améliorer la représentation de la requête. Nous utilisons un modèle de chaîne de Markov pour ajouter des termes reliés dans le modèle. En utilisant ce mécanisme, nous pouvons, en plus, considérer des termes qui sont indirectement reliés aux termes initiaux de la requête. Cette approche est utilisée pour étendre la requête non seulement pour la recherche monolingue, mais aussi pour la recherche translinguistique.
- A problème crucial dans l'expansion de la requête est le grand nombre de termes d'expansion ajoutés, ce qui peut grandement ralentir le processus de recherche. Ainsi, nous étudions aussi la question de comment réduire ces termes d'expansion à ceux qui sont utiles seulement. Nous proposons une approche d'apprentissage supervisé pour sélectionner des termes d'expansion selon leurs impacts potentiels sur la performance de recherche. Cette approche est aussi utilisée pour proposer des altérations de termes, et ceci constitue une alternative à la troncature de termes, qui sont souvent trop radicale. En comparaison avec les approches traditionnelles, nous pouvons non seulement réduire grandement le nombre de termes d'expansion, mais aussi améliorer la performance de recherche.

Toues les approches proposes ont été testées sur des collections de test de TREC et NTCIR. Les résultats expérimentaux montrent clairement que ces approches peuvent produire des améliorations substantielles comparées aux méthodes traditionnelles.

En conclusion, cette étude a montré que les modèles traditionnels peuvent être améliorés en considérant les relations entre termes, et ceci permet d'augmenter la performance de recherche.

Mots clés : Recherche d'information, Modèle de langue, Relation entre termes, Expansion de document, Expansion de requête

Abstract

With the exponential growth of information in the Web, information retrieval systems have become more and more important as an indispensable tool to locate the information that interests the users. The traditional information retrieval systems adopt the independence assumption in order to simplify the model construction. The independence lies in three aspects, i.e., among query terms, document terms, or between a query term and a document term. The independence assumption does not hold in practice, which results in the ambiguities in query and documents representation, as well as the "exact match" for relevant document retrieval.

In this thesis, we try to release the independence assumption by exploiting the relationships between words. Since we adopt the language modeling framework for document ranking, we have to estimate a probabilistic model with multinomial distribution for document and query respectively. Therefore, our basic approach is to improve the estimation of the two models by making use of the word relationships. In the thesis, we tried the following approaches:

- Document expansion, which aims to avoid the "exact-match" for relevant documents. We consider the word relationships when smoothing the document model so some related terms will be assign higher probabilities even they do not occur in the document.
- Query expansion to resolve the ambiguity in query representation. Particularly, we use a Markov chain model to exploit non-immediate word relationships. This framework is also extended to deal with cross-lingual information retrieval problems.

- We also proposed a supervised learning framework to select good query expansion terms and query alterations. The selection is according to the relationship between the selected term and other query terms. More particularly, the selection considers the impact of individual expansion terms.

All the proposed methods are evaluated with TREC or NTCIR benchmarks, and the experimental results show the methods achieve substantial improvements over some competitive baselines.

Keywords: Information Retrieval, Language Modeling, Word Relationship, Document Expansion, Query Expansion

Acknowledgement

First and foremost, I would like to express my greatest gratitude to my advisor, Professor Jian-Yun Nie. I could not complete this thesis without Jian-Yun's insightful guidance. Jian-Yun is a great advisor. Whenever I have a problem with my research projects, he is always ready to give me some useful advices. From him, I learned many things. I learned how to do serious research, how to combine theoretic ideas with practical applications, and how to write quality scientific papers.

I also have a lot of reasons to thank Dr. Jianfeng Gao at Microsoft Research, who introduced me to the fields of natural language processing and information retrieval. I began to work with Jianfeng when I was a Master degree student. He introduced me to the most important and fundamental knowledge to those fields and the basic skills to perform research work.

I would like to acknowledge the role played by Professor Stephen Robertson. I was deeply impressed by his insight to the research problems. He taught me how to analyze experimental results with extensive statistical approaches. Some of the work presented in this thesis benefited greatly from discussions with him.

I am thankful to Professor Bruce Croft, Philippe Langlais, Pascal Vincent and Alain Polguere for accepting to be members of my PhD committee. I am honored to have them in my committee.

I would also like to thank Prof. Guy Lapalme and Elliott Macklovitch of the RALI lab for their kindness and supports. In addition, I would also extend my gratitude to the fellow students, Jing Bai, Lixin Shi, Alex Patry, Fabrizio Gotti, Hugo Larochelle,

Youssef Kadri, Hughes Bouchard. Thanks for all their important support during the last four years. In particular, Fabrizio has been a helpful expert on Linux system. Whenever I had technique problems, he was always ready to suggest me a good solution.

I would like to thank my parents. Without their unconditional support, it was impossible for me to obtain any achievement.

Finally, I would like to thank my dear wife, Yao, for her continuous support and encouragement. Whenever I felt frustrated, she was always there to listen to all my bitter words, to comfort and to encourage me. She brought so much happiness to me during my PhD studies.

Table of Contents

List of Tables

List of Figures

List of Symbols

d: A document – a sequence of unordered words

q: A query – a sequence of unordered words

θ_d or $P(w|d)$: A probabilistic model for a document

$P_{ml}(w|d)$: the probabilistic model for a document estimated with Maximum Likelihood estimation

θ_q or $P(w|q)$: A probabilistic model for a query

$P_{ml}(w|q)$: the probabilistic model for a query estimated with Maximum Likelihood estimation

θ_F: The topic model of feedback documents

Chapter 1

Introduction

Recent years we have seen an explosive growth of the volume of information, especially on the web. The volume of information in the web can be measured by either the number of web sites or indexed web pages. According to a widely respected statistics issued by Netcraft[1], we can observe the number of web sites is increasing exponentially with time.

On the other hand, a recent study which estimated the number of web pages indexed by two popular search engines (Google and Yahoo) reveals that there are at least 45 billion web pages in the publicly indexable web[2]. Considering the number of the public accessible web pages are much less than the invisible ones, which are called deep web [Bergman, 2001], the whole Web should be much larger.

Other types of textual information, such as books, newspapers, and periodicals are also growing rapidly. According to a study by Lyman and Varian (2000), the worldwide production of original content, stored digitally using standard compression methods, is at least 1 terabytes/year of books, 2 terabytes/year for newspapers, 1 terabytes/year for periodicals, and 19 terabytes/year for office documents.

---

[1] http://news.netcraft.com/archives/web_server_survey.html

[2] http://www.worldwidewebsize.com/

The large volume of information makes it a significant challenge to effectively and efficiently manage the online information. Information retrieval is by far the most useful technique to address the problem. The retrieval of textual information is especially important, because the most frequently needed information is contained in texts; even though other multi-media information also exists. In this thesis, we focus on text retrieval. Therefore, hereafter, we mean text retrieval even if we call it information retrieval.

1.1 Information Retrieval Challenges

The task of information retrieval (IR) can be defined as to locating relevant documents from a large set of documents with respect to a user's information need, which is usually described by a query. Therefore, a retrieval process involves three issues: query formulation, document representation and relevance judgment. However, the IR task is poorly defined. In particular, the notion of relevance has many interpretations, and it is difficult to capture what relevance is in the user's mind. The difficulties are spread over all the above 3 aspects. We will review them in the following subsections.

1.1.1 Difficulties to Represent Information Need

The user's information need is usually expressed by a short natural language sentence, some Boolean expressions or even just some keywords. A query formulated in this way cannot exactly convey the user's information need. There are at least three reasons.

Firstly, a typical query typed in a search engine is extremely short. Most users would not be patient enough to type a long query. In January 10, 2007, the leading web ranking provider, RankStat.com[3], reported that most people use 2-word phrases in search engines. Of all search engine queries, 28.38 percent use 2 word phrases, 27.15 percent use 3 word

[3] www.rankstat.com

phrases. The detailed information is shown in Table 1. From the table, we observe that the distribution of the query length is approximately an exponential distribution with a long tail, and the average query length is less than 3 words. With such an extremely short query, it is hard for the user to express her information need precisely and completely.

Table 1. Query Length of World Wide Search Engines

Query Length	Percentage	Query Length	Percentage
1 word	13.48	6 words	3.67
2 words	28.38	7 words	1.63
3 words	27.15	8 words	0.73
4 words	16.42	9 words	0.34
5 words	8.03	10 words	0.16

Secondly, the keywords in a query may possess more than one sense, which makes a query very ambiguous. For example, the query term "java" has at least three senses. It may mean a popular programming language, a name of an island in the east-south Asia, as well as the name of one kind of coffee. It is very difficult for a human to determine the exact intent of the user without further information, not to say a machine.

Thirdly, the words used in a query are not the only way to express information need. For example, a user interested to buy a music player might search the item in the Internet with a query "ipod". However, relevant documents may use "apple music player" to represent this item. Here, both terms represent the same concept, which represent a digital music player. The documents using "apple music player" should be considered to be relevant to the query because it describes the same concept. However, many concepts can be expressed in multiple ways. Therefore, there may be a gap between terms used in the documents and those used in the queries. An approach relying on direct word-matching between the document and query will fail to retrieve many relevant documents.

1.1.2 Difficulties to Represent Documents

Besides the information need, it is also very difficult to represent a document efficiently and effectively. "Effective" means that the content of each document in the collection should be represented thoroughly. "Efficient" means that the internal

representation should make document retrieval very fast. The efficiency problem is usually solved by organizing the document index as a special data structure, i.e., inverted file (Salton and Buckley, 1988; Manning et al., 2008). This data structure supports efficient looking up the occurrence of a term within a document. For the effectiveness problem, there is no satisfactory solution yet. Most state-of-the-art experimental IR systems represent documents with the keywords occurring in them. The keywords can be weighted according to their occurrences within individual document and the whole collection, such as TFIDF (Salton and Buckley, 1988), according to the unigram language modeling (Zhai and Lafferty, 2001a), as well as according to the BM25 schema in Okapi system (Robertson and Walker et al., 1992). All of the above models assume keywords within a document to be independent, which lead to the so called "bag-of-word" approach. However, in natural languages, this strong assumption does not hold. The assumption is in fact a matter of mathematical convenience than a reality. For example, the word "program" is not independent from "algorithm". A document containing "program" may talk about algorithms. There is thus a relationship between the two terms. A crucial problem is how to take into account such relationships during document retrieval.

1.1.3 Difficulties to Judge Relevance

The documents that can satisfy a user's information need are called relevant documents, and thus, the retrieval task is to find all such relevant documents. The notion of "relevance" is very complex. Like a user's information need, relevance is generally imprecise and depends on the situation or context of the retrieval task. The criterion for judging whether a particular set of documents would satisfy a user's information need is inherently impossible to formalize. In order to simulate the notion of relevance in a system, two assumptions are often made to simplify the retrieval task. First, the relevance of one document is assumed to be independent of other documents, including those already retrieved (independent relevance assumption). This means that a document is relevant whatever the documents the user has already seen before. Second, the relevance

of a document is assumed to mean the level of topical relevance of the document with respect to the query (topical relevance assumption) (Zhai, 2003), i.e. a document is judged relevant if it is on the same topic as the query.

In fact, the two assumptions do not hold for real relevance. A document may be relevant or not depending on what other documents the user has already read. A document on the same topic as the query may still be irrelevant (Zhang et al., 2002; Zhai et al., 2003). However these assumptions are made in order to make the estimation of relevance tractable. Based on the two assumptions, most traditional models judge relevance according to the occurrence of query terms in the corresponding document. These models are indeed based on term matching, and they are commonly called "bag-of-words" models. Undoubtedly, a "bag-of-words" model, in many cases, works very well. In the example mentioned above, some users are interested in laptops can retrieve some relevant documents by typing the query "laptop" supposed these relevant documents contain this term. However, in many other cases, the term-matching strategy is insufficient. For example, the user is not interested in arbitrary laptops, but the Apple ibook. In this case, she formulates a query "ibook". With this query, only documents containing "ibook" can be retrieved by the "bag-of-word" models. However, a document containing "apple laptop" can also satisfy the user's need. In another example, a traveler to Indonesia is interested in the transportation information in Java Island. Then a possible query might be "find transportation in Java Island". We typed this query in two popular commercial search engines, Google and Live search. However, in the result lists returned by both search engines, most web pages are about the Java programming language. Among the top 10 results, only 2 or 3 results are about transportation systems. This error is caused by the multiple senses of the term "java". The overwhelming number of answers about "Java language" may be partly due to the fact that there are much more document about "Java language" than about "transportation in Java Island" on the Web. Nevertheless, for this particular user, most of the search results are irrelevant.

In the above sections, we mentioned three challenges of Information Retrieval. The three challenges are not independent from each other. A common factor behind the challenges is the unrealistic independence assumption made between terms. The

independence assumption involves three aspects: between query terms, between document terms as well as between a query term and a document term. The first two aspects cause the ambiguity in the representation of query and document respectively, while the third one leads to the exact term-matching strategy. The independence assumption was proposed from the very beginning of information retrieval. Robertson et al. (1982) did an extensive study on the nature of this assumption and proposed a unified model retrieval model based on the independence assumption. In fact, we can overcome the challenges to some degree if we relax the independence assumption. In the two examples mentioned earlier, if we take the relation between "*ibook*" and "Apple laptop" into account, we are able to identify that the documents containing "Apple laptop" but not "ibook" are also relevant. On the other hand, if we consider the relation between "java" and "island" in the second example, we should infer that the term "java" does not mean a programming language, but a name of an island in Indonesia. In the following section, we will briefly describe the principle of our approach to address these problems by exploiting word relationships.

1.2 Our Approach – Exploiting Word Relationships

In the thesis, we focus on the first and third problems mentioned in above section, i.e., the **ambiguity** in the query formulation and the **mismatching** problem between query and document terms.

- **Resolving Query Ambiguity with Query Expansion**

  We argue that the ambiguity of a query is tightly related to the incompleteness of the query, and when a query is enhanced with more related terms, its ambiguity is alleviated. We again use the earlier example "find transportation in Java Island" for illustration. As mentioned above, this query has ambiguity because the term "java" has more than one sense. With this query, most of the retrieved documents returned by the existing search engines are about Java programming languages. However, if we can identify the related

term such as "bus" and add them into the original query, both Live Search and Google are able to return more relevant documents. By expanding the query, the documents with query terms and related terms will be ranked higher, and usually these documents are more relevant. Therefore, query expansion is a good approach to reduce the query ambiguity. However, every coin has two sides. Beside the benefit of query expansion, it also brings some risks: noise and topic drift. If an expansion term is not tightly related to the query, it brings nothing except noise, i.e. unrelated documents. On the other hand, even the expansion terms are all related to the query, if we add too many expansion terms, the resulted query may drift from the previous query. For example, if we re-add other related terms such as "metro", "airport", "tourism", "ferry" etc beside "bus" into the original query, we will find the search engines return many documents about transportation all over the world instead of in the Java Island. Certainly, they are not relevant to the original query. This is because we consider each query term (including the original terms and expansion terms) equally. We emphasize too much about transportation by adding many terms related to it. To avoid this problem, we should assign suitable weight to each expansion term. We will talk about the details of query expansion in chapter 4.

- **Resolving Mismatch with Implicit Document Expansion**

There are two methods to overcome the mismatching between query terms and documents: query expansion and document expansion. In the previous section, we introduced query expansion. Analogously, document expansion is a technique to enhance the document representation with some related terms. This can be done implicitly or explicitly. In the explicit approach, the related terms are identified and added to the document representation, which is the same with query expansion. The implicit approach is quite different from the explicit one, in which no item is added to the document. As described in section 2.2, the term-matching strategy disregards the query terms which do not occur in the document. In our approach, we consider the closeness of the query term and its related terms in the document, so that a query term which does not occur in the

document can also contribute to document ranking if it has some related terms in the document. For example, given a query "recent natural disaster", a document about Chinese earthquake in May 12, 2008 is definitely relevant. Nevertheless, it cannot be retrieved, if it does not have any query term, according to the term-matching strategy. However, if we can identify that earthquake is also a kind of natural disaster, i.e., it is a hyponym of "natural disaster", and take this relationship into account when calculating the document score, this document will possibly be returned. In our work, we mainly used the implicit document expansion, which will be detailed in chapter 3.

The central idea in this thesis is to exploiting word relationship for query and document expansion. This idea is not new. However, we will try to make use of word relationship in different ways. In particularly:

- **We will try to combine multiple word relationships**

  We will build a flexible framework which is feasible to integrate arbitrary word relationships. Usually, the knowledge of related words comes from different sources, e.g. from a large corpus according to some statistical metrics, which can be co-occurrence, proximity and so on; or from some manually-created thesauri. All the resources are useful for IR. So, an important problem is to know how to combine them. This is the first problem we will address.

- **We extend immediate word relationships to indirect word relationships**

  Word relationship is transitive, and it can be transited from one word to another word. For example, "C++" is related to "programming", and "programming" is related to "computer", then we can infer that "C++" is also related to "computer" but may have different closeness. The indirect relationships will allow us to find more related terms. In our work, we consider not only immediate word relationship, but also indirect relationship. We use

Markov Chain (Ross, 2003) to model the relationship transition. The occurrence of a word is reinforced according to the presence of its directly or indirectly related words.

- **We will use statistical language modeling (LM) as our basic integration framework.**

  This choice is motivated by the solid theoretical foundation of the framework, its ability to deal with incomplete and noisy data, as well as its flexibility to be extended to integrate more criteria. We will show that LM framework can be extended to take into account arbitrary word relationships, and the importance of the relationship can be adjusted automatically in several ways.

Our approaches will be tested on several TREC collections. The experiments aim to validate the following hypotheses:

- Exploiting word relationship can construct better query/document representations so as to improve the retrieval effectiveness
- Different word relationships have different impacts to retrieval performance. We propose some learning methods to assign appropriate weights to the relationships, so that we can take advantage of each type of word relationship.

1.3 Our Contributions

Our work aims to make the contributions as follows:

- **Combining Statistical Word Relationship and Manually Created Relationship**

  Both of the two sources have some advantages and disadvantages. The statistically-derived knowledge has high coverage but low precision, and the

method is portable to any language; while the manually-created thesauri have low coverage but high precision. Therefore, the two sources are complementary. A combination of them is expected to benefit from each other.

- **Proposing a Framework to Incorporate Indirect Word Relationship**

As mentioned above, the word relation is transitive. An elegant mathematic tool to model the relation transition is Markov Chain (Ross, 2003). A Markov Chain is usually represented as a directed graph with a set of vertices and directed weighted edges. The vertices are the states of the Markov chain. Two states are transitable if and only if there is an edge between them. The weight associated with edge represents the transition probability. With this model, we view each word in the vocabulary as one state, and measure the word relatedness as transition probabilities. The transition probability between two words is non-zero if and only if they are directly related. Given a set of initial words, we can find the related terms by a random walk process. Therefore, the indirectly related words can be found by multi-step random walk.

- **Proposing a Framework to Incorporate Query Translation and Query Expansion for Cross-lingual Information Retrieval**

Cross-lingual information retrieval (CLIR) is an important task of IR, in which queries and documents are written in different languages. The key issue is query translation. Traditional method to deal with CLIR separates query translation and document retrieval into two phases: in the first step, the query is translated into the document language; then we perform a monolingual document retrieval. We propose a unified framework to incorporate the two phases. We view translation between query terms and documents as one kind of word relationship. This relationship is used for query translation. Other relations between monolingual terms, such as co-occurrence, are used for query expansion. A Markov Chain model is built for relation propagation.

- **Supervised Learning Method is used to adjust the Relative Importance of Individual Relationships**

  We measure the relative importance of individual relationships with weights. In this thesis, we proposed some supervised learning methods to adjust the weights by maximizing the retrieval effectiveness of training data directly. Our approach is quite different from other previous studies, which estimated the weights by maximizing some indirect metrics, such as the likelihood of training queries; while our learning methods try to maximize directly the objective function which is the retrieval effectiveness.

1.4 Organization of the Thesis

The remaining of the thesis is organized as follows.

In Chapter 2, we will introduce some basic IR models as well as the procedure of IR, such as document indexing, query processing, document retrieval and result evaluation.

In Chapter 3, we will address the problem of document expansion, in which the word relations are used to resolve the mismatching between query terms and document terms.

In Chapter 4, we extend the immediate word relation into indirect relations. We will describe how to use the Markov Chain model to handle indirect relations and how the relations are used to do query expansion.

In Chapter 5, we extend the monolingual word relation into bilingual setting, and the relations are used to do cross-lingual query expansion.

Chapter 6 and 7 mainly focuses on how to tackle word relations with supervised learning methods. The two chapters deal with pseudo-relevance feedback and query term stemming respectively.

Finally, general discussions about the thesis and some conclusions will be given in Chapter 8.

Chapter 2

Traditional Information Retrieval Approaches

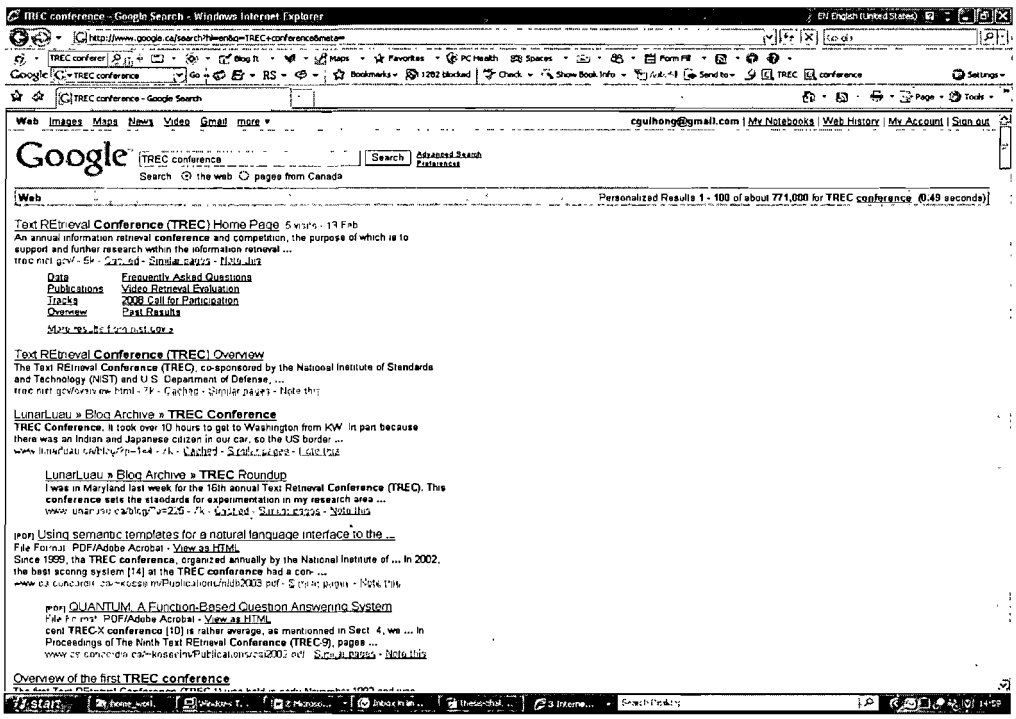

Figure 1. The user interface of Google Search Engine

Before the World Wide Web emerged, information storage and retrieval systems were almost exclusively used by professional indexers and searchers (Hiemstra, 2000). Typically, professional searchers act as "search intermediaries" for end users. They try to figure out in an interactive dialogue with the system and the user what the user needs, and how this information need should be used in a successful search. With the occurrence of the World Wide Web, especially its explosive growth since the late of 1990s, there is

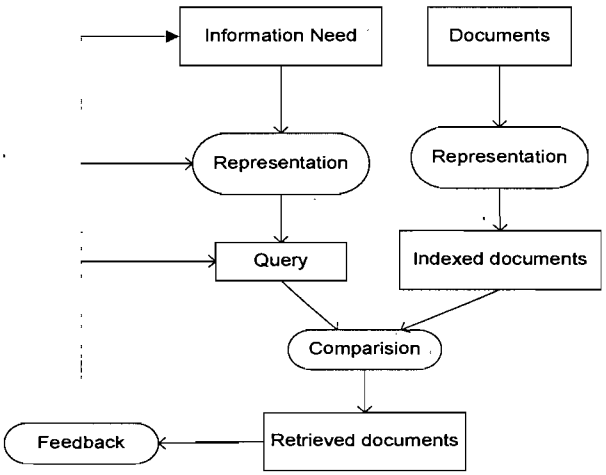

Figure 2. Information Retrieval Processes

very large volume of information on the Internet. People seek information on the Internet every day. As a consequence, no human expert can replace the searcher to play the role of intermediary. Modern information retrieval (IR) systems are used to answer this requirement. One of them is the on-line search engine. Figure 1 shows a typical interface of a search engine.

Despite the large variety of systems, user interface and performance, some basic techniques are commonly used. The following sections introduce briefly the discipline of information retrieval and some technical terms used throughout the thesis.

2.1 Definition and Basic Processes of IR

The discipline of IR is almost as old as the computer science itself. An early definition of information retrieval is the following by Mooers (1950).

> *"Information retrieval is the name of process or method whereby a prospective user of information is able to convert his need for information into an actual list of citations to documents in storage containing information useful to him."*

An IR system is a software program that stores and manages information on documents. The system helps the user to locate information needed. Different from

question answering systems [Voorhees, 2000], the system does not explicitly return information or answer to questions. Instead, it informs the existence and location of documents that might contain the needed information. Some of the documents may satisfy the user's need, and then they are called ***relevant*** documents.

A typical IR system supports three basic processes: the representation of the content of the documents, the representation of the queries, as well as the comparison between the two representations. The processes were organized as figure 2 by Croft (1993). In the figure, squared boxes represent data and rounded boxes represent processes.

The document representation is called ***indexing*** process. The process takes place off-line, so it is transparent to the users. The indexing process results in a formal representation of the document: the internal document representation. Often, the representation is stored in the storage devices (such as hard disks or tapes) as inverted files (Baeza-Yates and Ribeiro-Neto, 1999; Manning et al., 2008), in order to support efficient comparison process (or query evaluation). The indexing process usually includes the following procedures: finding the indexing units, filtering common and meaningless words (***stop words***) and performing morphological analysis. We take the indexing of an English document for instance. The indexing process is the following steps:

- **Tokenization**

  This process identifies the indexing units in a character stream. In IR with English documents, the indexing unit is usually set to be words. Normally, it recognizes punctuations and white space as separators.

- **Stopword Removal**

  When selecting words to represent an English document, we prefer to the words making the document different from others, i.e., the discriminative words. However, as we know, some words occur in text very frequently and thus lose the discriminative ability. These words include most function words, such as "in", "a" , some verbs and so on. We usually remove these words by looking them up in a

predefined word list (i.e., stop list). This process has at least two advantages: 1) reducing the size of indexing files; 2) improving the efficiency of query evaluation.

- **Stemming**

The process is done to get the root form of a word. For example, "computer", "computing" and "computed" are transformed into a single root form "comput". The stemming is done to increase the chance to match terms.

The process of representing the information need is often referred to as the ***query formulation*** process. The resulting formal representation is the query. This process is usually performed by the user independently. However, in some cases where interaction between the user and the system are allowed, the user can reformulate the query based on the feedback of the system. This reformulation is called ***relevance feedback***. If the interaction is not allowed, the system cannot judge which document is relevant explicitly. However, it can also reformulate the query based on its previous rank list by assuming the top retrieved documents to be relevant. This process is called ***pseudo-relevance feedback***, which is used widely in query expansion, and has shown to be effective across retrieval models (Zhai and Lafferty, 2001b; Lavrenko and Croft, 2001).

The comparison of the query against the document representations is also called the ***term matching*** process, ***query evaluation*** or ***document retrieval***. The retrieval process results in a rank list of relevant documents. Users will walk down the list of documents in search of the information they need. Document retrieval will hopefully put the relevant documents somewhere in the top of the ranked list, reducing the time the user has to spend to find the relevant information.

2.2 Some Existing IR Models

In order to be able to identify relevant documents, an IR system must assume some specific measure of relevance between a document and a query. This means that we have to have an operational definition of a relevant document with respect to a query. Thus, a

fundamental problem in IR is to formalize the concept of relevance. A different formalization of relevance generally leads to a different IR model. Over the decades, various retrieval models have been proposed, studied and tested. Their mathematical basis spans a large spectrum, including linear algebra, logic, probability and statistics. In this section, we will give a brief introduction to some existing IR models, particularly the newly introduced statistical language model since all of our investigations are made within this framework.

2.2.1 Boolean Models

The Boolean model (Baeza-Yates and Ribeiro-Neto, 1999) is a retrieval model based on set theory and Boolean algebra. In the simplest case, i.e., the classical Boolean model, each document is represented as a logic conjunction of a set of Boolean variables. Each Boolean variable corresponds to a term in the vocabulary of whole document collection *C*. The Boolean value, true or false, represents the term existing or non-existing in the document. Since the conjunction with false does not change the value of the Boolean expression, we only consider the document terms. Therefore, the document can be represented as: $d = d_1 \vee d_2 \vee d_3 \ldots$. Here d_i denotes a document term.

On the other hand, a query in the Boolean model is represented as a Boolean expression such as $q = (q_1 \wedge q_2) \vee q_3$. A document is considered as relevant if and only if we have $d \rightarrow q$.

Though the classical Boolean model is intuitive and efficient to be implemented, it has several problems:

- The term weighting is binary, i.e., true or false, which seems too rough. It can only model the existence or absence of a term within a document, but cannot model the importance of existing terms. Normally, a term occurring in the document frequently is considered to be more important than a less frequent term.

- The Boolean model predicts each document as relevant or non-relevant. There is no notion for partial match to the query conditions. For instance, let d be a document for which $d=(0,1,0)$. The three items in the vector denote the absence of term a, c and existence of term b. A query is formulated as $[q = a \wedge (b \vee c)]$, then d is non-relevant to q.

The classical Boolean model has been extended on the above expects. For example, term weighting has been integrated by using fuzzy logic (Kraft et al., 1983; Radecki, 1979) or p-norm (Salton et al., 1983), and documents can be ranked as more than two scales.

2.2.2 Vector Space Model

The vector space model (VSM) (Salton et al., 1975; Salton and McGill, 1983; Salton, 1989) recognizes binary weighting is too limited, and it proposes a framework in which partial matching is possible. VSM is a similarity-based model (Zhai, 2003), which assumes that the relevance status of a document with respect to a query is correlated with similarity between the query and the document at some level of representation; the more similar to the query a document is, the more relevant the document is supposed to be. In the vector space model, each document or query is represented as a vector in a high-dimensional term space. Each term is assigned a weight that reflects its "importance" to the document or the query. Given a query, the relevance status value of a document is given by the similarity between the query vector and document vector as measured by some vector similarity measures, such as the cosine of the angle formed by the two vectors.

Formally, a document d may be represented by a document vector $d = (d_1, d_2, ..., d_n)$, where n is the total number of terms and d_i is the weight assigned to term i. Similarly, a query q can be represented by a query vector $q = (q_1, q_2, ..., q_n)$. The weight of both document and query terms is used to measure its importance. There are many term weighting schemes (Manning et al. 2008). Among them, the most common used one is

the tf-idf weighting. Here, tf denotes the term frequency within the document, and idf is the inverse document frequency, which is usually calculated as:

$$idf(t_i) = \log \frac{N}{n(t_i) + 0.5} \tag{2.1}$$

where t_i is a term in the vocabulary, N is the number of documents in the whole collection and $n(t_i)$ is the number of documents with t_i, i.e., the document frequency of t_i. idf measures how common the term is. A common term has low idf, or verse visa. Then the term t_i is weighted as:

$$Wt(t_i, d) = tf(t_i, d) \times idf(t_i) \tag{2.2}$$

There are also some variants of the above tf-idf weighting schema. Some of them also consider the document length (Singhal, 2001).

In most cases, the query terms are usually weighted in the same way with the document terms. However, since the query is usually very short, the term frequency of each term is either 0 or 1, therefore some researchers argued to use a different weighting schema. Salton and Buckley (1988) suggested the following formula:

$$Wt(t_i, q) = \left(0.5 + \frac{0.5 tf(t_i, q)}{\max_j tf(t_j, q)} \right) \times idf(t_i) \tag{2.3}$$

With the cosine measure, we have the following similarity function of the document and query:

$$sim(d, q) = \frac{d \bullet q}{|d| \times |q|} \tag{2.4}$$

where $|d|$ is the length of the vector which is defined as:

$$|d| = \sqrt{\sum_{t_i \in V} (Wt(t_i, d))^2}$$

and $|q|$ is defined analogously.

The vector space model usually decomposes a retrieval model into three components: i) a term vector representation of query; ii) a term vector representation of document; iii) a similarity/distance measure of the document vector and the query vector. In fact, vector space model is a general framework, in which the document and query representation and similarity measure can be arbitrary defined (Zhai, 2003).

2.2.3 Probabilistic Models

In probabilistic model, we are often interested in the question "what is the probability that this document is relevant to this query?" (Sparck Jones et al., 2000). Given a query, a document is assumed to be either relevant or non-relevant, but a system cannot be sure about the relevance status of a document. So it has to rely on a probabilistic relevance model to estimate it.

Formally, let d and q denote a document and query respectively. Let R be a binary random variable that indicates whether d is relevant to q or not. It takes two values which we denote as r ("relevant") and $\bar{r}$ ("irrelevant"). The task is to estimate the probability of relevance, i.e., $P(R=r|d,q)$. Depending on how this probability is estimated, there are several special cases of this general probabilistic relevance model.

First, $P(R=r|d,q)$ can be estimated directly using a discriminative (regression) model. This model assumes that the relevance probability depends on some "features" that characterize the matching of d and q. Such a model was first introduced by Fox (1983), where features are term frequency, authorship, and co-citation. They were combined using linear regression.

Table 1. Contingency Table of Term Occurrence

	Relevant	Irrelevant	Total
#Doc containing the term *t*	*#r*	*n-#r*	*n*
#Doc not containing term *t*	*#R-#r*	*N-n – (#R-#r)*	*N-n*
# Total Doc	*#R*	*N-#R*	*N*

The Binary Independence Retrieval (BIR) model (Robertson and Spark Jones, 1976) is perhaps the most well known probabilistic model. The BIR model assumes that terms are independently generated by a relevance and an irrelevance model, so is essentially a use of Naïve Bayesian classifier for document ranking. The documents are sorted descendingly according to the log-odds between $P(r \mid d,q)$ and $P(\bar{r} \mid d,q)$. i.e.,

$$Score(d,q) = \log \frac{P(r \mid d,q)}{P(\bar{r} \mid d,q)} \propto \sum_{t \in d} \log \frac{p(t \mid q,r)(1 - p(t \mid q,\bar{r}))}{(1 - p(t \mid q,r))p(t \mid q,\bar{r})} \qquad (2.5)$$

where $p(t \mid q,r)$ is the generation probability of term t by the relevance model while $p(t \mid q,\bar{r})$ is the generation probability by irrelevance model. Therefore, we can view $\log \frac{p(t \mid q,r)(1-p(t \mid q,\bar{r}))}{(1-p(t \mid q,r))p(t \mid q,\bar{r})}$ as the weight of t. It is estimated based on the occurrence of t in the document d and the whole collection. For a more refined interpretation of the model, we start with the term incidence contingency Table 2.

With the information given in the table, we estimate the probabilities as:

$$p(t \mid q,r) = \frac{\#r}{\#R}$$

and $$p(t \mid q,\bar{r}) = \frac{\#R - \#r}{N - \#R}$$

Substitute the two equations into equation 2.5, we get:

$$Score(d,q) = \sum_{t \in D} \log \frac{p(t \mid q,r)(1-p(t \mid q,\bar{r}))}{(1-p(t \mid q,r))p(t \mid q,\bar{r})}$$
$$= \sum_{t \in d} \log \frac{\#r(N-n-\#R+\#r)}{(\#R-\#r)(n-\#r)} \quad (2.6)$$

2.2.4 Statistical Language Models

A statistical language model (SLM) provides a mechanisam to calculate the generation probability of a string. Based on the tokenization of the string, it could be word-based model or character-based model. SLM has been studied extensively and had a great success in the community of speech recognition and natural language processing (Jelinek, 1998; Brown et al., 1993; Gao et al., 2002). In speech recognition, the system calculates the probabilities of all utterances that can occur based on an estimated language model and select the one with the largest probability (Jelinek, 1998). Language models also play a central role in statistical machine translation (*SMT*) (Brown et al., 1993). Usually the *SMT* model is decomposed into two components: the translation model mapping the sentences in target language to the one in source language and the language model for target language sentences. Therefore, the language model corresponds to selecting a high quality, grammatical translation sentences.

2.2.4.1 Document Ranking

The language model (LM) approaches for IR was first introduced by Ponte and Croft in (Ponte and Croft, 1998) and later explored by (Hiemstra and Kraaij, 1998; Miller et al., 1999; Berger and Lafferty, 1999; Song and Croft, 1999; Zhai and Lafferty, 2001a; Bai et al., 2005). There are two ways to formulate the relevance status in LMs: one considers the likelihood of a query as being generated by a probabilistic model based on a document.

- **Query Likelihood**

We denote a query $q = q_1 q_2 ... q_n$ and a document $d = d_1 d_2 ... d_m$, this probability is denoted by the conditional probability *p(q|d)*. However, in order to rank the document, we are interested in the posterior probability *p(d|q)*, which can be calculated by Bayes rule in the following way:

$$p(d \mid q) = \frac{p(q \mid d)p(d)}{p(q)} \propto p(q \mid d)p(d) \tag{2.7}$$

where *p(d)* is the prior probability that *d* is relevant to *q*, and *p(q|d)* is the likelihood of *q* with respect to *d*, which thus captures how well *d* "fits" *q*. In the simplest case, *p(d)* is assumed to be uniform, and so does not affect document ranking. This assumption has been taken in most existing work (Ponte and Croft, 1998; Song and Croft, 1999; Zhai and Lafferty, 2001a; Cao et al., 2005). In other cases, *p(d)* can be used to capture non-textual information, e.g., the length of a document or links in web page, as well as other format/style features of a document (Kraaij et al., 2002). In our study, we assume *p(d)* as uniform if we use equation 2.7 to rank the documents. In this method, we consider that probabilistic models are only estimated from the documents, while the queries are viewed as observed term sequences. We rank the document according to the logarithm of the query likelihood to avoid underflow when the query is long. Then we have:

$$\log P(q \mid d) = \sum_{i=1}^{n} \log P(q_i \mid d) \tag{2.8}$$

Estimation of the probability $P(q_i \mid d)$ is a key issue in SLM approaches for IR. If we simply calculate this probability by Maximum Likelihood Estimation (MLE), the terms does not occur in the document will be assigned zero probability, so that the document would never be retrieved. Therefore, some smoothing methods should be applied to assign non-zero probabilities to the absent terms. We will describe the smoothing methods in next section. Let us denote the probability of an existing term ("seen") and absent term ("unseen") as $P_s(q_i \mid d)$ and $P_u(q_i \mid d)$ respectively. Then we have:

$$\sum_{i=1}^{n} \log P(q_i \mid d)$$
$$= \sum_{i:c(q_i,d)>0} \log P_s(q_i \mid d) + \sum_{i:c(q_i,d)=0} \log P_u(q_i \mid d)$$
$$= \sum_{i:c(q_i,d)>0} \log P_s(q_i \mid d) - \sum_{i:c(q_i,d)>0} \log P_u(q_i \mid d) + \sum_{i=1}^{n} \log P_u(q_i \mid d)$$
$$= \sum_{i:c(q_i,d)>0} \log \frac{P_s(q_i \mid d)}{P_u(q_i \mid d)} + \sum_{i=1}^{n} \log P_u(q_i \mid d) \qquad (2.9)$$

Without losing generality, let us assume the probability of unseen terms is calculated as the product of the probability of the term in whole collection and a document weight, i.e., $P_u(q_i \mid d) = \alpha_d P(q_i \mid C)$

Substitute it into equation 2.9, we get:

$$\log P(q \mid d) = \sum_{i:c(q_i,d)>0} \log \frac{P_s(q_i \mid d)}{\alpha_d P(q_i \mid C)} + n \log \alpha_d + \sum_{i=1}^{n} \log P(q_i \mid C) \qquad (2.10)$$

In equation 2.10, the right part has three components, and the third component is independent on *d*, so that it can be dropped for ranking purpose. This means that we just need to consider the documents sharing at least one term with the query. Therefore, only a small proportional documents in the whole collection will be counted, which makes the query evaluation process efficient. We are also interested in the first component. As we know, on the first glance, the use of *LM* appears much different from vector space model with *tf-idf* weighting schema, because *LM* seems only encoding term frequency— there appears to be no use of inverse document frequency weighting in the model. However, there is an underlying connection between the *LM* and the traditional heuristics, which can be shown by equation 2.10. In this equation, $\frac{P_s(q_i \mid d)}{\alpha_d P(q_i \mid C)}$ is equivalent to *tf-idf* with $P_s(q_i \mid d)$ corresponds to *tf* while $P(q_i \mid C)$ corresponds to *idf*.

- **KL-Divergence**

We note that in the above approach, only one probabilistic model for the document is built. Alternatively, the second method estimates two probabilistic models, from the documents and queries respectively. The relevance status is thus approximated by the distance between the document model and the particular query model. *KL* divergence (Cover and Thomas, 1991) is a natural way to model the distance between two probabilistic models. Intuitively, the smaller the distance, the more similar the two models are. Therefore, we rank the documents with the negative *KL* divergence. Formally, we define the model as follows: we denote the probabilistic models corresponding to q and d as θ_q and θ_d. Then documents are ranked by the following score:

$$score(d,q) = -KL(\theta_q \| \theta_d) \tag{2.11}$$

In the above equation, if document model θ_d and query model θ_q are assumed to be unigram models, equation 2.4 is then re-written as follows:

$$\begin{aligned} &score(d,q) = -KL(\theta_q \| \theta_d) \\ &= \sum_{w \in V} P(w|\theta_q) \log \frac{P(w|\theta_d)}{P(w|\theta_q)} \propto \sum_{w \in V} P(w|\theta_q) \log P(w|\theta_d) \end{aligned} \tag{2.12}$$

Therefore, the document are in fact ranked by the cross entropy between it and the particular query. If the query model, i.e., $P(w|\theta_q)$, is estimated with MLE, equation 2.12 is equivalent to equation 2.8. Therefore, the query likelihood ranking method is a special case of KL-divergence. However, KL-divergence constructs an explicit query model, which makes it possible to consider the relations between query terms. Therefore, we may prefer KL-divergence in our work.

2.2.4.2 Smoothing Methods

In previous studies (Zhai and Lafferty, 2001b; Zhai and Lafferty, 2002; Liu and Croft, 2004), smoothing of language models has been shown to be an important issue. The retrieval effectiveness is tightly related to which method is employed to smooth the document model. The primary purpose of smoothing is to avoid zero probability. If we

estimate the document without smoothing, i.e., simply with MLE estimation, the terms which do not occur in the document will be assigned zero probability. As a consequence, such a document, even it is relevant, would never be retrieved (no matter to use query likelihood or KL-divergence). Obviously, it is contradictory to our intuition. Smoothing is a technique to assign a small probability to an absent term to avoid zero probability, so that a document containing partial query terms can also be retrieved.

There are three common smoothing methods used, i.e., Jelinek-Mercer Smoothing, Absolute Discounting Smoothing and Dirichlet Smoothing.

$$\text{Jelinek-Mercer:}\quad P(t_i \mid \theta_d) = \lambda P_{ML}(t_i \mid \theta_d) + (1-\lambda) P_{ML}(t_i \mid \theta_C)$$

$$\text{Absolute Discounting:}\quad P(t_i \mid \theta_d) = \frac{\max\{tf(t_i d) - \delta, 0\}}{|d|} + \delta \frac{|d|_U}{|d|} P_{ML}(t_i \mid \theta_C)$$

$$\text{Dirichlet:}\quad P(t_i \mid \theta_d) = \frac{tf(t_i d) + \mu P_{ML}(t_i \mid \theta_C)}{|d| + \mu}$$

where λ is the interpolation parameter and θ_C is the collection model, $tf(t_i, d)$ is the term frequency of t_i in d, $|d|_U$ is the number of unique terms in the document, $|d|$ is the length of d, δ is discount factor, and μ is the Dirichlet prior (or pseudo count). The three parameters, λ, δ and μ can be tuned empirically using a training collection. The parameter λ can also be tuned automatically so as to maximize the likelihood of a set of feedback documents (Zhai and Lafferty, 2002). We will provide more details on this in later chapters.

2.3 Prior Work Go Beyond Term Independence Assumption

There are also some prior work go beyond the term independence assumption. Most of the models proposed and studied in the work are more complicated and less efficient.

Fagan examines how to identify and use non-syntactic (statistical) phrases [35]. Fagan identifies phrases using factors such as the number of times the phrase occurs in the collection and the proximity of the phrase terms. His results suggest no single method of phrase identification consistently yields improvements in retrieval effectiveness across a range of collections. For several collections, significant improvements in effectiveness are achieved when phrases are defined as any two terms within a query or document with unlimited proximity. That is, any two terms that co-occurred within a query or document were considered a phrase. However, for other collections, this definition proved to yield marginal or negative improvements.

In addtion to the unigram model we described above, there are also some other language model variants have been proposed that attempted to model term dependencies. Song and Croft (1999) studied the general n-gram model, such as bigram and even trigram models for information retrieval. Gao et al. (2004) proposed a dependency model which utilizes the query term links to rank document. This dependency model achieved consistent improvement on a set of TREC collections. Recently, Metzler and Croft (2007) used the Markov Random fields model to explore dependency between terms. Wei and Croft (2006) proposed an approach modeling term dependency based co-occurrence. All the models produced some improvements. However, these models can only integrate the statistical relations among terms, such as proximity and co-occurrence.

2.4 Evaluation of Information Retrieval Systems

IR system performance evaluation aims to compare which system is superior to other systems. Usually two aspects are compared: efficiency and effectiveness. Efficiency measures how much computational resource the system requires. The resource includes *CPU* time, memories, storage of hard disk. On the other hand, effectiveness measures to what extent the retrieved documents satisfied the user's need. Effectiveness is more a subjective measurement since two users may have different opinions on one system. Therefore, it is hard to evaluate a real word IR system objectively. However, it is possible to do so with controlled conditions in a laboratory. In the laboratory setting, several test

collections are built to perform evaluation. Each test collection includes a document set which consists of large number of documents, a batch of queries and ***relevance judgments***. The relevance judgments tell which documents are relevant to a query. With a test collection, the system can automate the evaluation process: in each iteration, it first accepts a query; then it calculates the relevance value of each document with respect the query (with one of the models we mentioned in section 2.2); after that, the system return a set of documents with highest relevance values; at last, it evaluates the retrieval effectiveness by comparing its returned list and the relevance judgments. In this report, we focus mainly on experiments performed in the laboratories. In the following, we will describe some common used evaluation metrics and the test collections.

2.4.1 Evaluation Metrics

The effectiveness of IR systems can be evaluated by several measures. The basic measures are precision and recall. They are defined as follows:

- *Recall* is the fraction of the relevant documents which has been retrieved, i.e.,

$$\text{Re}call = \frac{\#\text{ retrieved relevant documents}}{\#\text{ relevant documents in the collection}}$$

- *Precision* is the fraction of the retrieved documents which are relevant, i.e.,

$$\text{Pr}ecision = \frac{\#\text{ retrieved relevant documents}}{\#\text{ retrieved documents}}$$

There is a trade-off between precision and recall: the system with high precision usually has low precision, while a system with high recall usually has low precision. Therefore, just one metric can evaluate an *IR* system thoroughly. In some cases, the

precisions at 11 recall levels are computed and the system is evaluated by the average precision. The 11 recall levels is 0%, 10%, ..., 100%. For the recall level of 0%, the precision is obtained through an interpolation procedure (Baeza-Yates and Ribeiro-Neto, 1999). Another widely accepted measurement for evaluating effectiveness of ranked retrieval systems is the *Mean Average Precision* (*MAP*) (Kraaij et al., 2003), it is defined as:

$$MAP = \frac{1}{M}\sum_{j=1}^{M}\frac{1}{N_j}\sum_{i=1}^{N_j} pr(d_{ij})$$

$$pr(d_{ij}) = \begin{cases} \frac{r_{n_i}}{n_i}, \text{ if } d_{ij} \text{ retrieved and } n_i \le MAX \\ 0 \qquad\qquad \text{otherwise} \end{cases} \tag{2.13}$$

Here, n_i denotes the rank of the document d_{ij} which has been retrieved and is relevant to query j; r_{n_i} is the number of relevant documents found up to and including rank n_i; N_j is the total number of relevant documents of query j; M is the total number of queries and MAX is the cutoff rank (MAX is 1,000 in our experiments).

2.4.2 Standard Benchmarks for Relevance

In past decades, research in information retrieval was often criticized because it lacked robust, consistent and large scale benchmarks. The situation has been improved since the opening of TREC[4] conference in 1992. TREC is the abbreviation of "Text Retrieval Conference". The following words are extracted from the TREC official website to depict the purpose of the conference.

> *It is an on-going series of workshops co-sponsored by the National Institute of Standard Technology (NIST) and the Information Technology Office of the*

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

Defense Advanced Research Projects Agency (DARPA) as a part of TIPSTER Text Program. The annually conference aims to encourage research in information retrieval from large text applications by providing a large test collection, uniform scoring procedures and forum for organizations interested in comparing their results. Attendance at TREC conferences is restricted to those researchers and developers who have performed the TREC retrieval tasks and to selected government personnel from sponsoring agencies.

In each year, the participants of TREC are assigned a task according the specific track they attend. Usually the task includes a set of documents and a batch of queries. The participants are required to run their system, retrieving relevant documents with respect to each query. The documents are tagged with SGML to allow easy parsing. Major structures such as a field for the document number (identified by <DOCNO>) and a field for the document text (identified by <TEXT>) are common to all documents. Minor structures might be different across sub-collections to preserve arts of the structure in the original document. A part of a sample document is shown in Table 3.

Table 2. One Sample Document of TREC Collections

<DOC>
<DOCNO> SJMN91-06364024 </DOCNO>
<ACCESS> 06364024 </ACCESS>
<DESCRIPT> PROFESSIONAL; FOOTBALL; PLAYOFF; GAME; RESULT; BRIEF </DESCRIPT>
<SECTION> Sports </SECTION>
<HEADLINE> RAPID HEARTBEAT FORCES THOMAS TO LEAVE GAME
K.C. STAR IS EXPECTED TO PLAY NEXT WEEKEND </HEADLINE>
<MEMO> Pro Football; AFC Notebook </MEMO>
<TEXT> He was taken to a hospital as a precaution, although his heart rate was back to normal by the time he left the stadium. He remained overnight for
observation. ...
</TEXT>
<EDITION> Morning Final </EDITION>
</DOC>

In each task, TREC usually provides 50 queries. Each query has three fields: title, description and narrative. The participants are free to use any individual field or the combination of them. Here we show a sample of TREC query in Table 4.

Table 3. One Sample Query of TREC Collections

<top> <num> Number: 001 <title> Topic: Antitrust Cases Pending <desc> Description: Document discusses a pending antitrust case. <narr> Narrative: To be relevant, a document will discuss a pending antitrust case and will identify the alleged violation as well as the government entity investigating the case. Identification of the industry and the companies involved is optional. The antitrust investigation must be a result of a complaint, NOT as part of a routine review. </top>

Given the documents and queries, the participants are encouraged to exploiting different techniques to improve their system. Each participant submits the top n (usually, n=1000) documents which receive the highest relevance values for each query. The TREC organizer collects the top ranked documents and constructs the relevant document set for each query. The relevant documents are obtained from a pool of possible relevant documents. This pool is created by taking the top K documents (K is usually set to be 100) in the rankings generated by the various participating retrieval systems. The documents in the pool are then shown to human assessors who ultimately decide on the relevance of each document. As a consequence, each TREC conference can create a set of benchmarks for IR evaluation. Since its opening in 1992, TREC has created a large quantity of benchmarks for various tasks, such Ad-hoc retrieval, Web page retrieval, cross-lingual IR, information filtering, and so on. These benchmarks have been widely used in information retrieval research.

Besides the TREC conference, there are other similar conferences aiming to provide IR benchmarks, such as NTCIR and CLEF. NTCIR[5] is a series of workshops co-sponsored by Japan Society of Promotion of Science (JSPS) as a part of JSPS "Research for Future" program and National Centre for Science Information System since 1997. This conference is organized quite similar to TREC, but it has a preference on East Asian Language Processing, such Japanese, Chinese and Korean. CLEF is a forum aims to

[5] http://research.nii.ac.jp/ntcir/outline/prop-en.html

promote multi-lingual information retrieval among European languages. Interested readers can find more details from http://www.clef-campaign.org/

In the experiments reported in this thesis, we mainly used the TREC data for evaluation. Some cross-lingual IR experiments will also use NTCIR data.

Chapter 3
Exploiting Word Relations for Document Expansion

3.1 INTRODUCTION

In chapter 2, we mentioned some basics of the language modeling approach for information retrieval. In recent years, this approach has increased in popularity, due to its simplicity, clear probabilistic interpretation, as well as efficiency and state-of-the-art performance (Berger and Lafferty, 1999; Lafferty and Zhai, 2001a; Miller et al., 1999; Ponte and Croft, 1998). The key issue in the approach is the estimation of document model. When estimating the document model, the words in the document are assumed to be independent with respect to one another, leading to the so called "bag-of-word" model. However, from our own knowledge of natural language, we know that the assumption of term independence is a matter of mathematical convenience rather than a reality. For example, the words "computer" and "program" are not independent. A query requesting for "computer" might be well satisfied by a document about "program".

Some studies have been carried out to relax the independence assumption. This is generally done in two directions. The first one is data-driven, which tries to capture dependency among terms by statistical information derived from the corpus directly. For example, co-occurrences of terms may be used (Berger and Lafferty, 1999; Gao et al., 2004; Jin et al., 2002; Lafferty and Zhai, 2001; Nallapati and Allan, 2002; Zhai and Lafferty, 2001a). Term dependency can thus be integrated into language modeling.

However, since the dependencies extracted from co-occurrences are blindly obtained from data, much noise can be introduced, which could undermine the retrieval effectiveness. Another direction is to exploit hand-crafted thesauri, such as WordNet (Liu et al., 2002; Mandala et al., 1998; Srikanth and Srikanth, 2002). WordNet has been used to recognize compound terms and dependencies among terms in these studies. The thesaurus is incorporated within classical information retrieval models, such as vector space model and probabilistic model (Robertson et al. 1981). To our knowledge, no one has yet tried to incorporate such a thesaurus within the language modeling framework.

In comparison with relationships extracted from corpora, manually built thesauri only contain manually validated relationships. They are thus less noisy (although ambiguous). In addition, many manually identified relationships can be hardly extracted automatically from corpora. Synonymy relationships are such example: it is difficult to automatically extract the relationship between "query" and "request", as a document would usually use only one term to designate the same object.

In this chapter we propose and study a novel relational language model to incorporate both relationships of WordNet and co-occurrence within the language modeling framework for information retrieval. By considering word relationships, some relevant documents without any query terms may also be retrieved, and we refer these approaches as *Document Expansion*. The possible advantage of our model is twofold: On one hand, we can benefit from WordNet to cover related terms that cannot be identified automatically; on the other hand, we can rely on the manually recognized relationships that are supposed to be more precise, to complement the statistical relationships extracted from co-occurrences, while these latter insure generally a broad coverage of the possible relationships.

One of the difficulties for using WordNet in language modeling is that relations between terms in WorldNet are binary, i.e., one term is linked or not to another term. No weight is associated. When these relations are integrated into a language model, we will have to assign a probability to the link between two terms. A technique relying on term co-occurrences will be used for this. Another problem concerns the combination of

different types of relationships in a language model. We will deal with this problem through language model smoothing.

A series of experiments on standard TREC collections have been conducted to evaluate this method and the experimental results show that our approach is promising: by integrating each type of word relationship, we observe consistent improvements in retrieval effectiveness. This shows that manually built resources such as WordNet, as well as co-occurrence information, can be well incorporated into statistical language models to enhance IR.

The rest of the chapter is organized as: Section 3.2 reviews previous work on relaxing the independence assumption and the utilization of WordNet in information retrieval. Section 3.3 presents our dependency language model to incorporate WordNet and co-occurrence relationships. Section 3.4 discusses the details for estimating model parameters. A serial of experiments on TREC collection are presented in Section 3.5, together with some further discussions. Section 3.6 summarizes the chapter and suggests avenues for future work.

3.2 Previous Work

As mentioned in section 2.2.4, in the classical language modeling approaches (Zhai and Lafferty, 2001a) to IR, a document model $P(w|d)$ over terms is estimated for each document d in the collection C to be indexed and searched. This model is used to assign likelihood to a user's query $q=q_1 q_2 \ldots q_n$. In most cases, each query term is assumed to be independent of the others, so that the query likelihood is estimated by:

$$P(q|d) = \prod_{i=1}^{n} P(q_i|d) \tag{3.1}$$

The above quantity is used to rank the documents. As in speech recognition, a language model for information retrieval must be smoothed to adjust zero probability and small probabilities. Several smoothing strategies are discussed in (Zhai and Lafferty,

2001a). "One of the main effects of smoothing is its robust estimation of common, content-free words that are typically treated as 'stop-words' in many information retrieval systems" (Lafferty and Zhai, 2001). However the classical language model approach for IR does not address the problem of dependence between words.

The term "dependence" may mean two different things: dependence between words within a query or within a document; dependence between query words and document words. Under the first meaning, one may try to recognize the relationships between words in a sentence (either in a document or in a query). In doing so, a sentence is no longer a bag of words. Rather, some dependence will be recognized between words. The approach proposed in (Gao et al., 2004) aims to recognize this type of dependence. Then a query is understood as a set of words, together with some links among them. These links are used as additional criteria to be verified by the documents to be retrieved. Metzler (2007) proposed the Markov Random Fields method to exploit term dependencies. The general framework can considier various co-occurrence and proximity information.

Under the second meaning, dependence means any relationship that can be exploited during query evaluation, such as synonymy, in order to indirectly match a document with a query. For example, for a certain period of time, the document containing the word "Bush" may well answer the query containing the term "president". The relationship between "Bush" and "president" in this example is covered by the second meaning of dependence. Both types of dependence are important for IR. In this chapter, we will concentrate on the second type.

To incorporate term relationships into the document language model, Berger and Lafferty (1999) propose a translation model $t(q_i|w)$ for mapping a document term w to a query term q_i. In fact, the translation probability $t(q_i|w)$ describes the degree of link between the query term q_i and the document word w. With the translation model, the document-to-query model becomes

$$P(q \mid d) = \prod_{i=1}^{n} \sum_{w} t(q_i \mid w) P(w \mid d) \qquad (3.2)$$

Even though their model is more general than other language models, it is difficult to determine the translation probability $t(q_i|w)$ in practice. To solve this problem, Berger and Lafferty generate an artificial collection of "synthetic" data. They compose a short query for each paragraph in a document, and the query consists of some words occurring frequently in the paragraph. The query is assumed to be parallel to its corresponding paragraph. This is indeed a variant use of co-occurrence information, although it is formulated in a different, statistical machine translation setting. Then the synthetic data have the same limitations as co-occurrence information, i.e. only some of the interesting relationships can be extracted (provided that the terms co-occur often enough), and the extracted relationships contain much noise.

Lafferty and Zhai (2001) address this problem differently. They develop a more general model, Markov chain word translation model. It uses a random walk to derive the translation probability $t(q_i|w)$ from a set of documents in the collection. However, this probability is still estimated from term distribution or co-occurrences, without considering other term relationships explicitly. Jin and Hauptman (2002) also propose a different method. They consider a document title as a possible query, and assume that the document is relevant to its title. Then they have a set of document-query pairs to train the translation model between document words and "query" terms

In all of above models, since $t(q_i|w)$ is trained from the document collection, it can only describe the link between terms in the document collection. Several problems arise. The first is that some desired relationships may not be extracted such as true synonymy relationships. The second problem is that virtually, any pair of terms that co-occur within the same document (or paragraph) could be considered to be related. As a consequence, the gain from relaxed independence assumption may not outweigh the loss due to the noise introduced.

The second family of approaches exploits term links stored in a hand-crafted thesaurus, such as WordNet. Voorhees (1994) first exploits WordNet for query expansion. However,

her experiments did not show any gain in retrieval effectiveness when queries are expanded by related terms. In the same vein, Liu et al. (Liu et al., 2004) use WordNet to disambiguate word senses of query terms and to expand queries. In their work, whenever the sense of the query term is determined, its synonyms, hyponyms, words from its definition and its compound words are considered for possible additions to the query.

Instead of using WordNet alone, Mandala et al. (1998) use both WordNet and automatically constructed thesauri to expand queries. They build two thesauri from the corpus, a co-occurrence-based thesaurus and a predicate-argument-based thesaurus, and assign a weight to each associated term pair in the thesauri to represent the degree of association. Since a relation between two terms in WordNet has no weight, they assign it the average of the weights in co-occurrence-based thesaurus and predicate-argument-based thesaurus. They incorporate the three types of relationships within vector space model. Their experiments show that it is useful to combine WordNet with automatically construct thesauri for query expansion, and this results in improvements in retrieval effectiveness.

Intuitively, manually and automatically established relationships are complementary: the first ones are more precise but they have a limited coverage; the second ones have wider coverage but they contain much noise. By combining them in an appropriate way, we can benefit from the advantages of both. Our approach follows the same direction: we try to use both WordNet and relationships extracted from co-occurrences. However, an important difference is that we do not use ad hoc parameters to combine both types of relationship as Mandala et al. did. Instead, we will use a language modeling setting to combine them in a principled manner.

For a different problem – PP-attachment, (Toutanova et al., 2004) uses random walk models that also combine corpus statistics with other types of relationship such as synonymy relationships in WorldNet. In this respect, our approach follows the same direction.

3.3 Document Expansion by Combining WordNet and Co-occurrence

The model proposed by Berger and Lafferty (1999) provides a good general framework. In this paper, we will use a different formulation, which allows us to integrate different types of word relationships.

Given a query q and a document d, the query can be related to the document directly, or they can be related indirectly through some word relationships. An example of the first case is that the document and the query contain the same words. In the second case, a document can contain a different word, but synonymous or related to the one in the query. In this case, the query can still be satisfied by the document. In order to consider both cases into our modeling, we assume that there are two sources to generate a term from a document: one from a dependency model and another from a non-dependency model (which will be a unigram model in our case). Therefore, the likelihood of the query given a document can be expressed as follows:

$$\begin{aligned} P(q \mid d) &= \prod_{i=1}^{n} P(q_i \mid d) \\ &= \prod_{i=1}^{n} [P(q_i, \theta_D \mid d) + P(q_i, \theta_{\bar{D}} \mid d)] \\ &= \prod_{i=1}^{n} [P(q_i \mid d, \theta_D) P(\theta_D \mid d) + P(q_i \mid d, \theta_{\bar{D}}) P(\theta_{\bar{D}} \mid d)] \end{aligned} \tag{3.3}$$

where θ_D is the parameter of dependency model and $\theta_{\bar{D}}$ is the parameter of non-dependency model. $P(\theta_D \mid d)$ and $P(\theta_{\bar{D}} \mid d)$ are the probabilities of choosing the dependency model and non-dependency model respectively. As the non-dependency model tries to capture the direct generation of the query by the document (without considering any word relationships), we can model it by unigram document model, i.e.

$$P(q_i \mid d, \theta_{\bar{D}}) P(\theta_{\bar{D}} \mid d) = P_U(q_i \mid d) P(U \mid d)$$

where $P_U(q_i \mid d)$ is the probability of unigram model, and $P(U|d)$ is the probability to choose the unigram model.

For the dependency model, we imagine a Markov process to generate a query term. First, we select a term in the document randomly. Second, a query term is generated based on the observed term. Here, term relationship enters into play. If the selected term is "computer" at first step, it is more likely to generate "cpu" than "water" in the second step. Therefore we have:

$$P(q_i \mid d, \theta_D) = \sum_{w \in d} P(q_i \mid w) P(w \mid d, \theta_D) \tag{3.4}$$

This formulation is equivalent to that of the translation model of Berger and Lafferty (1999). As for the translation model, we also have the problem of estimating the dependency between two terms, i.e. $P(q_i|w)$. Instead of considering only co-occurrence information as in the previous studies, we take a different approach here. We assume that some word relationships have been manually identified and stored in a linguistic resource (e.g. WordNet), and some other relationships have to be found automatically according to co-occurrences. Therefore, we have at least two different sources of word relationships. A word can be linked to another word through one of them. The global relationship between them can be made by combining both resources together. This combination can be achieved by a linear interpolation smoothing. Thus:

$$P(q_i \mid w) = \lambda P(q_i \mid L, w) + (1 - \lambda) P(q_i \mid \overline{L}, w) \tag{3.5}$$

where $P(q_i \mid L, w)$ is the conditional probability of q_i given w according to WordNet, which is called Link Model; $P(q_i \mid \overline{L}, w)$ is the probability that the link between q_i and w is achieved by other means (in our case, co-occurrences); λ is the interpolation factor, which can be viewed as mixture weight if Equation 3.5 is considered as a two-component mixture model. In our study, we only consider co-occurrence information beside WordNet. So $P(q_i \mid \overline{L}, w)$ is just the co-occurrence model. The estimations of all these models will be explained later.

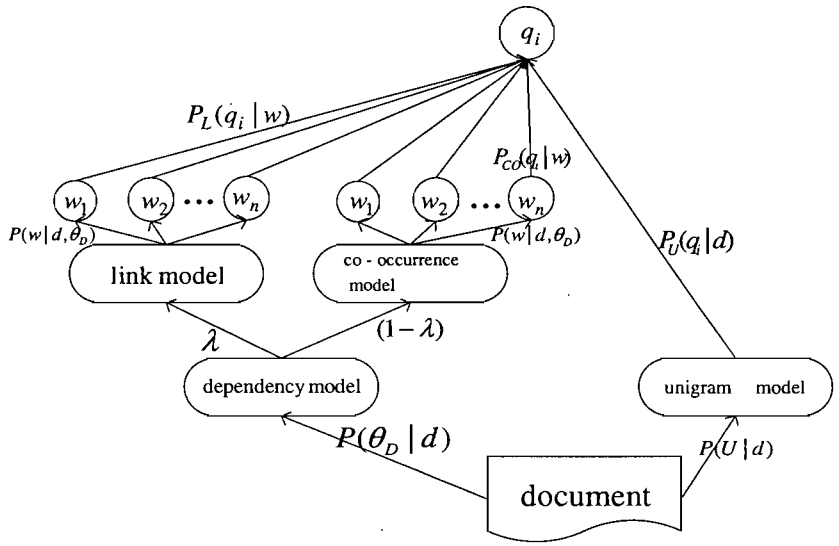

Figure 3. Bayesian Network for Generating a Query Term

For the simplicity of expression, we denote probability of link model as $P_L(q_i \mid w)$, i.e. $P_L(q_i \mid w) = P(q_i \mid L, w)$, and the co-occurrence model as $P_{CO}(q_i \mid w) = P(q_i \mid \overline{L}, w)$ hereafter. Substitute Equations 3.4 and 3.5 into 3.3, we obtain Equation 3.6.

$$
\begin{aligned}
P(q \mid d) &= \prod_{i=1}^{n} [P(q_i \mid d, \theta_D) P(\theta_D \mid d) + P_U(q_i \mid d) P(U \mid d)] \\
&= \prod_{i=1}^{n} [(\sum_{w \in d} P(q_i \mid w) P(w \mid d, \theta_D)) P(\theta_D \mid d) \\
&\qquad + P_U(q_i \mid d) P(U \mid d)] \\
&= \prod_{i=1}^{n} [\lambda P(\theta_D \mid d) \sum_{w \in d} P_L(q_i \mid w) P(w \mid d, \theta_D) \\
&\qquad + (1-\lambda) P(\theta_D \mid d) \sum_{w \in d} P_{CO}(q_i \mid w) P(w \mid d, \theta_D) \\
&\qquad + P_U(q_i \mid d) P(U \mid d)]
\end{aligned}
\tag{3.6}
$$

This equation may seem complicated, but it incorporates a very intuitive idea: the relationship between a document word and a query word can be realized in several ways – direct connection when they are identical, indirect connection either through WordNet or through co-occurrences. Figure 3 gives a Bayesian network illustration of Equation 3.6. The idea can become more obvious if we make some simplification in the formula. Let us define:

$$P_L(q_i \mid d) = \sum_{w \in d} P_L(q_i \mid w) P(w \mid d, \theta_D) \tag{3.7}$$

and

$$P_{CO}(q_i \mid d) = \sum_{w \in d} P_{CO}(q_i \mid w) P(w \mid d, \theta_D) \tag{3.8}$$

Equation 3.7 and 3.8 describe the probability of q_i in d from the link model and co-occurrence model respectively. Then Equation 3.6 can be put into the following simpler form:

$$\begin{aligned} P(q \mid d) = \prod_{i=1}^{n} [& \lambda P(\theta_D \mid d) P_L(q_i \mid d) \\ & + (1 - \lambda) P(\theta_D \mid d) P_{CO}(q_i \mid d) \\ & + P_U(q_i \mid d) P(U \mid d)] \end{aligned} \tag{3.9}$$

Equation 3.9 clearly shows that we have indeed a three-component mixture model consisting of link model, co-occurrence model as well as unigram model. For each component, it has a mixture weight. Let $\lambda_L, \lambda_{co}, \lambda_U$ denote the respect weights of link model, co-occurrence model and unigram model, then Equation 3.9 can be rewritten as:

$$\begin{aligned} P(q \mid d) = \prod_{i=1}^{n} [& \lambda_L P_L(q_i \mid d) + \\ & \lambda_{CO} P_{CO}(q_i \mid d) + \lambda_U P_U(q_i \mid d)] \end{aligned} \tag{3.10}$$

where $\lambda_L = \lambda P(\theta_D \mid d)$, $\lambda_{CO} = (1-\lambda) P(\theta_D \mid d)$ and $\lambda_U = P(U \mid d)$. The above equation defines the general principle of our approach, which places the approach of Mandala et al. into a language modeling framework.

In the above formulation, we consider only one type of relationship in WordNet. Indeed, several types of relationship are stored in WordNet, for example, synonymy relation, hypernymy relation, and so on. Different types of relation should not play the same role. It is more reasonable to separate the link model into several sub-models, each corresponding to a specific type of relation. For information retrieval, the most important terms are nouns, so we concentrate on three relations related to nouns: synonym,

hypernym and hyponym. Let $P_{SYN}(q_i \mid d)$, $P_{HYPE}(q_i \mid d)$ and $P_{HYPO}(q_i \mid d)$ denote the synonym model, hypernym model and hyponym model respectively. Then equation 3.10 can be extended to:

$$P(q \mid d) = \prod_{i=1}^{n} [\lambda_1 P_{SYN}(q_i \mid d) + \lambda_2 P_{HYPE}(q_i \mid d) + \lambda_3 P_{HYPO}(q_i \mid d) + \lambda_4 P_{CO}(q_i \mid d) + \lambda_5 P_U(q_i \mid d)] \quad (3.11)$$

where λ_i $(i = 1,...,5)$ are the mixture weights of the five models. In our discussion, we will refer to the dependency model with non-separated link model (Eq. 3.10) as ***NSLM*** and the one with separated link model (Eq. 3.11) as ***SLM*** hereafter. Now the remaining problem is to estimate the parameters in the models, such as the conditional probabilities, the weights of various models etc. We will discuss these problems in the next section.

3.4 Parameter estimation

In NSLM, 7 terms have to be estimated: $P_U(q_i|d)$, $P(w|d, \theta)$, $P_L(q_i|w)$, $P_{co}(q_i|w)$, and the three mixture weights. In SLM, $P_L(q_i|w)$ is split into three sub-elements, so is the associated mixture weight. So the number amounts to 11. In the following, we only describe the estimation of the parameters in NSLM. Those in SLM can be estimated in a similar way.

3.4.1 Estimating conditional probabilities

The unigram model $P_U(w_i|d)$ can be estimated using any existing method. In our case, we use the ***MLE*** estimation, smoothed by interpolated absolute discount (Zhai and Lafferty, 2001a), that is:

$$P_{abs}(w_i \mid d) = \frac{\max(c(w_i;d) - \delta, 0)}{|d|} + \frac{\delta |d|_u}{|d|} P_{MLE}(w_i \mid C) \quad (3.12)$$

where δ is the discount factor, $|d|$ is the length of the document, $|d|_u$ is the count of unique term in the document, and $P_{MLE}(w_i|C)$ is the maximum likelihood probability of the word in the collection C. This smoothing method is chosen among a set of other smoothing methods (such as Jelinek-Mercer smoothing and Dirichlet smoothing) because we found that this smoothing showed most stable performance in our experiments.

For $P(w|d,\theta_D)$ - the probability of w in document d according to dependency model(s), it can be approximated by the maximum likelihood probability $P_{MLE}(w|d)$. This approximation is motivated by the fact that the word w is primarily generated from d in a way quite independent from the model θ_D.

The key problem now is the estimation of $P_L(w_i|w)$ – the probability of a link between two words according to WordNet. We noticed that WordNet does not provide any weight to relations. So, a naïve method would be to assign the relationship a binary weight (or possibly with a normalization). However, this could not reflect correctly the strength of the connection between the words. Instead, we will rely on the text collection to determine the probability by counting the co-occurrences of these words in the collection. This corresponds to an actualization of the WordNet relations to the given document collection. Some relations will be weighted higher than some others, meaning that the former are more suitable for the topic domain in question. This approach uses a similar idea to that of Mandala et al (1998).

Co-occurrences are observed within some contexts. Using a whole document as co-occurrence context may be too risky: two terms co-occurring in the same document may not be related. To avoid such cases, we limit co-occurrences to a smaller context: the words should co-occur within a window W of certain size.

As many pairs of words in the vocabulary have no link in WordNet, $P_L(w_i|w)$ cannot be calculated by the relative frequency of co-occurrences alone. Smoothing has to be used. We tried four smoothing methods, Jelinek-Mercer, Dirichlet, Absolute Discount and Kneser-Ney as well as two smoothing strategies, backoff and interpolation (Chen and Goodman, 1998). It turns out that interpolated Absolute discount and Kneser-Ney have

the best performance, which is consistent with Chen and Goodman's conclusion (Chen and Goodman, 1998).

Equation (3.13) defines our estimation of $P_L(w_i|w)$ by interpolated Absolute discount:

$$P_L(w_i \mid w) = \frac{\max(c(w_i, w \mid W, L) - \delta, 0)}{\sum_{w_j} c(w_j, w \mid W, L)} + \frac{c(*, w \mid W, L)\delta}{\sum_{w_j} c(w_j, w \mid W, L)} P_{add-one}(w_i \mid W, L)$$
$$P_{add-one}(w_i \mid W, L) = \frac{\sum_{j=1}^{|v|} c(w_i, w_j \mid W, L) + 1}{\sum_{i=1}^{|v|}\sum_{j=1}^{|v|} (c(w_i, w_j \mid W, L) + 1)} \quad (3.13)$$

where w_i and w are assumed to have a relationship in WordNet, $C(w_i, w|W, L)$ is the count of co-occurrences of w_i with w within the predefined window W, and $C(*, w|W, L)$ is the number of unique terms which have a relationship with w_i in WordNet and co-occur with it in W.

Notice that the above estimation is similar to a biterm language model (Srikanth and Srikanth, 2002), in which word co-occurrences are considered without word order. The difference is that we only consider the pairs of words connected in WordNet.

The estimation of the components of the co-occurrence model $P_{co}(w_i|d)$ is similar to those of the link model $P_L(w_i|d)$ except that when counting the co-occurrence frequency, the requirement of having a link in WordNet is removed. It can be calculated by Equation (3.14), also smoothed by interpolated Absolute discount.

$$P_{CO}(w_i \mid w) = \frac{\max(c(w_i, w \mid W) - \delta, 0)}{\sum_{w_j} c(w_j, w \mid W)} + \frac{c(*, w \mid W)\delta}{\sum_{w_j} c(w_j, w \mid W)} P_{add-one}(w_i \mid W)$$
$$P_{add-one}(w_i \mid W) = \frac{\sum_{j=1}^{|v|} c(w_i, w_j \mid W) + 1}{\sum_{i=1}^{|v|}\sum_{j=1}^{|v|} (c(w_i, w_j \mid W) + 1)} \quad (3.14)$$

The estimation of synonym model, hypernym model and hyponym model in SLM follows the same way, except that each type of relation is considered separately in a sub-model.

3.4.2 Estimating mixture weights

In this section we introduce an EM algorithm to estimate the mixture weights in NSLM. Because NSLM is a three-component mixture model, the optimal weights should maximize the likelihood of the queries (Zhai and Lafferty, 2002). For each query q in the dataset (in our case, we use TREC topics 51-100), let $\theta_q = [\lambda_L, \lambda_{CO}, \lambda_U]$ be the mixture weights, we then have:

$$\theta_q^* = \arg\max_{\theta_q} \log \sum_{i=1}^{N} \pi_i \prod_{j=1}^{m} [\lambda_U P_U(q_j \mid d_i) + \lambda_L P_L(q_j \mid d_i) + \lambda_{CO} P_{CO}(q_j \mid d_i)] \quad (3.15)$$

where N is the number of documents in the whole collection, and m is the length of query q. It is also possible to set N to be the number of top ranked documents in a initial retrieval. Here, we consider all documents in the collection to avoid a initial retrieval. $\{\pi_i\}_{i=1}^{N}$ acts as the prior probability with which to choose the document to generate the query. Thus the query is generated from a mixture of N document models with unknown mixing weight $\{\pi_i\}_{i=1}^{N}$. Note that leaving $\{\pi_i\}_{i=1}^{N}$ unfixed is important, because what we really want is not to maximize the likelihood of generating the query from every document in the collection. Instead this maximization is modulated by $\{\pi_i\}_{i=1}^{N}$ which assign some weight to different documents according to their relatedness to the query: the more a document model can generate the query, the more we want to maximize it. With $\{\pi_i\}_{i=1}^{N}$ as free parameters to be estimated, we would indeed allocate higher weight to documents that generate the query well; presumably, these documents are also more likely to be relevant.

The method is similar in principle to pseudo-relevance feedback, which assumes the top n documents to be relevant to the query. Ranking at top level is equivalent to having a high weight in our case. Zhai and Lafferty (2002) employs the same method to learn mixture weights. However, there arises another problem. Some documents having high

weights are not truly relevant to the query. They contain noise. To account for the noise, we further assume that there are two distinctive sources to generate the query, one is the relevant documents, another is a noisy source, which is approximated by the collection *C*. Then Equation (3.15) is rewritten as:

$$\theta_q^* = \arg\max_{\theta_q} \log\left\{\begin{array}{l}(1-\alpha)\sum_{i=1}^{N}\pi_i\prod_{j=1}^{m}[\lambda_U P_U(q_j \mid d_i) \\ \quad + \lambda_L P_L(q_j \mid d_i) + \lambda_{CO} P_{CO}(q_j \mid d_i)] \\ + \alpha\prod_{j=1}^{m}[\lambda_U P_U(q_j \mid C) \\ \quad + \lambda_L P_L(q_j \mid C) + \lambda_{CO} P_{CO}(q_j \mid C)]\end{array}\right\} \tag{3.16}$$

where α is the weight of the noise, $P_U(q_j \mid C)$, $P_L(q_j \mid C)$ and $P_{CO}(q_j \mid C)$ are respectively unigram model, link model and co-occurrence model built from the collection. Here we fix α at a non-zero value, otherwise it would become close to zero because in that way, the documents would have higher likelihood and Equation 3.16 would reduce to Equation 3.15. In fact, the role of α is to add some robustness facing to the noise of the training data. In our experiments, α is set to 0.3. With this setting, the hidden $\{\pi_i\}_{i=1}^{N}$ and θ_q can be estimated using the EM algorithm (Dempster et al., 1977). The update formulas are as follows (we do not give their derivation here due to space limit):

$$\pi_i^{(r+1)} = \frac{\pi_i^{(r)}\prod_{j=1}^{m}[\lambda_U^{(r)} P_U(q_j \mid d_i) + \lambda_L^{(r)} P_L(q_j \mid d_i) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid d_i)]}{\sum_{i=1}^{N}\pi_i^{(r)}\prod_{j=1}^{m}[\lambda_U^{(r)} P_U(q_j \mid d_i) + \lambda_L^{(r)} P_L(q_j \mid d_i) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid d_i)]} \tag{3.17}$$

and

$$\lambda_U^{(r+1)} = \frac{1}{m} \frac{(1-\alpha)\sum_{i=1}^{N} \pi_i^{(r)} \lambda_U^{(r)} P_U(q_j \mid d_i) + \alpha \lambda_U^{(r)} P_U(q_j \mid C)}{\{(1-\alpha)\sum_{i=1}^{N} \pi_i^{(r)} [\lambda_U^{(r)} P_U(q_j \mid d_i) + \lambda_L^{(r)} P_L(q_j \mid d_i) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid d_i)] + \alpha[\lambda_U^{(r)} P_U(q_j \mid C) + \lambda_L^{(r)} P_L(q_j \mid C) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid C)]\}}$$

$$\lambda_L^{(r+1)} = \frac{1}{m} \frac{(1-\alpha)\sum_{i=1}^{N} \pi_i^{(r)} \lambda_L^{(r)} P_L(q_j \mid d_i) + \alpha \lambda_L^{(r)} P_L(q_j \mid C)}{\{(1-\alpha)\sum_{i=1}^{N} \pi_i^{(r)} [\lambda_U^{(r)} P_U(q_j \mid d_i) + \lambda_L^{(r)} P_L(q_j \mid d_i) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid d_i)] + \alpha[\lambda_U^{(r)} P_U(q_j \mid C) + \lambda_L^{(r)} P_L(q_j \mid C) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid C)]\}} \quad (3.18)$$

$$\lambda_{CO}^{(r+1)} = \frac{1}{m} \frac{(1-\alpha)\sum_{i=1}^{N} \pi_i^{(r)} \lambda_{CO}^{(r)} P_{CO}(q_j \mid d_i) + \alpha \lambda_{CO}^{(r)} P_{CO}(q_j \mid C)}{\{(1-\alpha)\sum_{i=1}^{N} \pi_i^{(r)} [\lambda_U^{(r)} P_U(q_j \mid d_i) + \lambda_L^{(r)} P_L(q_j \mid d_i) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid d_i)] + \alpha[\lambda_U^{(r)} P_U(q_j \mid C) + \lambda_L^{(r)} P_L(q_j \mid C) + \lambda_{CO}^{(r)} P_{CO}(q_j \mid C)]\}}$$

The five mixture weights in SLM can also be estimated by EM algorithm in a similar way. We do not list the formulas here.

To terminate the EM iteration, we set a threshold on the change of the log-likelihood of the query: If the change is less than the threshold, EM algorithm stops. In our experiments, we find that EM for NSLM converges very quickly: It usually converges after about 5 iterations. For SLM, it converges after 10 iterations.

The above algorithm is very similar to the one proposed by Zhai and Lafferty (2002) except that we introduce the noisy source into our model. In our experiments, it turns out that setting α to a non-zero value is slightly better than setting it to zero, which shows that it is beneficial to take into account the noise source in the model in an appropriate way.

3.5 Experiments

3.5.1 Experimental setting

Table 4. Statistics of Data Set

Coll.	Description	Size (MB)	# Doc.	Vocab. Size
WSJ	*Wall Street Journal* (1990-92), Disk 2	242	74,520	121,944
AP	*Associate Press* (1988-90), Disks 2&3	729	242,918	245,748
SJM	*San Jose Mercury News* (1991), Disk 3	287	90,257	146,512
Total		1,258	407,695	514,204

We evaluated our model described in the previous sections using three different TREC collections – WSJ, AP and SJM. Some statistics are shown in Table 5. All documents have been processed in a standard manner: terms were stemmed using the Porter stemmer and stopwords were removed. The queries are TREC 51-100. We used the title field and description field of the topics. These queries contain about 15-18 words. The document set comes from the TREC disks 2 and 3.

The version of WordNet we use for experiments is 2.0. For each word in the vocabulary of dataset, we extract its synonym, hypernym and hyponym from WordNet and build a pool of related terms for it. The processing is done offline. When counting the co-occurrences of terms in link model, the pool is used to determine whether the terms have a link. As we do not consider explicitly compound terms, all the compound terms in WordNet are decomposed into their component words.

The baseline of our experiment is the unigram model smoothed by interpolated Absolute discount. In the statistical language modeling approach for IR, there are some free parameters be estimated, for instance, the discount δ. In our experiments, we empirically set the parameters for unigram model by trial and error, and the parameters of the dependency model are blindly set at the same values as in the unigram model. So our dependency model is not tuned to its best. Even though, our dependency model outperforms the baseline substantially.

Table 5. Comparison between Unigram Model and Dependency Model

Coll.	Unigram Model		Dependency Model					
			NSLM			SLM		
	AvgP	Recall	AvgP	% change	Recall	AvgP	% change	Recall
WSJ	0.2466	1659/2172	0.2597	+5.31*	1704/2172	***0.2623***	+6.37*	1719/2172
AP	0.1925	3289/6101	0.2128	+10.54**	3523/6101	***0.2141***	+11.22**	3530/6101
SJM	0.2045	1417/2322	0.2142	+4.74	1572/2322	***0.2155***	+5.38	1558/2322

AvgP is the non-interpolated average precision. * and ** indicate that the difference is statistically significant according to t-test at the level of *p-value* < *0.05* and *p-value* < *0.01*.

The effectiveness of IR is mainly measured by the standard non-interpolated average precision (AvgP). For each query, we retrieve 1000 documents. The total recall (Rec.) for all 50 queries is shown as a complementary metric. We also calculated the t-test for statistical significance and conducted query-by-query analysis.

3.5.2 Experimental Results

We used Lemur3.0 (Ogilvie and Callan, 2001) to carry out experiments. For our purpose, Lemur has been extended to support our document expansion model. The baseline results are obtained directly by using Lemur. Table 6 shows the results of the first group of experiments, in which we compare unigram model with two kinds of models, NSLM and SLM.

We see that document expansion model (both NSLM and SLM) outperforms the unigram model over the three datasets. Specifically, the improvement on AP is greater than 10% and the other two datasets are above 5%. The improvement of WSJ and AP are statistically significant (at the level of *p-value* less than 0.05). The document expansion model also performs well in recall. For each dataset, it retrieves more relevant documents than the unigram model. This is because unigram model only uses direct matching between document and query while our model has the capability to expand the document so as to match different query words. The increase in recall confirms this expansion effect.

Table 6. Different combinations of unigram model, link model and co-occurrence model

Model	WSJ		AP		SJM	
	AvgP	**Recall**	**AvgP**	**Recall**	**AvgP**	**Recall**
UM	0.2466	1659/2172	0.1925	3289/6101	0.2045	1417/2322
CM	0.2205	1700/2172	0.2033	3530/6101	0.1863	1515/2322
LM	*0.2202*	*1502/2172*	*0.1795*	*3275/6101*	*0.1661*	*1309/2322*
UM+CM	0.2527	1700/2172	0.2085	3533/6101	0.2111	1521/2322
UM+LM	0.2542	1690/2172	0.1939	3342/6101	0.2103	1558/2332
UM+CM+LM	***0.2597***	***1704/2172***	***0.2128***	***3523/6101***	***0.2142***	***1572/2322***

We can also observe the difference between NSLM and SLM. It can be seen that differentiating the relations in WordNet (SLM) is better than mixing them (NSLM). We will further discuss this in section 3.5.4.

3.5.3 The role of link model

Compared with previous work on dependency language model, the difference of our work is the introduction of link model based on WordNet. So we conducted experiments to investigate the role of the latter. Table 7 shows the results. Here UM, LM and CM denote unigram model, link model and co-occurrence model respectively. From the table we can see that even though we cannot obtain good results using LM alone (which is expected), it is always helpful to incorporate it in the model: whenever LM is incorporated, we observe some improvements. The combination of all the three models (UM+CM+LM) always outperforms significantly other partial combinations. The results confirm our hypothesis that the relations contained in WordNet (link model) can well complement the statistical relationships extracted from co-occurrences and enhance the retrieval performance. The poor performance obtained when using LM alone may be explained by the fact that LM is too small to include enough information. In fact, in our experiments, LM is usually less than 10 MB, while CM is usually 40 times larger than it.

3.5.4 The role of different relations in the WordNet

Table 7. Average weight for different relations over all queries

Model	WSJ	AP	SJM
UM	0.3564	0.3006	0.4858
CM	0.1480	0.5282	0.1588
SM	0.1657	0.0883	0.1392
HEM	0.1745	0.0491	0.0963
HOM	0.1649	0.0338	0.11968
Total	1.0	1.0	1.0

In section 3.5.2, we draw the conclusion that separating the relations in WordNet and treating them differently results in better effectiveness than treating them without any differentiation. In this section, we investigate the impact of different relations on retrieval effectiveness. Table 8 shows the average weights of different components of SLM over all queries. Here SM, HEM and HOM denote the synonym, hypernym and hyponym models respectively. These weights indicate, to some degree, the contribution of each component to the global performance of the model.

We can see that the relations of WordNet have different contributions in various collections. This may indicate that these relations may be useful for IR at different degrees in different areas.

It is also interesting to observe the correlation between the weights assigned to WordNet relations and the increases that we can obtain when these relations are incorporated (Table 7). For WSJ, we observe quite strong weights for WordNet relations, and we also observe a quite large improvement of UM+LM over UM in Table 7. On the other hand, on AP, the weights assigned to WordNet relations are very weak. We also observe only a marginal of performance change from UM to UM+LM in Table 7 on this collection. This correlation tends to show that the suitability of WordNet to a particular document collection can be automatically determined by the parameter tuning process. In other words, the tuning process is able to determine the appropriate weights for WordNet relations according to their suitability to the area of the documents. Pushing our observation a step further: with an appropriate tuning process, the incorporation of WordNet in our model could not harm retrieval effectiveness. This observation also

applies to other resources such as co-occurrence information. Thus, it could be helpful to incorporate in a retrieval model as many resources of different kinds as possible.

3.6 Summary and future work

In this chapter, we propose and study a novel document expansion approach for information retrieval. In this approach we integrate word relationships into the language modeling framework. Relationships come from two sources: one is from co-occurrences of terms in the dataset and the other is from WordNet.

The advantage of incorporating co-occurrence information in language modeling has been confirmed by several previous studies (Gao et al., 2004; Jin et al., 2002; Lafferty and Zhai, 2002). However, no previous study has investigated a different type of manually defined relationship in language modeling. Our study is motivated by the intuition that the addition of a manual resource can have two advantages: On one hand, we can benefit from such a resource to cover related terms that cannot be discovered automatically; on the other hand, we can rely on the manually recognized relationships that are supposed to be more precise to complement the statistical relationships extracted from co-occurrences. Our experiments confirm this intuition: whenever WordNet is incorporated, we observe some consistent (although variable) increase in retrieval effectiveness. The same observation is also true for the incorporation of co-occurrence information. Then our global conclusion of this study is that it is always better to incorporate more resources of different kinds into a language model for IR, provided that there is an appropriate training process to determine the parameters of the model correctly.

In this chapter, we used EM algorithm to train the parameters. This method worked well for our experiments. In our work described in some later chapters, we will compare different parameter tuning methods, namely, unsupervised tuning using EM and supervised tuning using relevance judgments.

The co-occurrence model used in this study is not sophisticated. It is derived by observing term co-occurrences within texts, without making any filtering of noise. It would be interesting to integrate other more sophisticated methods such as those proposed in (Berger and Lafferty, 1999), (Jin et al., 2002) and (Lafferty and Zhai, 2001) in our link model.

In this chapter, we only studied the relationships between query words and document words. One interesting extension is to also consider the relations between query words or between document words (Gao et al., 2004). This can help solve the problem of ambiguity. A related area is to consider not only single words, but also compound terms in language modeling. This can also create a more precise representation of document contents.

In our work, we assumed that word dependencies are independent of document. This is a simplification assumption. In reality, there is some dependence. So another interesting research direction is to make the dependencies between words dependent on specific document. However, a serious problem concerns the large number of parameters to estimate. This is an interesting issue to be investigated in the future.

Chapter 4
Query Expansion with Markov Chain Models

4.1 INTRODUCTION

In the previous chapter, we expanded the document language model by exploiting term relations. This could be done using the generative model, i.e. to estimate the query likelihood. Similarly, one can also try to apply term relations to expand the query. This expansion process can be integrated into language modeling by using KL-divergence (Lafferty and Zhai, 2001).

As mentioned in section 1.1, the average length of Internet search queries is less than 3 words. With this extremely short query, it is very hard to express user's information need completely and precisely. Query expansion is a technique to improve the query by adding more related terms into it. The related terms can be obtained in different ways, for example, from an external thesaurus (Voorhees, 1994) or from co-occurrence statistics derived from a large corpus (Bai et al., 2005). Whatever related terms are used, the query expansion process can be considered as an inference process based on the term relationships (Nie et al., 2006). The principle is to determine what term can be inferred from the given query and to what extent. Similarly, the document expansion process can be considered as an inference process on the document.

In the previous studies, the above inference process has been implemented in LM either as document expansion (chapter 3) or query expansion (Bai et al., 2005; Xu and Croft, 1996; Zhai and Lafferty, 2001b; Metzler and Croft, 2007).

Although relationships have been used in several previous studies for inferences, their utilization usually has been limited to only one step. For example, they are able to consider that if "computer" is related to "programming", then a query on "computer" can be related to a document on "programming". However, they are unable to conclude that "computer" is also related to "algorithm" if only that "programming" is related to "algorithm" is known. In this chapter, we propose and study an approach based on Markov chain (MC) (Ross, 2003; Brin and Page, 1998; Toutanova et al., 2004) to perform multi-step inferences. In our approach, words correspond to states and word relationships are modeled as state transitions. The stationary distribution of Markov chain corresponds to the final query model. Since the stationary distribution is obtained iteratively, the probabilities of terms related to the query topic will be increased, whereas those of unrelated terms can be reduced. As a consequence, the resulting model is better than the original model, and our experiments will show this.

This chapter is organized as follows. The next section briefly describes the previous studies on pseudo-relevance feedback for IR. In section 4.3, we describe the Markov chain model for query expansion. In section 4.4, we present a series of experiments conducted on three TREC collections, showing the effectiveness of our approach. Finally, we summarize our work and suggest some future research avenues in section 4.5.

4.2 Pseudo-Relevance Feedback

With the KL-divergence ranking approach, we have to estimate two probabilistic models, one for the document and another for the query. Each document d is ranked for a query q as follows:

$$Score\ (d,q) = -\sum_{w_i \in V} P(w_i \mid \theta_q) \log \frac{P(w_i \mid \theta_q)}{P(w_i \mid \theta_d)}$$
$$= \sum_{w_i \in V} P(w_i \mid \theta_q) \log\ P(w_i \mid \theta_d) + c(q) \propto \sum_{w_i \in V} P(w_i \mid \theta_q) \log\ P(w_i \mid \theta_d) \qquad (4.1)$$

where w_i is a word belonging to the vocabulary V, θ_q and θ_d are the query and document model respectively, and $c(Q)$ is a constant independent of D, so it can be omitted for document ranking. While the query model can be estimated by Maximum Likelihood Estimation (MLE), the document model has to be smoothed, usually with the collection model, in order to avoid zero probability for the missing words in a document (Zhai and Lafferty, 2001a). Our work in chapter 3 focuses on how to improve the document model by exploiting word relationships.

As we mentioned, queries submitted by users are usually not good descriptions of users' information needs. So an MLE for query model is also insufficient. Query expansion is an often used technique to add some related terms into the original query. There are a lot of approaches for query expansion. Among them, pseudo-relevance feedback has shown to be effective across retrieval models (Rocchio, 1971; Tao and Zhai, 2006). This approach assumes the top ranked documents in the initial retrieval are relevant to the query. Some key terms are extracted from these documents and used to expand the original query. Several methods have been proposed: the feedback documents can be used to train a new language model, which is then mixed with the original query model (Zhai and Lafferty, 2001b); or they can be used to derive a relevance model (Lavrenko and Croft, 2001). In our preliminary tests, the first mixture model seems to produce better experimental results. So, we will concentrate on the utilization of mixture model for query expansion in this chapter.

In the mixture model, a new feedback model $P(w|\theta_F)$ is estimated from feedback documents, and then mixed with the original query model as follows to form the query model:

$$P(w|\theta_q) = \lambda P(w|\theta_o) + (1-\lambda) P(w|\theta_F) \qquad (4.2)$$

where $P(w|\theta_o)$ is the MLE probability of w in q, i.e., $P(w|\theta_o)=\frac{c(w,q)}{|q|}$ ($c(w,q)$ is the occurrence frequency of w in q, and $|q|$ is the length of q) . The feedback model is estimated by EM in (Zhai and Lafferty, 2001b) in such a way that the likelihood of the feedback documents can be maximized.

In this model, the feedback documents are assumed to be generated from two sources, i.e., the topic model, i.e., $P(w|\theta_F)$, which depicts the user's information need, and a noisy model, which is approximated by the collection model. For simplicity, we assume each term in the feedback documents is generated independently. Then the likelihood of the feedback documents can be calculated as:

$$\ell(F)=\sum_d\sum_w c(w;d)\log\big((1-\lambda)P(w|\theta_F)+\lambda P(w|C)\big) \tag{4.3}$$

with EM algorithm, it is straight forward to have that:

$$\begin{aligned} t^{(n)}(w)&=\frac{(1-\lambda)P^{(n)}(w|\theta_F)}{(1-\lambda)P^{(n)}(w|F)+\lambda P(w|C)}\\ P^{(n+1)}(w|\theta_F)&=\frac{\sum_d c(w;d)t^{(n)}(w)}{\sum_w\sum_d c(w;d)t^{(n)}(w)} \end{aligned} \tag{4.4}$$

Using EM algorithm to estimate the parameters for the multinomial model $P(w|\theta_F)$ is intuitive and ready to be implemented, but it may be time consuming because it finds the solution with an iterative process. Zhang et al. (2002) proposed a fast and direct algorithm to solve this problem, but we did not use the algorithm because it is not easy to be implemented and the EM algorithm is efficient enough for the relatively small scale problem we solve here.

The new query model $P(w|\theta_q)$ now contains the new terms selected from the feedback documents. This model is supposed to be a better description of the user's information need. Due to the added terms, the documents that do not contain the original query terms, but the new terms extracted from the feedback documents, can still be

retrieved. A practical problem in query expansion is the size of the query: when a query is expanded by many terms, the query evaluation time is also increased. Therefore, to limit the size of the query model, one has to limit the number of terms extracted from the feedback documents (for example, 80 strongest related terms).

Although pseudo-relevance feedback has proven to be an effective way to increase retrieval effectiveness, a critical problem is that it requires two steps of retrieval: one to obtain a set of top-ranked documents to extract expansion terms, and another one to retrieve documents with the expanded query. In addition, pseudo-relevance feedback also strongly relies on the assumption that related terms co-occur often in the feedback documents. Therefore, pseudo-relevance feedback exploits implicitly the term relationships encoded by their co-occurrences in the feedback documents. Although many useful term relationships manifest as co-occurrences in the feedback documents, there may be other useful relationships missing from these documents. Therefore, a natural question is how we can extend query expansion beyond the co-occurrence relationships embedded in the top ranked documents. This chapter addresses this question. We will propose several extensions: 1) We will use external resources such as Wordnet to obtain related terms instead of reling on co-occurrence relations only; 2) We will exploit indirect term relations instead of direct relations; 3) We will perform both document expansion and query expansion.

Previous studies have exploited explicitly several types of relationship between terms in different ways for either query expansion or document expansion. We review several ones as follows.

Assuming some relationships $t(q_i|w)$ between two terms, (Berger and Lafferty, 1999) proposed the following translation model to expand document model according to them:

$$P(w|\theta_d) = \lambda P_u(w|d) + (1-\lambda)\sum_{w' \in d} t(w|w')P_{ml}(w'|d) \qquad (4.5)$$

where $P_u(w|d)$ is a classical (smoothed) unigram document model, and $P_{ml}(w|d)$ is the *MLE* document model. In their approach, the probability $t(w|w')$ is estimated as the translation probability from a synthesized data by assuming a sentence to be parallel to the paragraph in which it appears. (Cao et al., 2005) further extends this method by integrating other types of term relationships, namely, co-occurrence relationships and lexical relationships from WordNet.

The above method tries to create a new document model $P(w|\theta_d)$ by integrating term relationships. It is a "document expansion" approach. A similar approach can also be used for query expansion. For example, (Bai et al., 2005) used co-occurrence relationships, as well as inference relationships induced by information flow (Song and Bruza, 2003), to expand query model.

Despite the fact that the above models are able to make inferences according to term relationships, inference has been limited to one step, i.e. only directly related terms are inferred and added during expansion. This limitation is unnecessary. An inference process without the limitation would have higher inference capabilities. A natural extension is to allow for multi-step inference. Markov chain (MC) is a suitable mechanism to implement multi-step inferences.

4.3 Markov Chain Model for Query Expansion

MC has been widely used in several previous studies (Brin and Page, 1998; Toutanova et al., 2004, Minkov et al., 2006). In LM framework, (Lafferty and Zhai, 2001) also uses MC for query expansion. In that paper, transitions between terms are made via documents: a transition from a term to some documents, then from these documents to another term. This method can naturally incorporate the effect of pseudo-relevance feedback, because when performing the transition from a term to documents, the documents can be limited to feedback documents, and the transition from document to

term is similar to query expansion via feedback documents. However, this particular way to estimate term relationships may suffer from the following limitation: it is unable to incorporate other types of term relationships (e.g. those in a thesaurus). In practice, many methods have been developed for extracting various term relationships from text collections, and there are also manually built thesauri that can provide term relationships. Therefore, in this chapter, we will propose a more general model that can integrate term relationships of different types.

MC has also been studied in (Collins-Thompson and Callan, 2005) for query expansion. However, the model uses several heuristics, and it does not exploit fully the capability of MC. The experiments only showed marginal improvement with the approach. In this chapter, we propose a more principled MC model, in which all the parameters will be estimated automatically. Therefore, our model can be easily adapted to different data sets. Our experiments will show that a more rigorous implementation of MC can significantly improve the retrieval effectiveness.

A recent work done by Metzler and Croft (2007) proposed to use Markov Random Fields to select expansion concepts from pseudo-relevant documents. In this model, they integrated some proximity relations between terms, but no semantic relations.

4.3.1 Markov Chain Preliminaries

Before going into the details of our model, let us briefly describe the basic properties of *MCs*. For more detailed information, please see (Brémaud, 1999; Ross, 2003).

A stochastic process $\{S_t\}$ is a family of random variables, where t ranges over an index set T. A Markov process $\{S_t\}$ is a stochastic process over a totally ordered index set satisfying the Markov properties: for any indices $k<t<m$, S_m is independent of S_k given S_t. A discrete time *MC* is a Markov process whose state space S is finite or countable and whose index set T is the set of natural numbers. The Markov property for discrete time *MC* can be written as follows:

$$P(S_t = j \mid S_0 = i_0,\ldots,S_{t-2} = i_{t-2}, S_{t-1} = i_{t-1}) = P(S_t = j \mid S_{t-1} = i_{t-1})$$
$$\forall t, j, i, i_0,\ldots,i_{t-2}, i_{t-1}$$

The *MC* is *stationary* or *time-homogeneous*, if the transition probabilities do not depend on the time *t*. More formally:

$$\forall t, j, i:$$
$$P(S_t = j \mid S_{t-1} = i) = P(S_1 = j \mid S_0 = i)$$

A discrete time stationary *MC* over a set of states *S* is specified by an ***initial distribution*** $p_0(S)$ over *S*, and a set of state transition probabilities $p(S_t|S_{t-1})$. The state transition probabilities can be represented by a matrix *P*, whose entries are $P_{ij} = P(S_t = j \mid S_{t-1} = i)$. A *MC* defines a distribution over sequences of states, via a generative process in which the initial states S_0 is first sampled according to p_0, and then states S_t (for $t=1,2,\ldots$) are sampled according the transition probabilities. Because *MC* used in our work is to model the relationships among terms, from now on, we will use *MC* to mean discrete time-homogenous *MC*.

Let $P(S_t=s)$ denote the probability that the random variables S_t have value *s*. This probability can also be referred to as the probability that the *MC* is in state *s* at time *t*. We can compute the probability distribution $p(S_t)$ at time *t* using the initial distribution and the state transition probabilities in the following way:

$$p(S_0) = p_0$$

$$p(S_1) = p_0 P$$

$$p(S_2) = p(S_1)P = p_0 PP$$

.....

$$p(S_t) = p_0 P^t$$

A *MC* has a *limiting distribution,* or *stationary distribution* π if, for every state *s*, the chain starting at *s* converges to the same distribution π (it is important to note that π is a column vector which assigns a probability to each state). Formally,

$$\forall s: \lim_{t \to +\infty} p(S_t \mid S_0 = s) = \pi$$

A *MC* has a stationary distribution π, if the chain stays in π if it is started according to π. More formally π is a stationary distribution if and only if:

$$\pi = P\pi$$

The *MCs* used in (Brin and Page, 1998; Lafferty and Zhai, 2001; Toutanova et al., 2004) have the property that on each step, there is a probability $\gamma > 0$ of resetting to the initial state distribution p_0 and a probability (1-γ) to transit to the next state. Thus, the state transition probabilities can be written

$$\hat{p}(S_t \mid S_{t-1}) = \gamma p_0(S_t) + (1-\gamma) p(S_t \mid S_{t-1}) \tag{4.6}$$

For some appropriate *p*, this ensures that the *MC* has a limiting distribution, and therefore it also has a unique stationary distribution (Brémaud, 1999).

Given a *MC, M,* with the initial distribution p_0 and the state transition probabilities $p(S_t|S_{t-1})$, we can construct another *MC,* $\hat{M}$. Its initial distribution is P_0, and state transition probability is $\hat{p}(S_t \mid S_{t-1})$. With this setting, it is not difficult to prove the following lemma:

Lemma 4.1:

The stationary distribution of $\hat{M}$ can be calculated as:

$$\hat{\pi}(s) = \lim_{T \to +\infty} \hat{P}(S_T = s)$$

$$= \gamma \sum_{t=0}^{\infty} (1-\gamma)^t P(S_t = s) \tag{4.7}$$

Since $0 < \gamma < 1$, we know from equation 4.7 that $\tilde{\pi}(s)$ must converge to a positive real number. It shows this Markov chain has a unique stationary distribution.

From equation 4.6, the *MC* also has another interpretation: consider the random walk begins from S_0 which is sampled according to the initial state probability P_0. At each step, it stops walking with a probability γ, and continues walking with probability $(1-\gamma)$. Moreover, it transits to another state according to the state transition probability $P(S_t \mid S_{t-1})$.

4.3.2 Query Expansion with MC Models

Let us return to the problem of query representation. A good query can be viewed as a good summary of an information need. So let us consider the process of query generation by a user. To create such a summary, the user first has to select a meaning or a concept to describe; then a term to describe it. Once the first meaning is summarized, he/she can select another related term to describe the same concept further; or choose the next concept to describe. This process corresponds exactly to a Markov chain process we just described.

Assume that q is a query expression; E is the set of potential expansion terms. We define a MC, M, on E to generate query terms. M has initial distribution $P(w \mid \theta_q^0)$ and state transition probability $P(w_i|w_j,q)$. Therefore, the generation of a query can be modeled by a *MC* as follows:

Step 0: The user chooses an initial word according to an initial distribution $P(w \mid \theta_q^0)$

Step *t*: Given the word w_j selected at step *t-1*, the user chooses to add a word w_i. In fact there are two ways to accomplish it: first the user can choose w_i related to an existing word w_j at probability $1-\gamma$, or to add a new unrelated word (i.e., reset to step 0) at probability γ. The selection of the related word is determined

by the transition probability $P(w_i \mid w_j, q)$. So the probability of w_i according to both cases is:

$$\hat{P}(w_i \mid w_j, q) = \gamma P(w \mid \theta_q^0) + (1-\gamma) P(w_i \mid w_j, q) \tag{4.8}$$

Therefore, we in fact define another MC with the initial distribution $P(w \mid \theta_q^0)$ and state transition probability $\hat{P}(w_i \mid w_j, q)$, which is denoted as $\hat{M}$. We allow the above transition process to continue until reaching a fixed point. According to lemma 4.1, $\hat{M}$ is guaranteed to have a stationary distribution $\hat{\pi}(w \mid q)$, which is expressed as follows:

$$\begin{aligned} &\hat{\pi}(w \mid q) = \lim_{T \to +\infty} \hat{P}_T(w \mid q) \\ &= \gamma \sum_{t=0}^{\infty} (1-\gamma)^t P_t(w \mid q)) \end{aligned} \tag{4.9}$$

where $P_t(w \mid q)$ is the state of M after t-th update. The above process can also be interpreted as a random walk: the random walk starts from w_0 which is sampled according to the initial state probability $P(w \mid \theta_q^0)$. At each step, it stops walking with a probability γ, or continues walking with probability $(1-\gamma)$. In the second case, it transits to another state according to the transition probability $P(w_i \mid w_j, q)$. According to its definition, the stationary distribution $\hat{\pi}(w \mid q)$ does not change with T. We interpret a change of probability (by the user) as evidence that the current probability distribution is not yet a good one, and the user has to modify it. For example, the user may have attributed too high a probability to a term, which turns out to be a poor descriptor. The MC model has the ability for mutual reinforce, i.e., the terms which are related to many original query terms will be emphasized, and those related to few original query terms will be deemphasized. So the stationary probability distribution corresponds to a query model which corrects the probability distribution. Therefore, we consider the stationary distribution as the ideal query expression for the user. So, we define the final query model as $\hat{\pi}(w \mid q)$, i.e., $P(w \mid \theta_q) = \hat{\pi}(w \mid q)$

Notice that the transition probability $P(w_i \mid w_j, q)$ is query dependent. Therefore, the transition from w_j to w_i will depend on other words in the query. This is a way to consider the dependence of words within the query, thus relax to some extent the independence assumption between query terms.

4.3.3 Estimation of MC Parameters

As mentioned in section 4.3.1, a *MC* is uniquely determined provided that its initial distribution and transition probabilities are given (Brémaud, 1999; Ross, 2003). Since $\hat{M}$ is induced from M, we only need to define the parameters of M. In this section, we will estimate its parameters and incorporate pseudo-relevance feedback within the estimation.

4.3.3.1 Initial Distribution

The initial probability for word w (or state) is $P_0(w \mid q)$, which can be viewed as the prior probability of w. Because the query is usually very short, it is only a shallow description of user's information need. We assume there is an underlying topic model for the query, which is denoted as θ. The generation of query terms has two sources: the original query and query topic model. In general θ is a random variable depending on Q given the document collection. The initial state distribution is:

$$P(w \mid \theta_q^0) = \lambda P(w \mid \theta_o) + (1-\lambda)\int_\theta P(w \mid \theta) P(\theta \mid q) d\theta$$

where $P(w \mid \theta_o)$ is the *MLE* probability of w in q, and λ is the coefficient of *MLE* model, which is set to be 0.5 for all the following experiments; $P(\theta \mid q)$ is the probability to choose a topic according to q.

In pseudo-relevance feedback approaches, we usually set θ as the topic model for top N retrieved document (Zhai and Lafferty, 2001b). So θ is fixed given q and the retrieval system. We denote this model as θ_F and the feedback documents as F. Therefore, it is

reasonable to assume $P(\theta_F \mid q)=1$ for the given q. Therefore, the above equation can be simplified to the following one:

$$P(w \mid \theta_q^0) = \lambda P(w \mid \theta_o) + (1-\lambda) P(w \mid \theta_F, q) \tag{4.10}$$

We have a number of approaches to estimate $P(w \mid \theta_F, q)$. Here, we use the mixture model presented in (Zhai and Lafferty, 2001b) which produced the best results in our experiments. Therefore, we have: $P(w \mid \theta_F, q) = P(w \mid \theta_F)$, where $P(w \mid \theta_F)$ is calculated in equation 4.4. Nevertheless, any other query expansion model can be extended by *MC*, for example, Lavrenko's relevance model (Lavrenko and Croft, 2001), if the initial distribution is defined on it.

4.3.3.2 Transition Probability considering specificity

Because the possible feedback documents F is more informative for query q compared to the whole collection, we also consider it in defining the transition probability and combine it with the background model via smoothing method. To avoid calculating integration, we also set F to be the top N documents in the first retrieval with original query. Then the transition probability is:

$$P(w_i \mid w_j, q) = \lambda_1^1 P_R(w_i \mid w_j, F) + \lambda_1^2 P_R(w_i \mid w_j) \tag{4.11}$$

where $P_R(w_i \mid w_j, F)$ is the transition probability extracted from F, while $P_R(w_i \mid w_j)$ is from the whole collection. λ_1^1, λ_1^2 are the coefficients for the two models. We will estimate it together with other coefficients in next section with a discriminative training method. $P_R(w_i \mid w_j, F)$ and $P_R(w_i \mid w_j)$ are calculated in the same way except that they are based on different texts. Therefore we only describe how to calculate $P_R(w_i \mid w_j)$ in the following.

In section 3.3, $P_R(w_i \mid w_j)$ is defined as the interpolation between two models: the link model and co-occurrence model. However, we observe that, if we also adopt the definition here, the resulting relationships often suggest common and unrelated expansion

terms. The problem lies in the fact that we only consider the relationships in one direction and not in the reverse direction. This problem can be best illustrated from a logic point of view: The transition probability, $P_R(w_i \mid w_j)$ represents indeed the certainty of inferring w_i from w_j, i.e. $(w_j \rightarrow w_i)$. This estimation is noisy: a term w_j can often entail a more general term w_i, not because the latter is related, but because it often co-occurs with the first one. As a consequence, common word such as *time*, *year*, *mr* etc. are often suggested as expansion terms. In fact, the desired expansion terms are those that are entailed by the original terms, but also entail the latter. In other words, the latter should be ***specific*** to the former. Specificity can be represented as the reverse implication $(w_i \rightarrow w_j)$. Therefore, we propose to integrate both implications. Let us call $P'_R(w_i \mid w_j)$ forward inference relation (*FIR*) and $P'_R(w_j \mid w_i)$ the backward inference relation (*BIR*). By combining them, we have:

$$P_R(w_i \mid w_j) = P'_R(w_i \mid w_j)^{\lambda_2^1} P'_R(w_j \mid w_i)^{\lambda_2^2} / Z \tag{4.12}$$

where $P'_R(w_i \mid w_j)$ represents the probability that w_i can be inferred from w_j; λ_2^1, λ_2^2 are the coefficients of *FIR* and *BIR* respectively; Z is a normalization factor that ensures $\sum_{w_i \in E} P_R(w_i \mid w_j) = 1$. $P'_R(w_i \mid w_j)$ is estimated in the same manner described in section 3.3, i.e.,

$$P'_R(w_i \mid w_j) = \lambda_3^1 P_L(w_i \mid w_j) + \lambda_3^2 P_{CO}(w_i \mid w_j).$$

The addition of specificity is a new extension to the traditional methods that only use *FIR*. The combination of both models can be compared to the logical equivalence. Therefore, the inference process encoded is stricter: a good expansion term should be equivalent to a query term. Similarly, $P_R(w_i \mid w_j, F)$ is estimated as:

$$P_R(w_i \mid w_j, F) = P'_R(w_i \mid w_j, F)^{\lambda_2^1} P'_R(w_j \mid w_i, F)^{\lambda_2^2} / Z \tag{4.13}$$

Equations (4.12) and (4.13) share the same coefficients. All the coefficients meet the constraints: $\lambda_i^j \in [0,1]$ *and* $\lambda_i^1 + \lambda_i^2 = 1$ *for all* $i=1,2,3$. These coefficients will be estimated with a discriminative training method described in section 4.3.4.

4.3.3.3 Estimation of the Unigram Model

The unigram model $P_U(w_i|d)$ can be estimated using any existing method. In our case, we use the ***MLE*** estimation, smoothed by interpolated absolute discount (Zhai and Lafferty, 2001a), that is:

$$P_{abs}(w_i \mid d) = \frac{\max(c(w_i;d) - \delta, 0)}{|d|} + \frac{\delta |d|_u}{|d|} P_{MLE}(w_i \mid C)$$

where δ is the discount factor (set to be 0.5), $|d|$ is the length of the document, $|d|_u$ is the count of unique term in the document, and $P_{MLE}(w_i|C)$ is the maximum likelihood probability of the word in the collection C. This smoothing method is chosen among a set of other smoothing methods (such as Jelinek-Mercer smoothing and Dirichlet smoothing (Zhai and Lafferty, 2001a)) because we found that this smoothing showed most stable performance in our experiments.

4.3.4 Discriminative Training Method to Estimate the Coefficients of Model Combination

We have several coefficients to be estimated: the probability γ in equation 4.8 to stop random walk and λ_i^j in section 4.3.3.2. Because our experimental results in section 4.5.2 show that the retrieval effectiveness is relatively insensitive to γ, here we just tune the parameters λ_i^j. There are two main strategies to optimize parameters: generative methods to maximize the likelihood of queries (or relevant documents) (Zhai and Lafferty, 2001b; Zhai and Lafferty, 2002) and discriminative methods to optimize the mean average precision (MAP) or the rank of relevant documents directly (Gao et al., 2005). Here we

use the latter, i.e., we optimize the parameters to maximize the MAP of training collections. We assume MAP is a function depending on λ_i^j. To avoid constrained optimization, we transform the constraints $\lambda_i^1 + \lambda_i^2 = 1$ into: $\lambda_i^j = \frac{\exp(\gamma_i^j)}{\exp(\gamma_i^1) + \exp(\gamma_i^2)}$ *i=1,2,3 and* $j=1,2$. Then optimization over γ_i^j becomes unconstrained. Therefore, our ultimate object function is:

$$f(\gamma_i^j) = MAP(\gamma_i^j) \tag{4.14}$$

Toutanova et al. (2004) used a similar approach for transformation. In addition, a gradient-descent-like method is used in (Toutanova et al., 2004). However, we cannot use gradient descent methods because $f(\gamma_i^j)$ is not differentiable with respect to γ_i^j. Gao et al. (2005) used line search which tries to look for an optimal value for each parameter in turn while keeping all the other parameters unchanged. However, it is easy to stop at local maximal. Here we use the Simulated Annealing (SA) algorithm (Kirkpatrick et al., 1983) to maximize $f(\gamma_i^j)$. Although *SA* is more time consuming than line search, it avoids being trapped at a local optimal solution. Our experiments show it works well.

4.4 Experiments

Table 8. Statistics of Test Collections

Coll.	Description	Size (MB)	# Doc.	Vocab. Size	Avg Doc Len	Query	
						Testing	Training
AP	*Associate Press* (1988-90), Disks 2&3	729	242,918	245,748	244	TREC topics 51-100 (Title + Desc.)	TREC topics 101-150 +201-250 (Title + Desc.)
WSJ	*Wall Street Journal* (1990-92), Disk 2	242	74,520	121,944	264	As AP	As AP
SJM	*San Jose Mercury News* (1991), Disk 3	287	90,257	146,512	217	As AP	As AP
CH1	People's Daily (91-93) & Xinhua Daily (94-95)	162	164,789	274,901	242	TREC CLIR CH1-28 (Title)	TREC CH29-54 (Title)
CH2	As CH1	As CH1	As CH1	As CH1	As CH1	As CH1 (Title +Desc.)	As CH1 (Title + Desc.)

We used five TREC collections to evaluate our models: three in English and two in Chinese. We used collections with different languages in order to test whether our models

are language independent. Table 9 shows the statistical information of the various collections.

All English documents and queries were processed in a standard manner: terms were stemmed using the Porter stemmer and stopwords were removed. The document set comes from the TREC disks 2 and 3.

The version of WordNet we use for experiments is 2.0. For each word in the vocabulary of dataset, we extract its synonym, hypernym and hyponym from WordNet and build a pool of related terms for it. The processing is done offline. When counting the co-occurrences of terms in link model, the pool is used to determine whether the terms have a link. As we do not consider explicitly compound terms, all the compound terms in WordNet are decomposed into their component words.

For Chinese, the entire dataset (including the documents and queries) was converted into the simplified encoding (GB2312). We carried out dictionary-based word segmentation. This dictionary is compiled by UC Berkley, and it has been used in several TREC experiments on Chinese IR. According to Foo and Li's work [2004], the best results are obtained when most Chinese words are two-character long; we therefore limited the length of the word to no longer than 3 characters. The queries and documents are processed in the same way and we did not filter out any stop word. Because there is no counterpart of WordNet in Chinese, we did not construct the WordNet model, i.e., λ_3^1 is consequently set to 1.

The effectiveness of IR is generally measured by the standard non-interpolated average precision (*AvgP.*). For each query, we retrieve 1000 documents. The total recall (Rec.) for all queries is shown as a complementary metric. We also calculated the *t-test* for statistical significance and conducted query-by-query analysis.

We also used Lemur3.0 (Ogilvie and Callan, 2001) as our retrieval system. For our purposes, this toolkit has been extended to support our experiments.

4.4.1 Results of Query Expansion with MC

Table 9. Comparison different models for query expansion

Coll.	UM		QE			MixM			MC		
	AvP.	Rec.	AvP.	%chg 1	Ret.	AvP.	%chg1	Ret	AvP.	%chg2	Ret.
AP	0.1925	3289/6101	0.1959	+1.76	3370	0.2350	+22.07**	3700	**0.2580**	+9.79*	3994
WSJ	0.2466	1659/2172	0.2483	+0.68	1636	0.2731	+10.75**	1730	**0.2860**	+4.72	1794
SJM	0.2045	1417/2322	0.2142	+4.74	1485	0.2298	+12.37**	1526	**0.2522**	+9.75*	1621

Ret. is the number of retrieved relevant documents. chg1 means the improvement over UM and chg2 means the improvement over MixM. * means the improvement is statistical significant (p-val <0.05) and **means very significant (p-val <0.001)

To examine the performance of MC-based query expansion, we compared the following models:

UM: unigram model. This is the basic LM without any expansion.

QE: the basic query expansion model, which uses term relationships extracted from co-occurrences and from WordNet (for English). This experiments aims to show the contribution of inference in query expansion based on term relationships solely. In fact, the expanded query model is formulized as follows:

$$P(w \mid \theta_q) = \lambda P(w \mid \theta_o) + (1-\lambda)\sum_{w' \in q} P_R(w \mid w')P(w' \mid \theta_o) \qquad (4.15)$$

where $P(w \mid \theta_o)$ is the *MLE* probability of w in query q, and λ is the coefficient which is set by manual trial. $P_R(w \mid w')$ is calculated as equation 4.12. We use 80 expansion terms in this experiment.

MixM: query expansion with the mixture model. We used top 20 documents for feedback and chose 80 terms to add to the query.

MC: query expansion with *MC*. All the parameters *MC* shares with *MixM* are set to be the same. We also set γ in equation 4.6 to be 0.3 for all other four collections

except CH2, in which it is set to be 0.8. All other parameters are tuned by discriminative training described in section 4.3.4.

Table 10 compares the four models. We can see that the basic query expansion model (*QE*) only marginally outperforms the unigram model. For Chinese, in particular, virtually no improvement has been obtained. This result is not really surprising and it is consistent with several studies on query expansion (Voorhees, 1994).

In contrast, we can see in the column *MixM* that the utilization of a feedback model to mix with the original query model is highly effective. This result is consistent with that of (Zhai and Lafferty, 2001b). The consideration of feedback documents clearly allow us to create a better query model.

What is interesting to observe is that, once feedback documents are used to enhance the query model, term relationships become more useful. This can be observed in the column *MC*, in which feedback documents are used in the following two ways: to create a feedback unigram model as initial distribution for *MC*, and to provide a subset of documents from which transition probabilities are extracted. This provides us with more related expansion terms.

The higher improvement obtained with *MC* is as expectable. When we extract term relationships from the feedback documents which are more informative to the query, a filtering has been made. It can be assumed that these documents are strongly related to the query's topic and the relationships extracted from them are also more related to the query. This observation is similar to that on global and local context analysis (Xu and Croft, 1996).

MC also outperforms *MixM*. It brings statistically significant improvements on four collections except WSJ. Our explanation for this is as follows: in the experiments, we only selected 80 terms with the largest initial probabilities and used them for random walk. *MC* is in fact a process to re-estimate term probabilities. Therefore, by defining transition probabilities with word relationships, *MC* increases the probabilities of important terms and their related terms, and this then results in a more accurate query

model. In fact, *MixM* is a special case of *MC* when we set the number of random walk to 0. So the difference between *MC* and *MixM* is directly due to the additions of expansion brought by random walk. This demonstrates the usefulness of random walk in query expansion.

4.4.2 Sensitivity of Stopping Probability in Random Walk

Our model does not optimize the stopping probability (γ in equation 4.8). Now let us examine whether it is sensitive to the performance of *MC*. We carried out a series of experiments, tuning γ from 0 to 1.0, for all the five collections and compared *MC* with *MixM* for all collections at each value of γ. Figure 4 shows the results. We observe that *MC* will outperform *MixM* if $\gamma \in [0.2,1)$ for all four collections except CH2, which requires γ greater than 0.5. Therefore, the performance of *MC* is fairly good even though γ is not optimized, i.e., it is not sensitive to γ.

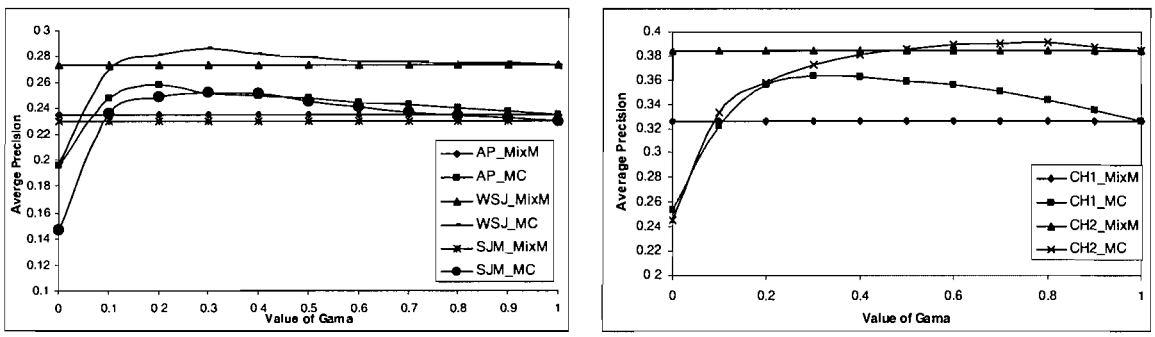

Figure 4. Sensitivity of for MC performance

4.4.3 Multi-step VS Single Step

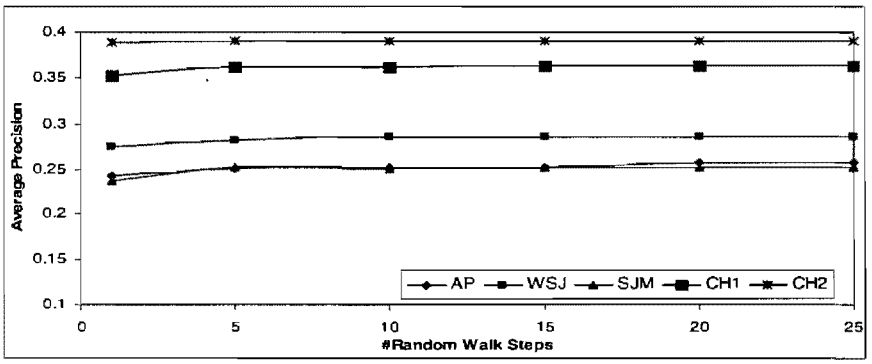

Figure 5. Convergence of Random Walk

In this experiment, we examine whether multi-step inference is better than single step inference. In this experiment, we set number of random walk steps from 1 to 25 and got the average precision for all five collections. The one-step random walk is actually the single step inference, in which we only consider the immediate word relationships. Figure 5 shows the results. From this figure, we observe that multi-step inference is better than single-step inference.

Table 10. Forward inference v.s. bidirectional inference

Coll.	FIM		FIM+BIM		
	AvP.	Ret.	AvP.	%chg.	Ret.
AP	0.2533	3913	**0.2580**	+1.85	3994
WSJ	0.2719	1774	**0.2860**	+5.19*	1794
SJM	0.2433	1614	**0.2522**	+3.66*	1621
CH1	0.3012	1726	**0.3637**	+20.55**	1910
CH2	0.3859	1985	**0.3906**	+1.21*	1993

AvP, Ret. * and ** has the same meaning with table 2; chg. means the improvement over FIM.

Another interesting observation from the figure is the quick convergence of MC. Although MC reaches its stationary distribute at infinite steps, equation 4.9 converges very fast because $\gamma \in [0,1]$. We see that MC converges in less than 20 steps for all the collections; in particular, CH2 converges after 5 steps. Since MC converges very fast and the state set is small (only containing 80 terms), query expansion can be very efficient. In our experiments, we observed that MC model took very limited additional time (only several seconds for each query).

4.4.4 Comparing Forward Inference with Bidirectional Inference

Table 11. Top expansion terms with FIR and FIR+BIR

FIR		FIR+BIR	
play	0.00670988	play	0.0040742
state	0.00670247	unravel	0.00230366
year	0.00274086	president	0.00230058
talk	0.00215404	upcoming	0.00230058
govern	0.00205798	congression	0.00207419
country	0.00205344	contract	0.00189123
nation	0.00184793	territory	0.00172689
president	0.00170653	prosecutor	0.00169048
party	0.00170508	master	0.00163359
million	0.00169628	serious	0.00161677

The addition of reverse implication to account for the specificity of expansion terms brings a notable improvement. Table 11 shows the results obtained with one- and two-directional implications. We can see that the addition of inference is useful on all collections, and is especially effective on CH1 (20.55% improvement). We observe that in general, when *BIR* is not added, many queries are expanded with general terms, which are not useful to identify relevant documents and hurt the performance. Table 12 shows the expansion terms[6] of the model using FIR or FIR+BIR for query #61. The original query is:

Title: Israeli Role in Iran-Contra Affair

Description: Document will discuss the role of Israel in the Iran-Contra Affair.

We excluded the original query terms from Table 12. We can see that in the column *FIR*, many common words, such as *year*, *million*, are expanded; while in *FIR+BIR*, most of the words are specific to the query.

[6] The expansion terms have been stemmed, here we restore the original form for easy understanding

4.5 Summary of Query Expansion with MC

In this chapter, we proposed and evaluated a query expansion model based on Markov Chain. Although *MC* models have been employed for query expansion in previous studies (Lafferty and Zhai, 2001; Collins-Thompson and Callan, 2005), our work is different from them in the following aspects:

- Our model is feasible to integrate arbitrary relationship between terms, while (Lafferty and Zhai, 2001) just consider the relationship between a document and a term.
- Our model has a well-grounded theoretic view, and all the parameters are automatically estimated without any heuristically setting.
- Our model is feasible to consider the multi-step relationships between terms, which can bring further improvement

We also consider the specificity when defining the word transition probabilities, which makes our model more robust in finding expansion candidates. A discriminative training method is used to estimate the parameters automatically. As a consequence, this model is ready to be adapted to other data sets.

A serial of experiments on standard TREC data have conducted to evaluate this model. Experimental results show that our model outperforms the mixture model (Zhai and Lafferty, 2001b) significantly. Moreover, the consideration of specificity in defining the word transition probability also improves the performance.

In this chapter, we apply the MC model for monolingual IR, so the relationships are between monolingual terms. Actually, we can view translation between bilingual terms as another kind of term relationships; therefore the MC model can also be applied to cross-lingual information retrieval. In next chapter, we will address this issue.

Chapter 5
Cross-Lingual Information Retrieval with Markov Chain Models

5.1 Introduction

Cross-Language Information Retrieval (CLIR) has attracted a large number of studies, and a variety of methods for query translation have been proposed (Ballesteros and Croft, 1998; Davis and Ogden, 1997; Lavrenko et al., 2002; Gao and Nie, 2006; Xue et al., 201; Kraaij et al., 2003; Wang and Oard, 2006; Xu and Weischedel, 2000). Many of these methods rely on dictionaries for query translation due to the simplicity of the methods and the availability of machine readable bilingual dictionaries (Gao and Nie, 2006; Gao et al., 2001; Hedlund et al., 2004; Hull and Grefenstette, 1996; Xue and Weischedel, 2005). Some studies have shown that dictionary-based approaches can produce very good CLIR results. However, several problems have also been repeatedly observed in them, and remain unsolved: On the one hand, translation is strongly limited by the coverage of the dictionary, and a manual extension of the dictionary coverage is difficult. On the other hand, even when a dictionary contains all the possible translations for a word, we are still faced with the problem of translation ambiguities. A selection should be made in order to reduce noise (i.e., inappropriate translation candidates). However, dictionaries do not provide any translation reliability measure or context information that can help select the appropriate translations. In most previous studies, dictionaries have been used as the only resource to suggest translation candidates. Although this may result

in reasonable suggestions in many cases, it is not sufficient for query translation in CLIR. In fact, unlike other translation tasks such as full text machine translation, a CLIR query can be translated not only by literal translation words (e.g., words that are stored in a dictionary), but also by semantically similar words. These latter have been found to be very useful to produce a desired query expansion effect (Kraaij, 2003). For example, a literal Chinese translation of the English term "program" is "程序", but the Chinese term "算法" (algorithm) is semantically related to "program" and is also useful for retrieving more relevant documents about "program".

In order to enhance the expansion effect, several studies have used explicit query expansion before and after translation using pseudo-relevance feedback (Ballesteros and Croft, 1998; McNamee and Mayfield, 2002). However, in all the previous studies, the translation step and the expansion step(s) are performed separately, i.e., they are only loosely connected to the IR model. Many parameters have to be set heuristically. In such a case, it is difficult to determine automatically the best settings of these separate steps so as to maximize their global effectiveness. A better method is to define a single model in which both translation and expansion work together to determine semantically related target words, and to use a principled method to determine the parameters automatically.

In this chapter we extend the idea presented in chapter 4 to cross-lingual IR. In chapter 4, we dealt with monolingual query expansion with Markov Chain (MC) models and produce encouraging results. In this chapter, we propose an approach based on MCs to integrate query expansion with query translation. Both monolingual (e.g. co-occurrences) and cross-lingual (e.g. dictionary translation) term relations are integrated into an MC model which is represented as a directed graph. The "translation" of a query is formulated as a random walk in the MC, where monolingual and cross-lingual term similarities are propagated among terms in both languages. This framework has several advantages: (1) It allows us to integrate both translation relations and monolingual relations such as co-occurrence statistics, by which the suggested terms can be translation terms or related target terms. Thus we are able to overcome the limitation by the coverage of the dictionary and to produce a query expansion effect; (2) The multi-step random walk of MC allows us to extend similarity relations from query terms to other indirectly

connected similar terms, which further extends the effect of query expansion; (3) The iterative adjusting of MC will result in a stationary probability distribution, which represents better relations between terms than a coarse initial distribution (we have witnessed this effect in chapter 4 for monolingual query expansion). Truly related target terms are expected to receive higher probabilities after adjusting. (4) There are several methods for automatic tuning of the parameters of MC (Minkov et al., 2006; Toutanova et al., 2004) in principal ways, which avoid us from having to assign parameters heuristically. Therefore, the MC models provide a solution to all the problems mentioned above.

MC has been used in several recent studies for query expansion (Collins-Thompson and Callan, 2005; Lafferty and Zhai, 2001; Cao et al., 2006). The principle is similar to our work in this chapter. However, in previous work, the MC was limited to monolingual terms, while we also integrate translation relations. To our knowledge, this study is the first attempt to apply MC to modeling cross-lingual query expansion.

We evaluated our approach on three TREC and NTCIR collections for English-Chinese CLIR. The experiments show that: (1) the use of MC can indeed lead to better translations than with the traditional approaches; (2) The integration of monolingual word relations can bring further improvements.

This chapter is organized as follows. Section 5.2 describes the background of our method. Section 5.3 presents the MC models for query translation. Section 5.4 presents the estimation of model parameters. The experiments are presented in Section 5.5. Section 5.6 compares our approach with previously proposed methods. Conclusions and future work will be given in Section 5.7.

5.2 Background

Traditionally CLIR has been considered as a two–step procedure: query translation by an external component, and monolingual retrieval (Gao and Nie, 2006; Hedlund et al.,

2004). Recent studies show that the separation of the two steps does not allow us to take into account effectively the uncertainties in each step, and an integrated approach is preferred (Kraaij et al., 2003; Xue et al., 2001). Language modeling has been shown to be an appropriate framework for such integration (Kraaij et al., 2003). In this chapter we follow the same principle, and consider query translation as a step embedded in the construction of the final query model in a language modeling setting. We use negative KL-divergence as the basic document ranking function (Lafferty and Zhai, 2001), defined as follows:

$$\begin{aligned} score(q,d) &= \sum_{w \in q} P(w \mid \theta_q) \log \frac{P(w \mid \theta_d)}{P(w \mid \theta_q)} \\ &\propto \sum_{w \in q} P(w \mid \theta_q) \log P(w \mid \theta_d) \end{aligned} \tag{5.1}$$

where q and d are query and document respectively, and θ_q and θ_d are respectively the parameters of query and document models. By integrating query translation, the above equation is extended to the following one:

$$\begin{aligned} score(q,d) &= \sum_{c \in V} P(c \mid \theta_q) \log P(c \mid \theta_d) \\ &= \sum_{c \in V} \sum_{e \in q} P(c,e \mid \theta_q) \log P(c \mid \theta_d) \\ &= \sum_{c \in V} \sum_{e \in q} P(c \mid e) P(e \mid \theta_q) \log P(c \mid \theta_d) \end{aligned} \tag{5.2}$$

where c is a term in document language (Chinese) and e a term in query language (English).

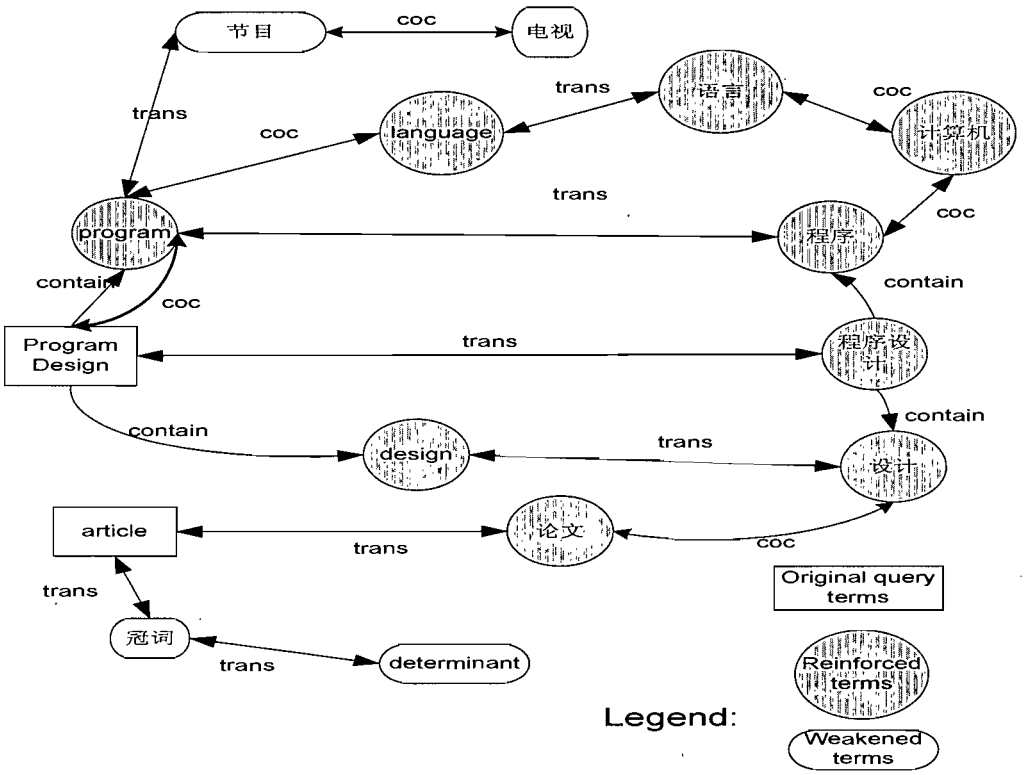

Figure 6. Illustration of Query Translation via Random Walk

Equation (5.2) defines a general language modeling framework for CLIR. The key problem is the estimation of the translation probability $P(c|e)$. It is this estimation that makes our approach different from the others.

Using language modeling framework has been studied by Lavrenko et al. (2002), and Xu et al. (2001). In their models, they did not build an explicit query model, which is different from us. On the other hand, using multiple resources for query translation has been studied by Xu and Weischedel (2005). However, they just combined multiple dictionaries, and the weight for each dictionary was assigned manually.

Due to the lack of the measurement of translation reliability in a dictionary, most previous studies based on dictionary used two naïve methods:

(1) $P(c|e)$ is assigned uniformly over all the candidates stored in the dictionary;

(2) $P(c|e)=1$ if c is the first translation of e in the dictionary and 0 for all other translations.

In some more recent studies, $P(c|e)$ is determined according to more sophisticated criteria such as the coherence between translation candidates (Ballesteros and Croft, 1998; Gao et al., 2001; Gao and Nie, 2006). However, as we mentioned earlier, in all the dictionary-based methods, the estimation of $P(c|e)$ is limited to the translation candidates stored in the dictionary. In order to produce an effect of query expansion, we argue that $P(c|e)$ should not be merely the literal translation probability of c given e, but a cross-lingual semantic similarity between c and e.

If $P(c|e)$ is estimated by a statistical translation model, such as one of IBM models (Brown et al., 1993), trained on a parallel corpus, it reflects cross-lingual term similarities implicitly (Kraaij et al., 2003). However, the reliability for the model to represent such similarities depends on a large degree upon the quality and size of the parallel corpus. Two terms would not be considered as similar terms if they never appear in any parallel sentence pair. Nevertheless, the terms that often co-occur with a literal translation word in parallel texts will receive a small translation probability of the source word. Therefore, a statistical translation model has a capability of producing query expansion effect during translation, by distributing a part of the translation probability to the words that co-occur with the true translation(s).

However, parallel corpora are not widely available for many language pairs (e.g. Chinese-English). Although it is possible to mine parallel materials on the Web for some language (Kraaij et al., 2003; Nie et al., 1999), dictionaries still remain the most available resources for most language pairs. Therefore, we will use dictionaries in our study. However, if a parallel corpus is available, the statistical translation model estimated from it can be easily integrated into our approach.

Notice that the ability of connecting related words in a translation model is a side-effect rather than the desired goal of a statistical translation model – the translation model aims to capture literal translation relations. Its training process tries to limit the possibility

of connecting related non-translation words rather than to favor it. This is contrary to our goal of query "translation" in CLIR, in which we would like to favor the connections to related non-translation terms as well. Therefore, it is desirable to include related words into query "translation".

Pre- and post-translation query expansions have been exploited as a means to perform such an extension (Ballesteros and Croft, 1998; Gao et al., 2001). In pre-translation expansion, the original query is first expanded using a set of feedback documents retrieved in the source language. The expanded query is then translated (e.g. with the help of a dictionary). In post-translation expansion, the translation of the query is used to retrieve a set of feedback documents, which are then used to expand the translated query. In previous studies, both expansion processes have shown some effect on the retrieval effectiveness. However, the expansion steps have been considered to be separated from the retrieval process. They have been used as a means to produce a more appropriate "translation" of the initial query. In these steps, we have to set several parameters manually: the number of feedback documents to be used, the number of terms to be added into the query (or translation), and the weights to be attributed to the additional terms (with respect to the original terms).

In addition to the above practical problems, pre- and post-translation expansion can only consider part of the term relations. As shown in (Xu and Croft, 1996), both global and local analyses can suggest useful terms to expand queries. Using pre- and post-translation expansions, we are indeed using a local analysis, which can suggest related terms appearing in the feedback documents, either in the source or the target language. As shown in (Xu and Croft, 1998), it would be beneficial to add global analysis in the expansion step. Following this work, the global analysis could be used as yet another external component outside the retrieval model. However, as we stated earlier, such a combination is highly dependent on the manually setting of parameters. An alternative is to integrate the term relations extracted from global analysis directly into the model, so that their parameters can be optimized together with those of the translation relations. This means that we extend the methodology of statistical model training to further extending the function $P(c|e)$ from term translation to term similarity relations. To

achieve this goal, in this paper we propose to integrate explicitly different types of terms relation into a MC model.

The utilization of MC for query translation in CLIR is not new. (Monz and Dorr, 2005) used MC to determine the best translation terms. However, only translation relations stored in a dictionary are modeled by the MC. In our case, we integrate other types of term relation in addition to translation relations. In the following section we will describe the details of the model.

5.3 Query Translation as a Random Walk

5.3.1 Principle

Instead of considering query translation as a traditional translation process, now we view it as a process of finding cross-lingual, semantically similar terms. The latter terms can be not only translation terms, but also semantically related terms. Similarly to the principle of pre- and post-translation expansion, related terms can be determined in two ways: they can be target language terms that are related to some translation terms (similar to post-translation expansion), or they can be terms that are translations of related terms in the source language (similar to pre-translation expansion). For example (see Figure 6), given an English (source language) query term "program", besides its literal translation "程序" in Chinese, the Chinese word "计算机" (computer) related to "程序" is also a useful Chinese query term. Similarly, the translation "语言" (language) of a related English term "language" can also be added.

The MC model that we propose tries to integrate the above relations within the source and target languages with translation relations. Our model follows the same principle as pre- and post-translation expansion; but we implement the idea in a very different way. Indeed, we try to determine the related terms in the source and the target languages using a global analysis, i.e. we make use of a global analysis of the whole document corpora,

instead of relying on feedback documents which can only be determined on the fly during the retrieval process. As (Xu and Croft, 1996) showed, it is beneficial to combine global and local term relations. Therefore, even when global term relations are integrated with translation relations in our MC model, it is still possible to use blind feedback to perform local analysis, similarly to pre- and post-translation expansion.

Another major difference between the previous approaches using pre- and post-translation expansions and ours is the integration of the expansion and retrieval processes. In our case, both processes are integrated within the same framework, making it possible to optimize their parameters together and avoiding the necessity to set the parameters manually (as is the case in all the pre- and post-translation expansions).

An additional advantage of using MC is that, given a query, the word relations (either within one language or between two languages) that are strongly related to the query will be reinforced by each other. The final probability distribution after the iterative adaptation of MC is expected to be better for the query than the initial distribution. For example, suppose that the original English query is "articles about program design". A part of the MC is shown in Figure 6. The two key terms in this query "article" and "program design" can be respectively translated by the following words in Chinese:

article: 冠词 (determinant), 论文 (paper), 物品 (object), etc.

program design: 程序设计

In Figure 6, we also show some term relations within the same language (co-occurrence – *coc* and *contain*, see next section). We can see that through monolingual term relations, the correct translation candidates 论文 (paper) for the ambiguous word "article" is more tightly connected to the original query terms. Through the iterative updating, this term will be assigned a higher probability than the other irrelevant translation candidates. On the other hand, the probability of the words which are less related to the original query, such as "节目" ([TV] program), "电视" (TV), "冠词" (determinant) and "determinant", is reduced.

The above example shows that MC also offers a possible solution to the translation ambiguity problem. Indeed, the principle of mutual reinforcement during random walk is used (although to a very limited degree) in some previous approaches to query expansion. For example, (Qiu and Frei, 1993) proposed to determine the expansion terms not according to the strength of their relation with one of the original query terms, but according to their relations to all the query terms. An expansion term having relation with several original query terms will likely be preferred to another one related to only one query term (assuming that their strengths are similar). This approach has proven to be effective.

Transferring the same principle to CLIR, we want to favor translation candidates that are related to more original query terms. In Figure 6, we can see that the translation candidate "论文" (paper) is related to both original query terms (via direct or indirect links). Therefore, its probability is higher than another candidate, "冠词", which is related to only one of the original query terms. This preference is, however, not imposed by using heuristics. Rather, the updating process of MC (Brémaud, 1999; Ross, 2003) can naturally reinforce the more related translation candidates. This is another major advantage of using MC as our model.

5.3.2 Representing Word Relationships with a MC Model

In this section we describe the principle of modeling term similarity in a MC. Each MC model defines a set of states. A state is linked to other states by transitions with different probabilities. Two states are transitional if and only if the transition probability between them is non-zero. A *MC* model is usually represented as a weighted directed graph G as illustrated in Figure 6. It consists of a set of nodes and a set of weighted, directed edges. We use the following notations:

1. A node is denoted as v. We use nodes to represent terms.

2. An edge from v_i to v_j with a label (or relation type) l represents a transition of type l, denoted as: $v_i \xrightarrow{l} v_j$. Each type of edge corresponds to a type of term relation, which will be described later in this section.
3. Each edge $v_i \xrightarrow{l} v_j$ is also assigned a probability $P(v_j|v_i,l)$. This probability will be determined according to different criteria described in Section 5.4.1.

If only translation relations are represented in a MC, the MC model can only assign a translation probability to the translation candidates stored in the dictionary. To extend translation to broader cross-lingual similarity relations, we incorporate two additional monolingual relations: *co-occurrence* and *contain*. The former connects frequently co-occurring terms. It has been shown to be useful for CLIR (Ballesteros and Croft, 1998; Gao and Nie, 2006). The latter considers the relation between a longer term (e.g. *"program design"*) and a shorter constituent term (e.g. *"program"*). This relationship is particular useful for Chinese, which does not have any space between words. Therefore variable word segmentation can be produced for the same character sequence. For example, the sequence "程序设计" (program design) can be segmented either as a single word (in fact a phrase) or as two shorter words "程序" (program) and "设计" (design), depending on circumstances and segmentation programs. If "program design" is translated to "程序设计", it matches directly neither "程序" nor "设计" (for the latter will be considered as different indexes). By considering the *contain* relation, we can link "程序设计", and thus "program design", to "程序" and "设计". This is a way to propagate the translation relation to the constituent terms in the target language.

More types of relation can be integrated in this framework, but we limit our investigation in this study to the three relations: *translation*, *co-occurrence* and *contain*. We will denote them by *trans*, *coc* and *contain*, respectively. The *trans* relations are defined between terms in different languages, the *coc* and *contain* relations between terms of the same language.

Given an MC model, random walk is a process that adjusts the transition probabilities iteratively as follows. In each iteration, we assume a 2-step process of moving from a

node to another. First, from a node v_i (i.e., term) one can select an edge (i.e., relation) l with probability $P(l|v_i)$. We assume here that this selection is independent of v_i, so $P(l|v_i)=P(l)$. This means that the probability of selection of a type of link is the same for all the states (terms). This may not be realistic, but we choose to use this assumption to make the process simpler and more computationally tractable. Second, v_j is chosen by $P(v_j|l,v_i)$. Considering a set L of all possible edge labels (i.e., relations), the probability to arrive at v_j from v_i is

$$P(v_j \mid v_i) = \sum_{l \in L} P(v_j \mid l, v_i) P(l) \tag{5.3}$$

with $\sum_{l \in L} P(l) = 1$.

For example, there are two relations between the terms "program design" and "program" in Figure 6: *coc* and *contain*. The similarity between them is then determined by:

P(program|program design) =

P(program|contain, program design) × ***P(contain)*** + ***P(program|coc, program design)*** × ***P(coc)***

The estimation of $P(v_j \mid l, v_i)$ and $P(l)$ will be described in Section 5.4.

5.3.3 Random Walk for Query Translation

This section describes how query translation is performed as a random walk in an MC model.

The query translation process can be stated as follows. Let θ_q^0 denote the distribution of an original English query, i.e., θ_q^0 gives non-zero probabilities to the nodes corresponding to the English query terms (words or phrases). The translation process

corresponds to the propagation of these probabilities, through random walks, to other terms, especially in the target language. First, a term v_0 is chosen according to the initial distribution θ_q^0. Then one can decide whether to stay in this state (with probability γ) or to transit to another state (with probability 1-γ). The first choice retains the part γ of the probability in v_0, while the second choice transfers (1-γ) of its probability to the related nodes, according to the transition probability (similarity) to them. The process continues in this manner. After k steps of walk, we get a new probability distribution θ_q^k on terms. This latter can be interpreted as the measure of similarity of terms to the original query terms. In particular, the probabilities assigned to target language words are the cross-lingual similarities to the original query.

For the purpose of CLIR, one can normalize the probabilities assigned to the terms in the target language. This would mean that the probabilities of target-language terms will be increased so that they sum to unity. However, this normalization will only affect the query, and it is not document-dependent, i.e. it will not affect document ranking. Therefore, we can just use the probabilities assigned to target-language terms as their translation probabilities.

We notice that the above process interprets the procedure to construct a query-oriented MC model instead of a global MC considering all terms. We call the nodes having non-zero probability in θ_q^k active nodes. Each active node must either be a directly similar node to at least one node corresponding to a term in the original query, or be linked to a node in the query via intermediate nodes. The query-oriented MC model has at least two advantages comparing with the global MC model constructed independently from the query. First, it is much easier to manage and faster to update because the number of active nodes is much smaller than the number of all nodes (the sum of the number of English and Chinese terms). The time required to perform a random walk is thus much shorter and this can be performed online during the query time. In our experiments, it only takes several seconds to update the probabilities and to translate one query. Second, the query-oriented MC model can reduce noise to some degree because it only considers

the nodes related to the query so that it avoids distributing probabilities to non-related nodes.

More specifically, let M_{ij} be the probability of being at node v_j at step $t+1$ given that one is at v_i at step t in the walk, we have:

$$M_{ij} = \begin{cases} (1-\gamma)\sum_{l \in L} P(v_j \mid l, v_i) P(l) & i \neq j \\ \gamma & i = j \end{cases} \tag{5.4}$$

where $L=\{trans, coc, contain\}$ is the set of relationships. If we take *k-step* random walk, the similarity between terms is denoted by $M^{(k)}$, then we have:

$$M_{ij}^{(k)} = \left(\gamma \sum_{t=0}^{k} (1-\gamma)^t M^t\right)_{ij} \tag{5.5}$$

where M is the matrix consisting of M_{ij}. If we set k to be infinite, the MC will reach a stationary distribution, which is considered to be optimal (Lafferty and Zhai, 2001; Minkov et al., 2006; Toutanova et al., 2004; Cao et al., 2006). Since $0 < \gamma < 1$, $M^{(k)}$ is guaranteed to converge. In our experiments, we only consider at most 4 iterations, as the convergence is very fast.

Assume that θ_q^0 is the initial probability distribution of the nodes, then the distribution after a *k-step* walk is proportional to

$$\theta_q^k = \theta_q^0 M^{(k)} \tag{5.6}$$

The document ranking formula for CLIR, i.e., Equation (2), can be re-written as:

$$score(q,d) = \sum_{c \in V} P(c \mid \theta_q^k) \log P(c \mid \theta_d) \tag{5.7}$$

where θ_q^k is given by Equation (5.6) and θ_q^0 is the original parameter setting of query model, i.e., θ_q, in Equation (5.2).

Now, let us illustrate the mutual influence between similar terms during the random walk. Given the MC model in Figure 6, we assume the initial query terms to be "program design" and "article". The literal translations of these terms are "程序设计" (program design) and "论文" (thesis). Other words enclosed in circles are related terms (through mono-lingual relations). In the figure, we see that "article" has two translations "冠词" (article - in linguistics sense) and "论文" (thesis/paper). As "冠词" (article) does not have any other similar terms except "article", its probability will stay low during the random walk. On the other hand, the probabilities of "程序设计", "论文", "程序" and "设计" will be increased, because they will receive probabilities transmitted from related terms. These terms are strongly related to the query, so the effect is desired. This example shows that MC models naturally integrate query expansion and translation.

5.4 Parameter Estimation

In this section we describe in detail how we estimate the parameters of the MC models. We have seven parameters to estimate, i.e., three probabilistic models $P(v_j \mid v_i, l)$, with $l \in \{trans, coc, contain\}$, each for one of the three types of relationship; the three probabilities corresponding to type selection, $P(l)$, as well as the stopping rate γ . In section 5.4.1, we will describe how to estimate the three probabilistic models, and then in Section 5.4.2, we use line search algorithm to estimate the other parameters, in which we optimize one variable while keeping other variables at each time (Gao et al., 2005; Och, 2003).

5.4.1 Probabilities of Relationships

In this study, we use a bilingual dictionary as the translation resource. The probability $P(v_j|v_i, trans)$, i.e., the translation probability between two terms can be estimated in several ways given the bilingual dictionary:

(1) Uniform distribution: we assign equal probabilities to all candidates, that is:

$$P(v_j \mid trans, v_i) = \frac{1}{|\{v_k \mid v_k \textit{ is a translation of } v_i\}|} \tag{5.8}$$

where $|\{.\}|$ is the number of unique elements in a set. This is one of the simple methods used in previous studies, but it may introduce much noise.

(2) Assignment by translation model (GIZA++): a bilingual dictionary can be treated as a parallel corpus: Each English word (or phrase) is aligned to the set of its translations, which is considered as a sentence. We thus can train a statistical translation model using tools such as GIZA++ (Och and Ney, 2000). We only trained IBM model 1 (Brown et al., 1993). This method tries to determine translation probabilities so as to maximize the likelihood of the given sentence alignments. A translation that appears in more aligned "sentences" will be assigned a higher probability than the one that appears in less aligned "sentences". Thus the probability indirectly reflects how often a translation is frequently used between two languages. It is usually more reasonable than the uniform assignment.

The estimation of *contain* relation is similar to the uniform translation model. We count the number of terms v_j which can be a part of the term v_i, and assign the probability uniformly:

$$P(v_j \mid contain, v_i) = \frac{1}{|\{v_k \mid v_k \textit{ is a part of } v_i\}|} \tag{5.9}$$

Monolingual co-occurrence relations can be estimated on large monolingual corpora by counting the number of windows of a fixed size containing the two terms. The English corpora we used are AP88-90 and the Chinese corpora are the document collection that we use for CLIR experiments (see Section 5.5). For two terms v_i and v_j, let $M(v_i, v_j)$ be a measure of closeness of the two terms. Then the relation between v_i and v_j is defined as follows:

$$P(v_j \mid contain, v_i) = \frac{M(v_i, v_j)}{\sum_k M(v_k, v_j)} \tag{5.10}$$

where $M(v_i, v_j)$ can be any statistical metric measuring the association between the two terms such as relative frequency (as the one we used in chapter 3 and 4), mutual information, information gain and log-likelihood ratio (Dunning, 1993). We use log-likelihood ratio because it produced the best results in our experiments. To filter noise, we only keep the 30 strongest co-occurring terms for each term.

5.4.2 Parameter Tuning

We estimate the probability of selecting each of the three relationships, *i.e.*, $P(l)$ *and* $l \in \{trans, coc, contain\}$ and the stopping rate γ. For estimating these parameters, various methods can be used, such as Gradient descent-like approaches (Diligenti et al., 2005; Toutanova et al., 2004), Boosting algorithm (Minkov et al., 2006), and so on. However, the objective functions used in these methods only are loosely related to the Mean Average Precision (MAP) which is used to measure the effectiveness of IR systems. Here we choose an alternative approach based on line search to optimize the parameters so as to maximize the MAP on training data directly. This approach has been used in (Gao et al., 2005; Morgan et al., 2004; Metzler and Croft, 2005) and proven to be very effective. Let us denote the three parameters by a vector $\theta \in \{P(trans), P(coc), P(contain)\}$, and each dimension of the model θ is denoted as $\theta_i, i = 1,2,3,4$.

Given a test collection with relevance judgments for a set of queries, the MAP resulting from θ is denoted by $MAP(\theta)$. The learning approach can thus be formulated as:

$$\theta^* = \arg\max_\theta MAP(\theta) \tag{5.11}$$

The optimization problem can be cast as the multi-dimensional function optimization algorithm (Gao et al., 2005; Och, 2003). The procedure works as follows: $\theta_i, i = 1,2,3,4$ are taken as a set of directions. Line search moves along the first direction while keeping the

other unchanged, so as to maximize the MAP; then it moves from there along the second direction to maximize the MAP, and so on.

Cycling through the whole set of directions as many times as necessary, until the MAP stops to increase, we obtain the values of the parameters. This method is intuitive and efficient, but it may converge to different local maxima with different start points. Therefore, we perform the procedure multiple times with random start points, and select the parameters that produce the best MAP. $P(l)$ is normalized to become a probability.

5.5 Experiments

Table 12. Statistical Information of Dataset

Coll	Description	#Doc	#Qry
TREC5&6	People's Daily (1991-1993) & Xinhua News Agency (1994-1995)	164,789	54
TREC9	HongKong Commercial Daily News, HongKong Daily News and Takungpao News	127,938	25
NTCIR3	Chinese Times, Central Daily News, China Daily and United Daily News	381,681	50

5.5.1 Experimental Setting

We evaluated the MC models with three benchmark English to Chinese CLIR collections: TREC5&6, TREC9 and NTCIR3. Table 13 shows the statistical information of these collections.

We conducted our experiments using cross-validation: The models evaluated on the TREC9 collection were learned on TREC5&6 datasets; the models evaluated on TREC5-6 collections were trained on the TREC9 dataset; the models evaluated on NTCIR3 were trained on both TREC9 and TREC5&6.

All Chinese documents and the translated queries are segmented using dictionary-based approach. The Chinese dictionary was compiled by UC Berkeley, which contains 137,613 words. When indexing document collections, we used all possible words in the

dictionary and all single Chinese characters as indexing units (Kwok, 2000). All English queries are stemmed with Porter stemmer and the stop words are removed. Since we do not have a phrase recognizer, we only recognize phrases stored in our bilingual dictionary. Each query in TREC and NTCIR collection has three fields: title, description and narrative. We used two versions of queries: short queries that contain only titles and long queries that contain all the three fields.

Model	Short Query				Long Query			
	MAP	% of ML	Imp. Over UM	Imp. Over GizaM	MAP	% of ML	Imp. Over UM	Imp. Over GizaM
ML	0.3754		-----	-----	0.4929	-----	-----	-----
UM	0.1281	34.12%	-----	-----	0.2708	54.94%	-----	-----
FM	0.1325	35.03%	3.43%	-----	0.2734	55.47%	0.96%	-----
GizaM	0.3414	90.94%	166.5%**	-----	0.4341	88.07%	60.30%**	-----
UM+MC	0.2918	77.73%	127.8%**	-17.45%	0.4463	90.55%	64.80%**	2.81%
GizaM+MC	0.3720	99.09%	190.3%**	8.96%*	0.4594	93.30%	69.64%**	5.82%

Table 13. Compare Different Model for TREC5&6 Collection

Model	Short Query				Long Query			
	MAP	% of ML	Imp. Over UM	Imp. Over GizaM	MAP	% of ML	Imp. Over UM	Imp. Over GizaM
ML	0.2819	-----	-----	-----	0.2961	-----	-----	-----
UM	0.0976	34.62%	-----	-----	0.1110	37.49%	-----	-----
FM	0.1220	43.28%	24.99%	-----	0.1354	45.73%	21.98%	-----
GizaM	0.2542	90.17%	160.5%**	-----	0.2693	90.95%	142.6%**	-----
UM+MC	0.2750	97.55%	181.7%**	8.18%	0.2622	88.55%	136.2%**	-2.63%
GizaM+MC	0.2897	102.77%	196.8%**	13.97%*	0.2730	92.20%	145.9%**	13.74%*

Table 14. Compare Different Model for TREC9 Collection

Model	Short Query				Long Query			
	MAP	% of ML	Imp. Over UM	Imp. Over GizaM	MAP	% of ML	Imp. Over UM	Imp. Over GizaM
ML	0.2222	-----	-----	-----	0.2840	-----	-----	-----
UM	0.0626	28.17%	-----	-----	0.1212	42.68%	-----	-----
FM	0.0611	27.50%	-2.40%	-----	0.1460	51.41%	20.46%	-----
GizaM	0.1422	63.99%	127.1%**	-----	0.1800	63.38%	48.51%**	-----
UM+MC	0.1442	64.90%	130.3%**	1.41%	0.1987	69.96%	63.94%**	10.38%
GizaM+MC	0.1489	67.01%	137.8%**	4.71%	0.2130	75%	75.74%**	18.33%*

Table 15. Compare Different Model for NTCIR3 Collection

We empirically defined different window sizes to extract term co-occurrences from English and Chinese corpora, respectively. The window width is 8 words for English, and 10 characters for Chinese.

The bilingual dictionary we used is a combination of two human compiled bilingual lexicons, including the LDC English-Chinese dictionary and a bilingual lexicon generated form a parallel corpus. The dictionary contains 123,747 English entries, including 108,799 words and 1,4948 phrases.

We have developed an experimental IR system based on Lemur 4.2 (Ogilvie and Callan, 2001). The main evaluation metric is the Mean Average Precision (MAP). Different from TREC evaluation, NTCIR uses two relevance judgments: *rigid relevance* which only considers highly relevant documents (similarly to TREC), and *relaxed relevance* which also considers partially relevant documents. We use rigid relevance for our evaluation. T-test is also conducted for significance test. We will try to answer several questions in our tests.

5.5.2 Does the MC Model work for CLIR?

In this section we present comparison results of the MC models with other traditional CLIR models. Tables 14, 15 and 16 show the main results on the three collections using short and long queries. Two variants of the MC models are tested, in which the translation probability is respectively the uniform probability and the translation probability generated by applying GIZA++ on the dictionary. To evaluate the effectiveness of the MC model, four baselines are compared:

ML (Monolingual). In this model, the documents are retrieved with the manually translated Chinese query set provided in the collections. Its performance is usually considered as the upper bound of CLIR.

UM (Uniform Model). This model assigns a uniform distribution of translation probability to all the translation candidates stored in the dictionary. When

translating an English query, if we encounter an English phrase in the query that exists in the dictionary, then the phrase translations are used; otherwise, translation of single words are used.

FM (First-one Model). The total translation probability is distributed to the first translation candidate. As UM, phrase translation is used in preference to word translation.

Table 16. Translation obtained by Each Models for One Query

English query: *Forest Railway in Mount Ali* ML: 阿里 山 森林 火车 FM: 森林 (forest) 0.5; 林 (woods) 0.5;路线 (road) 0.5; 轨道 (rail) 0.5; 登上 (go up) 0.5 .... UM: 森林 (forest) 0.5; 造林 (plant trees) 0.5; 山林 (forest in the mountain) 0.5; 植树 (plant trees) 0.5 ; 野 (wild) 0.5... GizaM: 林 (woods) 0.657819;森 (forest)0.161144; 铁 (iron) 0.455182;铁路 (railway) 0.295346; 装 (set up) 0.395038; 安 (install) 0.222885; 阿里(Ali) 0.293588... UM+MC: 铁路 (railway) 0.0565199; 森林 (forest) 0.0528217; 经铁路 (via railway) 0.0508112; 铁道 (railway) 0.0508112;阿里 (Ali)0.049161; 蒸汽 (steam) 0.0241262... GizaM+MC: 林 (woods) 0.10024;铁路 (railway) 0.0651897; 阿里 (Ali) 0.0606648; 森 (forest) 0.0403337;森林 (forest)0.0367658; 嘉义 (Jiayi) 0.021745...
(Note: The probability of translations in FM is 0.5 because we used the first translations from two different dictionaries.)

The above two methods may be too simplistic to serve as a state-of-the-art baseline methods. Nevertheless, we include them in the tables because many previous studies used these methods. A more reasonable baseline method is the following one:

GizaM (GIZA Model). The translation probabilities in this model are obtained with the GIZA++ toolkit, which extracts a statistical translation model from the bilingual dictionary, considered as a parallel corpus. GizaM model considers the frequency of translation of one word. If a translation appears several times, either as a translation item for the given word alone, or as a part of a translation of a compound term containing the given word, then the translation word will be assigned a higher probability. Some previous studies (Grefenstette, 1999) have exploited the frequency of translation terms in a document collection in order to

select the most frequent translation word. The GizaM model exploits a similar principle, by assuming that the more a translation word corresponds to a source word in the dictionary, the more it is a frequent one and thus should be favored.

As we can see in Tables 14-16, this model is a reasonable baseline because it results in retrieval effectiveness comparable to most of the previous studies on the same test collections (Gao and Nie, 2006; Gao et al., 2001; He and Gao, 2001; Xu and Weischedel, 2005; Xu et al., 2001).

Once the translation model is trained on the dictionary, we select the top 10 translations for each term for the short queries and top 3 for long queries. These same numbers are selected for the following two MC models.

UM+MC. The queries are translated with MC model. The initial translation probabilities are obtained from UM.

GizaM+MC. This model is similar to UM+MC, but the initial translation probabilities are obtained from GizaM.

From Tables 14-16, we find that UM performed the worst among all the methods. This is because it treated all translation candidates of a query term equivalently and introduced much noise (irrelevant translation terms).

FM performed slightly better than UM in almost all runs except for short queries of NTCIR3. The reason is that FM only selects the first candidate, which is often the more frequently used translation. This method can avoid including noise translation candidates to some degree. However, this "aggressive" selection can also remove relevant translation terms, whereby limiting the desirable query expansion effect.

GizaM can assign a translation probability between two terms according to how often one appears as a translation of another in the dictionary. The translation probabilities have been trained using the EM algorithm (Dempster et al., 1977) to maximize the likelihood of translating each English term by its Chinese translations (the parallel

sentence in the dictionary). The advantage of GizaM is that it can assign a strong probability to a translation term if the latter is a specific and unambiguous translation term of the former. However, in GizaM, the whole translation probability is still distributed only to the translations stored in the dictionary. In the above tables, we can see that GizaM performed fairly well. Its effectiveness is around 90% of that of ML in four runs (both short and long queries) of TREC5&6 and TREC9.

For MC models, we observe that the two MC variants are all promising: UM+MC model outperformed UM significantly in all the six runs. GizaM+MC outperformed GizaM in all runs, and it even outperforms the ML for short queries of TREC9. This result confirms the advantages of our MC approach. To see better where the superior effectiveness comes from, let us analyze the example shown in table 17 for the query "forest railway in Mount Ali".

In this example, *mount* is translated by FM incorrectly as a verb. For the UM model, we only list the translations of "forest" and we can observe that many translations are unrelated to the query. GizaM seems to be able to distribute strong probabilities to related translation terms. Compared with UM and GizaM, the probabilities assigned by MC models seem generally more appropriate. In addition, they can also suggest some non-translation but related words such as 蒸汽 (steam) and 嘉义 (Jiayi) which is a city connecting Mount Ali. The example confirms the two advantages of MC that we expected:

1. The integration of more term relations can extend translation to broader similar terms, thus producing larger query expansion effect;
2. The iterative probability adjustment process can produce a better probability distribution.

The above results are produced with a random walk of 4 steps for UM+MC and 2 steps for GizaM+MC. We observed that the performance was improved when increasing the steps. This indicates that iterative adjusting similarities between terms are useful for retrieval. We also observed that UM+MC outperformed GizaM in four runs (i.e., long

query of TREC5&6, short query of TREC9 and two runs of NTCIR3) and achieved comparable results with UM+MC in the other two runs. This shows that MC models can capture the same characteristics as GizaM. Indeed, both models work with similar principles: They use an iterative learning procedure to assign a high probability to strong translation candidates. Therefore, UM+MC and GizaM performed similarly. However, GizaM+MC can further improve the performance of GizaM in most of the cases. The difference between them is directly attributed to the addition of more relations in GizaM+MC.

5.5.3 The impact of Different Relationships

Table 17. Different Relation Combinations for long queries

Relation	TREC5&6		TREC9		NTCIR3	
	MAP	Imp. Over T	MAP	Imp. Over T	MAP	Imp. Over T
UM	0.2708	-----	0.1110	-----	0.1212	-----
T	0.4372	-----	0.2431	-----	0.1904	-----
T+C	0.4458	1.97%	0.2618	7.69%	0.1927	1.21%
T+Con	0.4391	0.43%	0.2578	6.05%	0.1987	4.36%
T+C+Con	0.4463	2.08%	0.2622	7.86%	0.1987	4.36%

Table 18. Different Relation Combinations for short queries

Relation	TREC5&6		TREC9		NTCIR3	
	MAP	Imp. Over T	MAP	Imp. Over T	MAP	Imp. Over T
UM	0.1281	-----	0.0976	-----	0.0626	-----
T	0.2761	-----	0.2616	-----	0.1257	-----
T+C	0.2902	5.10%*	0.2719	0.11%	0.1431	13.84%
T+Con	0.2829	2.46%	0.2746	1.10%	0.1267	7.95%
T+C+Con	0.2918	5.68%*	0.2750	1.25%	0.1442	14.71%

In this section we investigate the impact of different relationships on the retrieval effectiveness. Tables 18 and 19 show the results of MC models with uniform translation probability (UM+MC) on the three collections. In the tables, UM is the uniform model mentioned in section 5.5.2; T represents the MC model only using translation relation; T+C represents model using translation relation plus co-occurrence relation; T+Con represents the model using both translation relation and contain relation; T+C+Con represents the model using all three relations. The T model is indeed equivalent to the one used in (Monz and Dorr, 2005).

The tables show that the T model outperforms UM substantially. This can be explained by two reasons. First, after propagating the similarity via a random walk, the translation distribution in the T model is changed from uniform distribution to the one which assigns a higher probability to a term if it is connected to many query terms. Second, we only select the top m terms as query translation. This helps filter out some noise (which typically has a low probability).

Indeed, as we mentioned earlier, UM is too simplistic to serve as a baseline method. However, T is a reasonable baseline method, which corresponds to the state-of-the-art (Monz and Dorr, 2005).

We observed that when more relationships are added into the MC model, the effectiveness is further improved. The best model is the one that uses all the three relationships. On the other MC model, GizaM+MC, we have observed a similar behavior The experimental results confirm our hypotheses: 1) Integrating more term relations than translation can improve query translation in CLIR; 2) Using an iterative random walk process in MC leads to a more reasonable probability distribution.

5.6 Related Work

The MC model we used here integrates both query translation and query expansion in a unified framework. Query expansion has been investigated in the context of CLIR in a number of previous studies. Ballesteros and Croft (1999) explored query expansion methods for CLIR by combining pre- and post-translation expansion, and they found that the method can effectively improve retrieval effectiveness. McNamee and Mayfield conducted a series of experiments to compare CLIR query expansion techniques (McNamee and Mayfield, 2002). They also found similar results to (Ballesteros and Croft, 1998). The pre- and post-translation expansions are conceptually similar to our addition of more term relations. Thus our experiments confirm their observation.

However, our work is different from the above two in the following aspects:

1). Pre- and post-translation expansions have been separated from translation. In fact, as illustrated in (McNamee and Mayfield, 2002), their models are divided into three phases (pre-translation expansion, query translation, and post-translation expansion), that have been handled independently. In contrast, our MC model incorporates the three phases together within the same framework.

2). Comparing to the expansion process, our MC model-based approach is theoretically more sound, and easier to extend. We also used a principled way to optimize all parameters.

MC models have been used for many other tasks. (Minkov et al., 2006) used the random walk model to disambiguate person's names in e-mails, but the relationships in their model are binary. In IR, infinite random walks have been used for document or webpage re-ranking (Kurland and Lee, 2005; Page et al., 1998). The idea of representing semantic similarities by a graph has also been used in NLP and IR. (Lafferty and Zhai, 2001; Collins-Thompson and Callan, 2005) used a random walk model for monolingual query expansion. But they only use one type of relationship. In chapter 4, we presented a MC based model for query expansion, which is feasible to integrate multiple relations. (Toutanova et al., 2004) presented a MC model for pp-attachment disambiguation. (Monz and Dorr, 2005) used MC for query translation in CLIR. However, the MC is built on a dictionary, so translation suggestions are bounded by the dictionary. In our case, we extended the translation relations to cross-language semantic similarity relations. In so doing, we can create more effect of query expansion.

5.7 Summary and Future Work

CLIR is different from traditional monolingual IR in that it requires query translation. Dictionary-based approaches are widely used to translate queries in CLIR because of their simplicity and the availability of machine-readable dictionaries. However, we are faced with several problems: limited coverage and lack of a measurement for the reliability of the translation candidates. On the other hand, query translation in CLIR is

different from text translation because the translation terms selected in CLIR are not necessarily to be literal translations. It just needs to be semantically similar terms. In this chapter we extended the MC model proposed in chapter 4 to cross lingual context, which integrates several types of monolingual term relation, in addition to the translation relation. As a result, query translation is extended to cross-lingual query expansion.

As used for monolingual query expansion, the MC models also adjust the probabilities of terms automatically through a random walk. We showed in our experiments that the final distribution produces higher retrieval effectiveness than the original one. This shows that the random walk can effectively adjust terms' cross-lingual similarity to the query so that strongly related target terms are assigned higher probabilities.

In this chapter we only investigate three types of relation: translation, co-occurrence and containment. However, the method can be easily extended to include more types of relations. Among other useful relations are synonymy, hyponymy and hypernymy.

A possible way of improving our approach is to consider dependency between terms. In our current model the resulting translation candidates are considered independently once they have been generated. In fact, other criteria, such as the coherence between the candidates, can also be useful to help select better candidates (Gao and Nie, 2006). We leave it to future work to integrate these criteria into MC models. Currently, the estimation of transition probabilities is made according to the whole collection. It might be more reasonable to estimate them using local contexts related to a given query. This leads to a query-dependent MC model – another area of our future work.

Chapter 6
Selecting Good Expansion Terms for Pseudo-Relevance Feedback

6.1 Introduction

One fact repeatedly mentioned in this thesis is that typical user queries are usually too short to describe the information need accurately. Many important terms can be absent from the query, leading to a poor coverage of the relevant documents. To solve this problem, query expansion has been widely used (Metzler and Croft, 2007; Rocchio, 1971; Xu and Croft, 1996; Zhai and Lafferty, 2001b). Among all the approaches, pseudo-relevance feedback (PRF) exploiting the retrieval result has been the most effective (Xu and Croft, 1996). In general, the expansion terms are extracted from pseudo-feedback documents either according to the term distributions in the feedback documents (i.e. one tries to extract the most frequent terms); or according to the comparison between the term distributions in the feedback documents and in the whole document collection (i.e. to extract the most specific terms in the feedback documents). Several additional criteria have been proposed. For example, *idf* is widely used in vector space model (Rocchio, 1971). Query length has been considered in (Kwok et al., 2000) for the weighting of expansion terms. Some linguistic features have been tested in (Smeaton and Rijsbergen, 1983).

However, few studies have directly examined whether each individual expansion term extracted from pseudo-feedback documents by the existing methods can indeed help

retrieval. In general, one has been concerned only with the global impact of a set of expansion terms on the retrieval effectiveness.

A fundamental question often overlooked at is whether the expansion terms extracted are truly related to the query and are useful for IR. In fact, as we will show in this chapter, the assumption that most expansion terms extracted from the feedback documents are useful does not hold, even when the global retrieval effectiveness can be improved. Among the extracted terms, a non-negligible part is either unrelated to the query or is harmful, instead of helpful, to retrieval effectiveness. So a crucial question is: how can we better select useful expansion terms from pseudo-feedback documents?

In this chapter, we propose to use a supervised learning method for term selection. The term selection problem can be considered as a term classification problem – we try to separate good expansion terms from the others directly according to their potential impact on the retrieval effectiveness. This method is different from the existing ones, which can typically be considered as an unsupervised learning. SVM (Joachims, 1998; Vapnik, 1998; Bishop, 2006) will be used for term classification, which uses not only the term distribution criteria as in previous studies, but also several additional criteria such as term proximity.

This approach proposed has at least the following advantages: 1) Expansion terms are no longer selected solely based on term distributions and other criteria indirectly related to the retrieval effectiveness. It is done directly according to their possible impact on the retrieval effectiveness. We can expect the selected terms to have a higher impact on the effectiveness. 2) The term classification process can naturally integrate various criteria, and thus provides a framework for incorporating different sources of evidence. 3) The further selection of expansion terms can reduce the number of terms added into the query, whereby reducing the time required for query evaluation.

We evaluate our method on three TREC collections and compare it to the traditional approaches. The experimental results show that the retrieval effectiveness can be improved significantly when term classification is integrated. To our knowledge, this is the first

attempt trying to investigate the direct impact on retrieval effectiveness of individual expansion terms in pseudo-relevance feedback.

The remaining of the chapter is organized as follows: Section 6.2 reviews some related work and the state-of-the-art approaches to query expansion. In section 6.3, we examine the PRF assumption used in the previous studies and show that it does not hold in reality. Section 6.4 presents some experiments to investigate the potential usefulness of selecting good terms for expansion. Section 6.5 describes our term classification method and reports an evaluation of the classification process. The integration of the classification results into the PRF methods is described in Section 6.6. In section 6.7, we evaluate the resulting retrieval method with three TREC collections. Section 6.8 concludes this chapter and suggests some avenues for future work.

6.2 Related Work

Pseudo-relevance feedback has been widely used in IR. It has been implemented in different retrieval models: vector space model (Rocchio, 1971), probabilistic model (Robertson and Spark-Jones, 1976), and so on. In the language modeling framework (chapter 4 and 5; Zhai and Lafferty, 2001b), the PRF principle has also been implemented to improve the query model, i.e., θ_q, by exploiting the feedback documents.

As we mentioned before, the query model describes the user's information need. In most traditional approaches using language modeling, this model is estimated with MLE without smoothing. We denote this model by $P(w|\theta_o)$. In general, this query model has a poor coverage of the relevant and useful terms, especially for short queries. Many terms related to the query's topic are absent from (or has a zero probability in) the model. Pseudo-relevance feedback is often used to improve the query model. We have mentioned two representative approaches to exploit pseudo-feedback documents: relevance model and mixture model. Here, we will discuss about them in more detail because our approach is directly related to them.

The relevance model (Lavrenko and Croft, 2001) assumes that a query term is generated by a relevance model $P(w|\theta_R)$. However, it is impossible to define the relevance model without any relevance information. Lavrenko and Croft (2001) thus exploits the top-ranked feedback documents by assuming them to be samples from the relevance model. The relevance model is then estimated as follows:

$$P(w|\theta_R) \approx \sum_{d \in F} P(w|d)P(d|\theta_R) \tag{6.1}$$

where F denotes the feedback documents. On the right side, the relevance model θ_R is approximated by the original query q. Applying Bayesian rule and making some simplifications, we obtain:

$$P(w|\theta_R) \approx \sum_{d \in F} \frac{P(w|d)P(q|d)P(d)}{P(q)} = \sum_{d \in F} P(w|d)P(q|d) \tag{6.2}$$

That is, the probability of a term w in the relevance model is determined by its probability in the feedback documents (i.e. $P(w|d)$) as well as the correspondence of the latter to the query (i.e. $P(Q|D)$). The above relevance model is used to enhance the original query model by the following interpolation:

$$P(w|\theta_q) = (1-\lambda)P(w|\theta_0) + \lambda P(w|\theta_R) \tag{6.3}$$

where λ is the interpolation weight (set at 0.5 in our experiments). Notice that the above interpolation can also be implemented as document re-ranking in practice, in which only the top-ranked documents are re-ranked according to the relevance model.

The mixture model (Zhai and Lafferty, 2001b; section 4.2) also tries to build a language model for the query topic from the feedback documents, but in a way different from the relevance model. It assumes that the query topic model to be extracted corresponds to the part that is the most distinctive from the whole document collection. This distinctive part is extracted as follows: Each feedback document is assumed to be generated by the topic model to be extracted and the collection model, and the EM algorithm (Dempster et al., 1977) is used to extract the topic model so as to maximize the likelihood of the feedback

documents. Then the topic model is combined with the original query model by an interpolation similarly to the relevance model. We denote the topic model as $P(w|\theta_F)$. Some more details have been described in chapter 4.

Although the specific techniques used in the above two approaches are different, both assume that the strong terms contained in the feedback documents are related to the query and are useful to improve the retrieval effectiveness. In both cases, the strong terms are determined according to their distributions. In fact, in relevance model, a term from pseudo-feedback documents is weighted high if it appears frequently in these documents, i.e., $P(t|d)$ is high for a feedback document d. This means that the selected expansion terms are those that are frequent in the feedback documents, or, the selection of expansion terms is based on their distribution among the feedback documents. On the other hand, in the mixture model, one tries to extract the part of feedback model that is the most distinctive from the general model of the collection. This is achieved through the application of the EM algorithm to extract the feedback model. In most other studies about PRF, these criteria have been generally used in other PRF approaches (e.g. (Xu and Croft, 1996)).

In addition to term distributions, several additional criteria have been used to select terms related to the query. Robertson (1990) proposed the principle that the selected terms should have a higher probability in the relevant documents than in the irrelevant documents. This principle is similar to the one used in the mixture model, if we consider that the irrelevance model can be approximated by the whole collection. However, Robertson's approach relies on a more precise identification of relevant and irrelevant documents. This is difficult to implement in practice. Xu and Croft (1996) proposed to use local context information for expansion term selection and produced good results.

For document filtering, term selection is more widely used in order to update the topic profile. For example, (Zhang and Callan, 2001) extracted terms from true relevant and irrelevant documents to update the user profile (i.e. query) using the Rocchio method. Kwok et al. (Kwok et al., 2000) also made use of the query length as well as the size of the vocabulary. Smeaton and Van Rijsbergen (1983) examined the impact of determining expansion terms using minimal spanning tree and some simple linguistic analysis.

Despite the large number of studies on PRF, a crucial question that has not been directly examined is whether the expansion terms selected in a way or another are truly useful for the retrieval. One was usually concerned with the global impact of a set of expansion terms. This is true even additional criteria are used for term selection.

Indeed, in many experiments, improvements in retrieval effectiveness have been observed with PRF (Lavrenko and Croft, 2001; Metzler and Croft, 2007; Tao and Zhai, 2006; Zhai and Lafferty, 2001b). This might suggest that most expansion terms selected from the feedback documents are useful. Is it really so in reality? We will examine this question in the next section.

Notice that some studies have tried to understand the effect of query expansion. For example, Peat and Willett (1991) analyzed the distribution of the expansion terms, and observed that many expansion terms are frequent ones, which have a low capability (or discrimination power) to distinguish relevant documents from irrelevant ones. However, this study has examined the terms extracted from the whole collection according to co-occurrences instead of from the feedback documents. In addition, it also focused on the term distribution aspects.

In the next section, we will examine the usefulness of the expansion terms from feedback documents directly on their impact on retrieval effectiveness.

6.3 A Re-examination of the PRF Assumption

The general assumption behind PRF can be formulated as follows:

> *Most frequent or distinctive terms in pseudo-relevance feedback documents are useful and they can improve the retrieval effectiveness when added into the query.*

To examine this assumption, we will consider all the terms extracted from the feedback documents using the mixture model and examine each of them to see if it contributes in increasing retrieval effectiveness.

Notice that the contribution of one expansion term may be dependent on the other terms in the query. For example, a term may contribute positively to retrieval effectiveness if some other terms are also in the query. However, it is difficult to design a simple test that considers all the term dependencies. In this section, we will perform a simpler test by ignoring the dependency between terms. We assume that a term is useful if it can contribute positively in increasing retrieval effectiveness if it is added into the original query. In this simplified setting, the possible dependency between the expansion term and the other expansion terms is ignored. Nevertheless, such a test can still reveal how the selected expansion terms are useful.

Each of the expansion terms (selected by the mixture model) is added to the original query as follows:

$$Score(d,q) = \sum_{w \in V} P(t \mid \theta_o) \log P(t \mid \theta_d) + w \log P(e \mid \theta_d) \qquad (6.4)$$

where t is a query term, $P(t \mid \theta_o)$ is the original query model as described in section 2, e is the expansion term under consideration, and w is its weight. The above expression is a simplified form of query expansion with a single term. In order to make the test simpler, we further fix the weight of the expansion term at the weight w - the weight w is set at 0.01 or -0.01.

We classify expansion terms into three groups: good, neutral and bad. Good expansion terms are those that improve the effectiveness when w is 0.01 and hurt the effectiveness when w is -0.01; bad expansion terms produce the opposite effect. Neutral expansion terms are those that produce similar effect when w is 0.01 or -0.01. Therefore we can generate three groups of expansion terms: good, bad and neutral. Ideally, we would like to use only good expansion terms to expand queries.

Let us describe the identification of the three groups of terms in more detail. Suppose $MAP(q)$ and $MAP(q \cup e)$ are respectively the MAP of the original query and expanded query (expanded with e). We measure the performance change due to e by the ratio

$chg(e) = [MAP(q \cup e) - MAP(q)]/MAP(q)$. We set a threshold at 0.005 i.e., good and bad expansion terms should produce a performance change such that $|chg(e)|>0.005$.

In addition to the above performance change, we also assume that a term appearing less than 3 times in the feedback documents is not an important expansion term. This allows us to filter out some noise.

Table 19. Proportions of each group of expansion terms selected by the mixture model

Collection	Good Terms	Neutral Terms	Bad Terms
AP	17.52%	47.59%	36.69%
WSJ	17.41%	49.89%	32.69%
Disk4&5	17.64%	56.46%	25.88%

The above identification produces three lists of expansion terms according to their usefulness to retrieval. Now, we will examine the query expansion hypothesis to see whether (most of) the candidate expansion terms proposed by the mixture model are good terms. Our verification is made on three TREC collections: AP, WSJ and Disk4&5. The characteristics of these collections are described in Section 6.7.1. We consider 150 queries for each collection and 80 expansions with the largest probabilities for each query. Table 20 shows the proportion of good, bad and neutral terms for all the queries in each collection.

As we can see, only less than 18% of the expansion terms used in the mixture model are good terms in all the three collections. The proportion of bad terms is higher. This shows that the expansion process indeed added more bad terms than good ones.

We also notice from Table 20 that a large proportion of the expansion terms are neutral terms, which have little impact on the retrieval effectiveness. Although this part of the terms does necessarily not hurt retrieval, adding them into the query would produce a long query and thus heavier query traffic (longer evaluation time). It is then desirable to remove these terms, too.

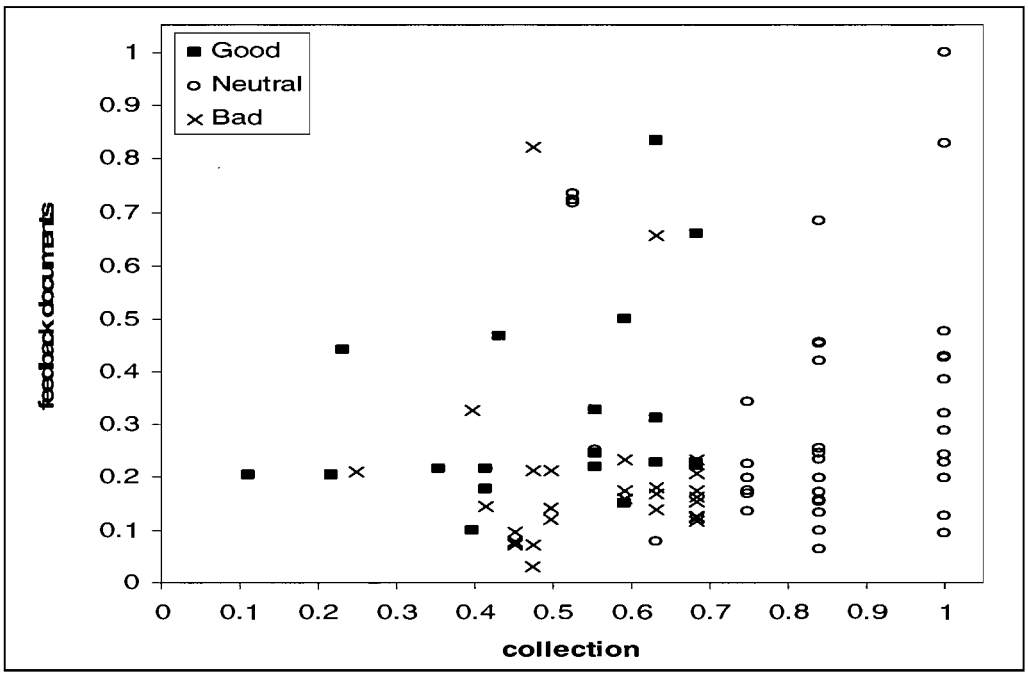

Figure 7. Distribution of the expansion terms for "airbus subsidies" in the feedback documents and in the collection

The above analysis clearly shows that the term selection process used in the mixture model is insufficient. Similar phenomenon is observed on the relevance model and can be generalized to all the methods exploiting the same criteria. This suggests that the term selection criteria used - term distributions in the feedback documents and in the whole document collection, is insufficient. This also indicates that good and bad expansion terms may have similar distributions because the mixture model, which exploits the difference of term distribution between the feedback documents and the collection, has failed to distinguish them.

To illustrate the last point, let us look at the distribution of the expansion terms selected with the mixture model for TREC query #51 "airbus subsidies". In Figure 7, we place the top 80 expansion terms with the largest probabilities in a two-dimensional space – one dimension represents the logarithm of its probability in the pseudo-relevant documents and another dimension represents that in the whole collection. To make the illustration easier, a simple normalization is made so that the final value will be in the range [0, 1]. Figure 7 shows the distribution of the three groups of expansion terms. We can observe that the

neutral terms are somehow isolated from the good and the bad terms to some extent (on the lower-right corner), but the good expansion terms are intertwined with the bad expansion terms.

This figure illustrates the difficulty to separate good and bad expansion terms according to term distributions solely. It is then desirable to use additional criteria to better select useful expansion terms.

6.4 Usefulness of Selecting Good Terms

Table 20. The impact of oracle expansion classifier

Models	AP	WSJ	Disk4&5
LM	0.2407	0.2644	0.1753
REL	0.2752^{L}	0.2843^{L}	0.1860^{L}
REL+Oracle	$\mathbf{0.3402}^{R,L}$	$\mathbf{0.3518}^{R,L}$	$\mathbf{0.2434}^{R,L}$
MIX	0.2846^{L}	0.2938^{L}	0.2005^{L}
MIX+Oracle	$\mathbf{0.3390}^{M,L}$	$\mathbf{0.3490}^{M,L}$	$\mathbf{0.2418}^{M,L}$

Before proposing an approach to select good terms, let us first examine the possible impact with a good term selection process. Let us assume an oracle classifier that separate correctly good, bad and neutral expansion terms as determined in Section 6.3.

In this experiment, we will only keep the good expansion terms for each query. All the good terms are integrated into the new query model in the same way as either relevance model or mixture model. Table 21 shows the MAP (Mean Average Precision) for the top 1000 results with the original query model (**LM**), the expanded query models by the relevance model (**REL**) and by the mixture model (**MIX**), as well as by the oracle expansion terms (**REL+Oracle** and **MIX+Oracle**). The superscript, "*L*", "*R*" and "*M*" indicates that the improvement over LM, REL and MIX is statistically significant at $p<0.05$.

We can see that the retrieval effectiveness can be much improved if term classification is done perfectly. The oracle expansion terms can generally improve the MAP of the relevance model and the mixture model by 18-30%. This shows the usefulness of correctly classifying the expansion terms and the high potential of improving the retrieval

effectiveness by a good term classification. The MAP obtained with the oracle expansion terms represents the upper bound retrieval effectiveness we can expect to obtain using pseudo-relevance feedback. Our problem now is to develop an effective method to correctly classify the expansion terms.

6.5 Classification of Expansion Terms

6.5.1 SVM Classifier

Any classifier can be used for term classification. Here, we use SVM. More specifically, we use the SVM (Bishop, 2006) because of its effectiveness and simplicity (Vapnik, 1998; Cristianini and Shawe-Taylor, 2001).

Support Vector Machines (SVM) became popular some years ago for solving problems in classification, regression, and information filtering and novelty detection. As a Max-Margin classifier, SVM has several beautiful properties. 1) SVM has solid theoretic basis. It is based on the *Structural Risk Minimization* principle (Vapnik, 1998; Burges, 1998) from statistical learning theory. The idea of structural risk minimization is to find a hypothesis h (i.e., the classifier) which can guarantee the lowest generalization error. The generalization error is the error when the hypothesis is tested with an unseen and randomly selected sample. The SVM has a very nice property that the upper bound of generalization error is tightly related to the margin. Therefore, maximizing the margin can minimize the generalization error (Joachims, 1998; Burges, 1998). 2) The training of SVM classifier, or the parameter estimation, is formulized as a convex optimization problem, which we will see in the later of this section. Therefore, it has a unique optimal solution. 3) With kernel functions, SVM can map non-separable instances in low dimensional space into separable instances in higher dimensional space (possibly infinite dimensional space).

Since our goal is to determine whether an expansion term is good, we adopt the binary classifier, in which the positive instances are good expansion term while the negative instances are either bad or neutral terms. Formally, suppose there are a set of training instances $< x_i, y_i > \in I$, where $y_i \in \{-1,+1\}$ denotes the classification label and $x_i \in \Re^n$ denotes a vector in the feature space, the SVM is a hyperplane to separate the instances. The hyperplane can be described as $W^T x + b = 0$, where W^T is the transpose of the normal vector of the hyperplane. This can be illustrated with figure 6.2.

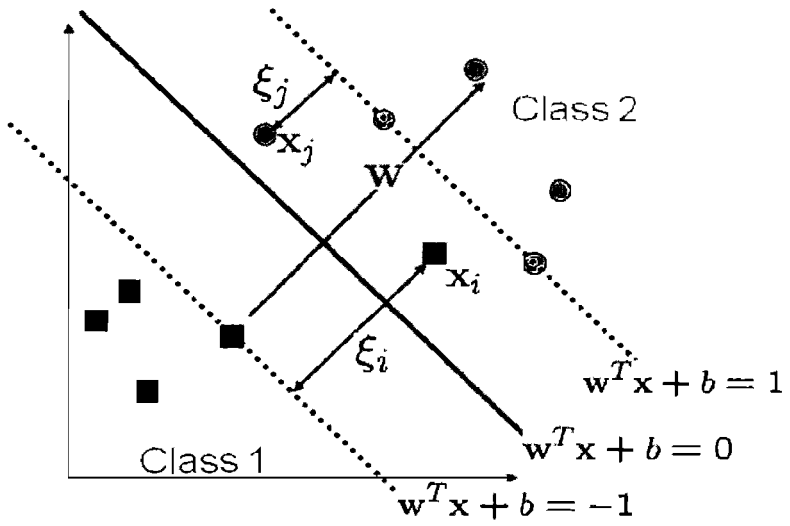

Figure 8. Binary SVM with Non-separable Instances

In figure 8, the square points denotes the instances belonging to class 1 while the round points represent the instances belonging to class 2. There are also three hyperplanes in the figure, i.e., $W^T x + b = 0$ and ± 1. The first is called the decision boundary. The hyperplane, $W^T x + b = -1$, governs the negative instances (class 1) and the hyperplane, $W^T x + b = +1$, governs the positive instances (class 2). In the linear separable case, all instances are outside the two hyperplanes. However, in the non-separable case, some of the instances fall within the two hyperplanes, so we have to introduce the slack variable ξ_i. This variable is defined as the distance of the instance to its corresponding hyperplane. For example, in figure 8, ξ_i is the distance of x_i to the hyperplane $W^T x + b = -1$, while ξ_j is the distance of x_j to the hyperplane $W^T x + b = +1$.

Therefore, the slack variables measure how well the SVM separates the training instances. The perpendicular distance between other two hyperplanes is defined as the margin of the SVM. Maximizing the margin results in minimum generalization error. Following this principal, training SVM can be formulized as:

$$\begin{aligned} &\min_{W,b,\xi} \frac{1}{2} W^T W + C\sum_i \xi_i \\ &Subject\ To: \\ &\qquad W^T x_i + b \geq 1 - \xi_i \\ &\qquad \xi_i \geq 0 \end{aligned} \tag{6.5}$$

In fact, the objective function has two parts. The right part can be viewed as the training error, and the left part is proportional to the inverse of the classifier margin, so it corresponds to the generalization error. Therefore, C is the balance factor between the training and generalization factor.

The optimal solution can be found by many approaches. One of the most effective methods is the SMO algorithm (Platt, 1998). After obtaining the optimal solution, the incoming instance can be classified with the decision boundary, i.e.:

$$y = sign(W^T x + b) \tag{6.6}$$

One important advantage of SVM is the usage of kernel function. With kernel function, it is able to map non linearly separable instances in the lower dimensional space into linearly separable instances in a higher dimensional space. The effect of kernel function can be illustrated with figure 9. In this figure, the left side instances are not linearly separable, with kernel function ($\varphi(x)$), the instances are mapped into a higher dimensional space, in which they are linearly separable.

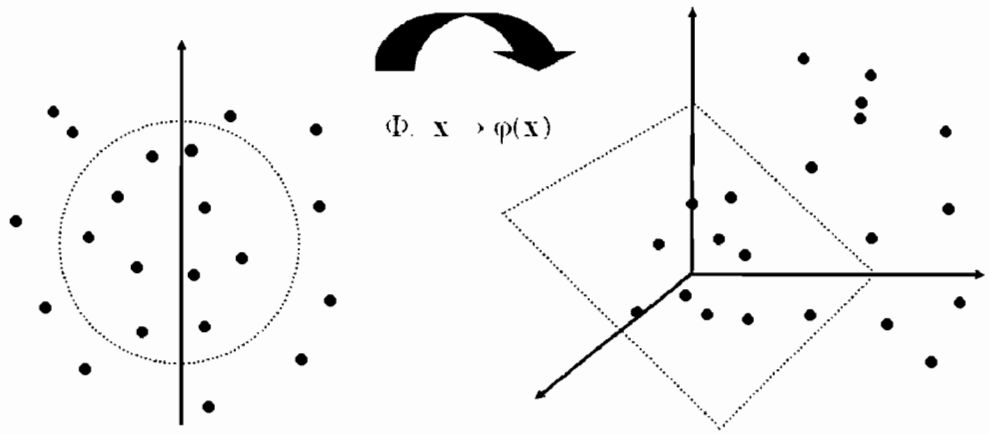

Figure 9. The Effect of Kernel Function

Several kernel functions can be used in SVM (Burges, 1998; Bishop, 2006). We use the radial-based kernel function (RBF) because it has relatively fewer hyper parameters and has shown to be effective in previous studies (Bishop, 2006; Hsu et al., 2007). This function is defined as follows:

$$K(x_i, x_j) = \exp\left(-\| x_i - x_j \|/2\sigma^2\right) \tag{6.7}$$

where σ is a parameter controlling the shape of the RBF function. The function gets flatter when σ is larger. Another hyper parameter is C in equation 6.5. Both parameters are estimated with a 5-fold cross-validation to maximize the classification accuracy of the training data.

In our term classification, we are interested to know not only if a term is good, but also the extent to which it is good. This latter value is useful for us to measure the importance of an expansion term and to weight it in the new query. Therefore, once we obtain a classification score, we use the method described in (Platt, 2000) to transform it into a posterior probability: Let *s(x)* denote the classification score calculated by equation 6.6. Then the probability of x belonging to the class of good terms (denoted by **+1**) is defined by:

$$P(+1 \mid x) = \frac{1}{1+\exp(As(x)+B)} \tag{6.8}$$

where A and B are the parameters, which are estimated by minimizing the cross-entropy of a portion of training data, namely the development data. This process has been automated in LIBSVM (Hsu et al., 2007). We will have $P(+1|x)>0.5$ if and only if the term x is classified as a good term. More details about this model can be found in (Platt, 2000). Note that the above probabilistic SVM may have different classification results from the simple SVM, which classifies instances according to equation 6.6. In our experiments, we have tested both probabilistic and simple SVMs, and found that the former performs better. We use the SVM implementation LIBSVM (Hsu et al., 2007) in our experiments.

6.5.2 Features Used for Term Classification

Each expansion term is represented by a feature vector $F(e)=[f_1(e), f_1(e),..., f_N(e)]^T \in \Re^N$, where T means a transpose of a vector. Useful features include those already used in traditional approaches such as term distribution in the feedback documents and term distribution in the whole collection. As we mentioned, these features are insufficient. Therefore, we consider the following additional features:

- co-occurrences of the expansion term with the original query terms;
- proximity of the expansion terms to the query terms.

We will explain several groups of features below. Our assumption is that the most useful feature for term selection is the one that makes the largest difference between the feedback documents and the whole collection (similar to the principle used in the mixture model). So, we will define two sets of features, one for the feedback documents and another for the whole collection. However, technically, both sets of features can be obtained in a similar way. Therefore, we will only describe the features for the feedback documents. The others can be defined similarly.

- **Term distributions**

The first features are the term distributions in the pseudo-relevant documents and in the collection. The feature for the feedback documents is defined as follows:

$$f_1(e) = \log \frac{\sum_{d \in F} tf(e,d)}{\sum_t \sum_{d \in F} tf(t,d)}$$

where F is the set of feedback documents. $f_2(e)$ is defined similarly on the whole collection. These features are the traditional ones used in the relevance model and mixture model.

- **Co-occurrence with single query term**

Many studies have found that the terms that co-occur with the query terms frequently are often related to the query (Bai et al., 2007; Cao et al., 2006). Therefore, we define the following feature to capture this fact:

$$f_3(e) = \log \frac{1}{n} \sum_{i=1}^{n} \frac{\sum_{D \in F} C(t_i, e \mid d)}{\sum_t \sum_{D \in F} tf(t,d)}$$

where $C(t_i,e|d)$ is the frequency of co-occurrences of query term t_i and the expansion term e within text windows in document d. The window size is empirically set to be 12 words.

- **Co-occurrence with pairs query terms**

A stronger co-occurrence relation for an expansion term is with two query terms together. Bai et al. (2007) has shown that this type of co-occurrence relation is much better than the previous one because it can take into account some query contexts. The text window size used here is 15 words. Given the set Ω of possible term pairs, we define the following feature, which is slightly extended from the previous one:

$$f_5(e) = \log \frac{1}{|\Omega|} \sum_{(t_i,t_j) \in \Omega} \frac{\sum_{D \in F} C(t_i, t_j, e \mid d)}{\sum_t \sum_{D \in F} tf(t,d)}$$

- **Weighted term proximity**

The idea of using term proximity has been used in several studies (Tao and Zhai, 2007). Here we also assume that two terms that co-occur at a smaller distance is more closely related. There are several ways to define the distance between two terms in a set of

documents (Tao and Zhai, 2007). Here, we define it as the minimum number of words between the two terms among all co-occurrences in the documents. Let us denote this distance between t_i and t_j among the set B of documents by $dist(t_i,t_j|B)$. For a query of multiple words, we have to aggregate the distances between the expansion term and all query terms. The simplest method is to consider the average distance, which is similar to the average distance defined in (Tao and Zhai, 2007). However, it does not produce good results in our experiments. Instead, the weighted average distance works better. In the latter, a distance is weighted by the frequency of their co-occurrences. We then have the following feature:

$$f_7(e) = \log \frac{\sum_{i=1}^{n} C(t_i, e) dist(t_i, e \mid F)}{\sum_{i=1}^{n} C(t_i, e)}$$

where $C(t_i, e)$ is the frequency of co-occurrences of t_i and e within text windows in the collection. The window size is set to 12 words as before.

- **Document frequency for query terms and the expansion term together**

The features in this group model the count of documents in which the expansion term co-occurs with all query terms. We then have:

$$f_9 = \log\left[\sum_{D \in F} I\left(\left(\wedge_{t \in q}\, t \in d\right) \wedge e \in d\right) + 0.5\right]$$

where $I(x)$ is the indicator function whose value is 1 when the Boolean expression x is true, and 0 otherwise. The constant 0.5 here acts as a smoothing factor to avoid zero value.

To avoid that a feature whose values varies in a larger numeric range dominates those varying in smaller numeric ranges, scaling on feature values is necessary (Hsu et al., 2007). The scaling is done in a query-by-query manner. Let $e \in GEN(q)$ be an expansion term of the query q, and $f_i(e)$ is one feature value of e. We scale $f_i(e)$ as follows:

$f_i'(e) = (f_i(e) - \min_i)/(\max_i - \min_i)$ where $min_i = min_{e \in GEN(q)} f_i(e)$ and $max_i = max_{e \in GEN(q)} f_i(e)$. With this transformation, each feature becomes a real number in [0, 1].

In our experiments, only the above features are used. However, the general method is not limited to them. Other features can be added. The possibility to integrate arbitrary features for the selection of expansion terms indeed represents an advantage of our method.

6.5.3 Classification Experiments

Table 21. Classification results of SVM

Coll.	Percentage of good terms	SVM		
		Accuracy	Rec.	Prec.
AP	0.3356	0.6945	0.3245	0.6300
WSJ	0.3126	0.6964	0.3749	0.5700
Disk4&5	0.3270	0.6901	0.3035	0.5970

Let us now examine the quality of our classification. We use three test collections (see Table 23), with 150 queries for each collection. We divide these queries into three groups of 50 queries. We then do leave-one-out cross validation to evaluate the classification accuracy. The gold standard for classification is the classification result we obtained in Section 6.3. To generate training and test data, we use the method described in section 6.3 to label possible expansion terms of each query as good terms or non-good terms (including bad and neutral terms), and then represent each expansion with the features described in section 6.5.2. The candidate expansion terms are those that occur in the feedback documents (top 20 documents in the initial retrieval) no less than three times.

Table 22 shows the classification results. In this table, we show the percentage of good expansion terms for all the queries in each collection – around 1/3. Using the SVM classifier, we obtain a classification accuracy of about 69%. This number is not high. In fact, if we use a naïve classifier that always classifies instances into non good class, the accuracy (i.e. one minuses the percentage of good terms) is only slightly lower. However, such a classifier is useless for our purpose because no expansion term is classified as good term. Better indicators are recall, and more particularly precision. Although the classifier only identifies about 1/3 of the good terms (i.e. recall), around 60% of the identified ones are truly good terms (i.e. precision). Comparing to Table 20 for the expansion terms selected by the mixture model, we can see that the expansion terms select by the SVM

classifier are of much higher quality. This shows that the additional features we considered in the classification are useful, although they could be further improved in the future.

In the next section, we will describe how the selected expansion terms are integrated into our retrieval model.

6.6 Re-weighting Expansion Terms with Term Classification

The classification process performs a further selection of expansion terms among those proposed by the relevance model and the mixture model respectively. The selected terms can be integrated in these models in two different ways: hard filtering, i.e. we only keep the expansion terms classified as good; or soft filtering, i.e. we use the classification score to enhance the weight of good terms in the final query model. Our experiments show that the second method performs better. We will make a comparison between these two methods in Section 6.7.4. In this section, we focus on the second method, which means a redefinition of the models $P(w|\theta_R)$ for the relevance model and $P(w|\theta_F)$ for the mixture model. These models are redefined as follows: For a term e such that $P(+1|e)>0.5$,

$$P(w|\theta_R)^{new} = \left(P(e|\theta_R)^{old}\left(1+\alpha P(+1|e)\right)\right)/Z$$

$$P(w|\theta_F)^{new} = \left(P(e|\theta_F)^{old}\left(1+\alpha P(+1|e)\right)\right)/Z \qquad (6.9)$$

where Z is the normalization factor, and α is a coefficient, which is estimated with some development data in our experiments using line search (Gao et al., 2005), which tries to find a better value in turn until no improvement can be achieved. Once the expansion terms are re-weighted, we will retain the top 80 terms with the highest probabilities for expansion. Their weights are normalized before being interpolated with the original query model. The number 80 is used for a fair comparison with the relevance model and the mixture model.

6.7 IR Experiments

6.7.1 Experimental Settings

Table 22. Statistics of evaluation data sets

Name	Description	#Docs	Train Topics	Dev. Topics	Test topics
AP	Assoc. Press 88-90	24,918	101-150	151-200	51-100
WSJ	Wall St. Journal 87-92	173,252	101-150	151-200	51-100
Disk4&5	TREC disk4&5	556,077	301-350	401-450	351-400

We evaluate our method with three TREC collections, AP88-90, WSJ87-92 and all documents on TREC disks 4&5. Table 23 shows the statistics of the three collections. For each dataset, we split the available topics into three parts: the training data to train the SVM classifier, the development data to estimate the parameter α in equation 6.10, and the test data. We only use the title for each TREC topic as our query. Both documents and queries are stemmed with Porter stemmer and stop words are removed.

The main evaluation metric is Mean Average Precision (MAP) for top 1000 documents. Since some previous studies showed that PRF improves recall but may hurt precision, we also show the precision at top 30 and 100 documents, i.e., P@30 and P@100. We also show recall as a supplementary measure. We do a query-by-query analysis and conduct t-test to determine whether the improvement on MAP is statistically significant.

The Indri 2.6 search engine (Strohman et al., 2004) is used as our basic retrieval system. We use the relevance model implemented in Indri, but implemented the mixture model following (Zhai and Lafferty, 2001b) since Indri does not implement this model.

6.7.2 Ad-hoc Retrieval Results

In the experiments, the following methods are compared:

Table 23. Ad-hoc retrieval results on AP data

Model	P@30	P@100	MAP	Imp	Recall
LM	0.3967	0.3156	0.2407	-----	0.4389
REL	0.4380	0.3628	0.2752	14.33%**	0.4932
REL+SVM	*0.4513*	*0.3680*	*0.2959*^{R}	*22.93%***	*0.5042*
MIX	0.4493	0.3676	0.2846	18.24%**	0.5163
MIX+SVM	*0.4567*	*0.3784*	*0.3090*^{M,R}	*28.36%***	*0.5275*

Table 24. Ad-hoc retrieval results on WSJ data

Model	P@30	P@100	MAP	Imp	Recall
LM	0.3900	0.2936	0.2644	--------	0.6516
REL	0.4087	0.3078	0.2843	7.53%**	0.6797
REL+SVM	*0.4167*	*0.3120*	*0.2943*	*11.30%***	*0.6933*
MIX	0.4147	0.3144	0.2938	11.11%**	0.7052
MIX+SVM	*0.4200*	*0.3160*	*0.3036*^{R}	*14.82%***	*0.7110*

Table 25. Ad-hoc retrieval results on Disk4&5 data

Model	P@30	P@100	MAP	Imp	Recall
LM	0.2900	0.1734	0.1753	----------	0.4857
REL	0.2973	0.1844	0.1860	6.10%*	0.5158
REL+SVM	*0.2833*	*0.1990*	*0.2002*^{R}	*14.20%***	*0.5689*
MIX	0.3027	0.1998	0.2005	14.37%**	0.5526
MIX+SVM	*0.3053*	*0.2068*	*0.2208*^{M,R}	*25.96%***	*0.6025*

LM: the KL-divergence retrieval model with the original queries;

REL: the relevance model;

REL+SVM: the relevance model with term classification;

MIX: the mixture model;

MIX+SVM: the mixture model with term classification.

These models require some parameters, such as the weight of original model when forming the final query representation, the Dirichlet prior for document model smoothing and so on. Since the purpose of this paper is not to optimize these parameters, we set all of them at the same values for all the models. Tables 24, 25 and 26 show the results obtained with different models on the three collections. In the tables, **imp** means the improvement rate over LM model, * indicates that the improvement is statistically significant at the level of

$p<0.05$, and ** at $p<0.01$. The superscripts "R" and "M" indicate that the result is statistically better than the relevance model and mixture model respectively at $p<0.05$.

Table 26. Expansion terms of two queries. The terms in italic are real good expansion terms, and those in bold are classified as good terms

"machine translation"			
Expansion terms	$P(t_i \mid \theta_F)$	Expansion terms	$P(t_i \mid \theta_F)$
compute	**0.0162**	year	0.0043
soviet	**0.0095**	work	0.0038
company	**0.0082**	make	0.0040
50	0.0074	***typewriter***	**0.0038**
english	**0.0072**	*busy*	0.0021
ibm	**0.0051**	increase	0.0021
people	0.0050	.....	....
"natural language processing"			
Expansion terms	$P(t_i \mid \theta_F)$	Expansion terms	$P(t_i \mid \theta_F)$
english	**0.0132**	publish	0.0041
word	**0.0092**	*nation*	0.0040
french	**0.0092**	**develop**	**0.0039**
food	0.0064	**russian**	**0.0038**
make	0.0050	***program***	**0.0037**
world	0.0047	***dictionary***	**0.0012**
gorilla	**0.0045**	.....	.....

From the tables, we observe that both relevance model and mixture model, which exploit a form of PRF, can improve the retrieval effectiveness of LM significantly. This observation is consistent with previous studies. The MAP we obtained with these two models represent the state-of-the-art effectiveness on these test collections.

Comparing the relevance model and the mixture model, we see that the latter performs better. The reason may be the following: The mixture model relies more on the difference between the feedback documents and the whole collection to select the expansion terms, than the relevance model. By doing this, one can filter out more bad or neutral expansion terms.

On all the three collections, the model integrating term classification performs very well. When the classification model is used together with a PRF model, the effectiveness is always improved. On the AP and Disk4&5 collections, the improvements are more than

7.5% and are statistically significant. The improvements on the WSJ collection are smaller (about 3.5%) and are not statistically significant.

About the impact on precision, we can also see that term classification can also improve the precision at top ranked documents, except in the case of Disk4&5 when SVM is added to REL. This shows that in most cases, adding the expansion terms does not hurt, but improves, precision.

Let us show the expansion terms for the queries "machine translation" and "natural language processing", in Table 27. The stemmed words have been restored in this table for better readability. All the terms contained in the table are those suggested by the mixture model. However, only part of them (in *italic*) is useful expansion terms. Many of them are general terms that are not useful, for example, "food", "make", "year", "50", and so on.

The classification process can help identify well the useful expansion terms (in **bold**): although not all the useful expansion terms are identified, those identified (e.g. "program", "dictionary") are highly related and useful. As the weight of these terms is increased, the relative weight of the other terms is decreased, making their weights in the final query model smaller. These examples illustrate why the term classification process can improve the retrieval effectiveness.

6.7.3 Supervised vs. Unsupervised Learning

Compared to the relevance model and the mixture model, the approach with term classification made two changes: it uses supervised learning instead of unsupervised learning; it uses several additional features. It is then important to see which of these changes contributed the most to the increase in retrieval effectiveness.

In order to see this, we design a method using unsupervised learning, but with the same additional features. The unsupervised learning extends the mixture model in the following way:

Each feedback document is also considered to be generated from the topic model (to be extracted) and the collection model. We try to extract the topic model so as to maximize the likelihood of the feedback documents as in the mixture model. However, the difference is that, instead of defining the topic model $P(w \mid \theta_F)$ as a multinomial model, we define it as a log-linear model that combines all the features:

$$P(w \mid \theta_F) = \exp\left(\lambda^T F(w)\right) / Z \tag{6.10}$$

where *F(w)* is the feature vector defined in section 6.5.2, λ is the weight vector and *Z* is the normalization factor to make $P(w \mid \theta_T)$ a real probability. λ is estimated by maximizing the likelihood of the feedback documents. To avoid overfitting, we do regularization on λ by assuming that it has a zero-mean Gaussian prior distribution (Bishop, 2006). Then the objective function to be maximized becomes:

$$L(F) = \sum_{D \in F} \sum_{w \in V} tf(w,D) \log\left((1-\alpha)P(w \mid \theta_C) + \alpha P(w \mid \theta_F)\right) - \partial \lambda^T \lambda \tag{6.11}$$

where ∂ is the regularization factor, which is set to be 0.01 in our experiments. α is the parameter representing how likely we use the topic model to generate the pseudo-relevant document. It is set at a fixed value as in (Zhai and Lafferty, 2001b) (0.5 in our case). Since *L(F)* is a concave function w.r.t. λ, it has a unique maximum. We solve this unconstrained optimization problem with Limited memory *BFGS* (*L-BFGS*) algorithm (Nocedal and Wright, 2006).

Table 28 shows the results measured by MAP. Again, the superscript, "M" and "L" indicate the improvement over MIX and Log-linear model is statistically significant at $p<0.05$.

From this table, we can observe that the log-linear model outperforms the mixture model only slightly. This shows that an unsupervised learning method, even with additional features, cannot improve the retrieval effectiveness by a large margin. The possible reason

is that the objective function, *L(F)*, does not correlate very well with MAP. The parameters maximizing *L(F)* do not necessarily produce good MAP.

In comparison, the MIX+SVM model outperforms largely the log-linear model on all the three collections, and the improvements on AP and Disk4&5 are statistically significant. This result shows that a supervised learning method can more effectively capture the characteristics of the genuine good expansion terms than an unsupervised method.

Table 27. Supervised Learning VS Unsupervised Learning

Model	AP	WSJ	Disk4&5
MIX	0.2846	0.2938	0.2005
Log-linear	0.2878	0.2964	0.2020
MIX+SVM	$0.3090^{M,L}$	0.3036	$0.2208^{M,L}$

6.7.4 Soft Filtering vs. Hard Filtering

We mentioned two possible ways to use the classification results: hard filtering of expansion terms by retaining only the good terms, or soft filtering by increasing the weight of the good terms. In this section, we compare the two methods. Table 29 shows the results obtained with both methods. In the table, "M", "R", and "H" indicate the improvement over MIX, REL and HARD are statistically significant with $p<0.05$

From this table, we see that both hard and soft filtering improves the effectiveness. Although the improvements with hard filtering are smaller, they are steady on all the three collections. However, only the improvement over MIX model on the AP and Disk4&5 data is statistically significant.

In comparison, the soft filtering method performs much better. Our explanation is that, since the classification accuracy is far from perfect (actually, it is less than 70% as shown in Table 22), some top ranked good expansion terms, which could improve the performance significantly, can be removed by the hard filtering. On the other hand, in the soft filtering case, even if the top ranked good terms are misclassified, we will only reduce

their relative weight in the final query model rather than removing them. Therefore, these expansion terms can still contribute to improving the performance. In other words, the soft filtering method is less affected by classification errors.

Table 28. Soft Filtering VS Hard Filtering

Model	AP	WSJ	Disk4&5
MIX	0.2846	0.2938	0.2005
MIX+HARD	0.2902^{M}	0.2989	0.2024^{M}
MIX+SOFT	$0.3090^{M,H}$	0.3036	$0.2208^{M,H}$
REL	0.2752	0.2843	0.1860
REL+HARD	0.2804	0.2864	0.1890
REL+SOFT	$0.2959^{R,H}$	0.2943	0.2002^{R}

6.7.5 Reducing Query Traffic

Table 29. Soft filtering with 10 terms

Model	AP	WSJ	Disk4&5
MIX-80	0.2846	0.2938	0.2005
MIX-10	0.2824	0.2913	0.2015
MIX+SOFT-10	*0.2932*	*0.2915*	*0.2125*

A critical aspect with query expansion is that, as more terms are added into the query, the query traffic, i.e. the time needed for its evaluation, becomes larger. In the previous sections, for the purpose of comparison with previous methods, we used 80 expansion terms. In practice, this number is too large – one cannot afford to increase the size of a query so drastically. In this section, we examine the possibility to further reduce the number of expansion terms.

In this experiment, after a re-weighting with soft filtering, instead of keeping 80 expansion terms, we only select the top 10 expansion terms, which is a more reasonable number. These terms are used to construct a small query topic model $P(w \mid \theta_F)$. This model is interpolated with the original query model as before. The following table describes the results using the mixture model.

As expected, the effectiveness with 10 expansion terms is lower than with 80 terms. However, we can still obtain much higher effectiveness than the traditional language model LM, and all the improvements are significantly significant.

The results with 10 expansion terms can also be advantageously compared to the mixture model with 80 expansion terms: for both AP and Disk4&5 collections, the effectiveness is higher than the mixture model. The effectiveness on WSJ is very close.

This experiment shows that we can reduce the number of expansion terms, and even with a reasonably small number, the retrieval effectiveness can be greatly increased. This observation allows us to control query traffic within an acceptable range, and make the method more feasible in the search engines.

6.8 Summary of This Chapter

Pseudo-relevance feedback, which adds additional terms extracted from the feedback documents, is an effective method to improve the query representation and the retrieval effectiveness. The basic assumption is that most strong terms in the feedback documents are useful for IR. In this study, we re-examined this hypothesis on three test collections and showed that the expansion terms determined in traditional ways are not all useful. In reality, only a small proportion of the suggested expansion terms are useful, and many others are either harmful or useless. In addition, we also showed that the traditional criteria for the selection of expansion terms based on term distributions are insufficient: good and bad expansion terms are not distinguishable on these distributions.

Motivated by these observations, we proposed to further classify expansion terms using additional features based on term relationships. In addition, we aim to select the expansion terms directly according to their possible impact on the retrieval effectiveness. This method is different from the existing ones, which often rely on some other criteria that do not always correlate with the retrieval effectiveness.

Our experiments on three TREC collections showed that the expansion terms selected using our method are significantly better than the traditional expansion terms. In addition, we also showed that it is possible to limit the query traffic by controlling the number of expansion terms, and this still lead to quite large improvements in retrieval effectiveness.

This study shows the importance to examine the crucial problem of usefulness of expansion terms before the terms are used. The method we propose also provides a general framework to integrate multiple sources of evidence.

This study suggests several interesting research avenues for our future investigation: The results we obtained with term classification are much lower than with the oracle expansion terms. This means that there is still much room for improvement. In particular, improvement in classification quality could directly result in improvement in retrieval effectiveness. The improvement of classification quality could be obtained by integrating more useful features. In this chapter, we have limited our investigation to only a few often used features. More discriminative features can be investigated in the future. In particular, in chapter 4, we found that semantic relationships between terms are also useful for query expansion, so we can consider them in the future.

The basic idea, i.e., selecting an expansion term according the expected improvement, can be used in other tasks, such as query suggestions, query alteration and query term stemming (Peng et al., 2007). In next section, we will apply the same principle to another task - context sensitive query term stemming, which can also be viewed as a special form of query expansion. We will see that the approach is also effective in that task.

Chapter 7
Exploiting Word Relations for Context Sensitive Stemming

7.1 Introduction

Word stemming is a basic NLP technique used in most of Information Retrieval (IR) systems. It transforms words into their root forms so as to increase the chance to match similar words/terms that are morphological variants. For example, with stemming, "controlling" can match "controlled" because both have the same root "control". Most stemmers, such as the Porter stemmer (Porter, 1980) and Krovetz stemmer (Krovetz, 1993), deal with stemming by stripping word suffixes according to a set of morphological rules. The rule-based approaches are intuitive and easy to implement. However, while in general, most words can be stemmed correctly; there is often erroneous stemming that unifies unrelated words. For instance, "jobs" is stemmed to "job" in both "find jobs in Apple" and "Steve Jobs at Apple". This is particularly problematic in Web search, where users often use special or new words in their queries. A standard stemmer such as Porter's will stem them uniformly regardless to the context of utilization.

To better determine stemming rules, Xu and Croft (1998) propose a selective stemming method based on corpus analysis. They refine the Porter stemmer by means of word clustering: words are first clustered according to their co-occurrences in the text collection. Only word variants belonging to the same cluster will be conflated.

Despite this improvement, the basic idea of word stemming is to transform words in both documents and queries to a standard form. Once this is done, there is no means for users to require a specific word form in a query – the word form will be automatically transformed, otherwise, it will not match documents. This approach does not seem to be appropriate for Web search, where users often specify particular word forms in their queries. An example of this is a quoted query such as "Steve Jobs", or "US Policy". If documents are stemmed, many pages about job offerings or US police may be returned ("policy" conflates with "police" in Porter stemmer). Another drawback of stemming is that it usually enhances recall, but may hurt precision (Kraaij and Pohlmann, 1996). However, general Web search is basically a precision-oriented task.

One alternative approach to word stemming is to do query expansion at query time. The original query terms are expanded by their related forms having the same root. All expansions can be combined by the Boolean operator "*OR*". For example, the query "*controlling acid rain*" can be expanded to "*(control OR controlling OR controller OR controlled OR controls) (acid OR acidic OR acidify) (rain OR raining OR rained OR rains)*". We will call each such expansion term an *alteration* to the original query term. Once a set of possible alterations is determined, the simplest approach to perform expansion is to add all possible alterations. We call this approach ***Naive Expansion***. One can easily show that stemming at indexing time is equivalent to *Naive Expansion* at retrieval time. This approach has been adopted by most commercial search engines (Peng et al., 2007). However, the expansion approaches proposed previously can have several serious problems: First, they usually do not consider expansion ambiguity – each query term is usually expanded independently. However, some expansion terms may not be appropriate. The case of "Steve Jobs" is one such example, for which the word "job" can be proposed as an expansion term. Second, as each query term may have several alterations, the naïve approach using all the alterations will create a very long query. As a consequence, query traffic (the time required for the evaluation of a query) is greatly increased. Query traffic is a critical problem, as each search engine serves millions of users at the same time. It is important to limit the query traffic as much as possible.

In practice, we can observe that some word alterations are irrelevant and undesirable (as in the "Steve Jobs" case), and some other alterations have little impact on the retrieval effectiveness (for example, if we expand a word by a rarely used word form). In this chapter, we will address these two problems. Our goal is to select only appropriate word alterations to be used in query expansion. This is done for two purposes: On the one hand, we want to limit query traffic as much as possible when query expansion is performed. On the other hand, we also want to remove irrelevant expansion terms so that fewer irrelevant documents will be retrieved, thereby improving the retrieval effectiveness.

To deal with the two problems we mentioned above, we will propose two methods to select alterations. In the first method, we make use of the query context to select only the alterations that fit the query. In the second method, we try to predict the usefulness of an alteration and the selection of alterations is made accordingly.

In the first method, the query context is modeled by a bigram language model. To reduce query traffic, we select only one alteration for each query term, which is the most coherent with the bigram model. We call this model ***Bigram Expansion***. Despite the fact that this method adds far fewer expansion terms than the naïve expansion, our experiments will show that we can achieve comparable or even better retrieval effectiveness.

Both the Naive Expansion and the Bigram Expansion determine word alterations solely according to general knowledge about the language (bigram model or morphological rules), and no consideration about the possible effect of the expansion term is made. In practice, some alterations will have virtually no impact on retrieval effectiveness (as is the case of neutral expansion terms in the previous chapter). They can be ignored. Therefore, in our second method, we will try to predict whether an alteration will have some positive impact on retrieval effectiveness. Only the alterations with positive impact will be retained. This idea has been adopted in chapter 6 for query expansion term selection using a binary SVM classifier. In this chapter, instead of the binary classifier, we will use a regression model to predict the impact on retrieval effectiveness. Compared to the bigram expansion method, the regression method results

in even fewer alterations, while experiments show that the retrieval effectiveness is even better.

Experiments will be conducted on two TREC collections, Gov2 data for Web Track and TREC6&7&8 for ad-hoc retrieval. The results show that the two methods we propose both outperform the original queries significantly with less than two alterations per query on average. Compared to the *Naive Expansion* method, the two methods can perform at least equally well, while query traffic is dramatically reduced.

In the following section, we provide a brief review of related work. Section 7.3 shows how to generate alteration candidates using a similar approach to Xu and Croft's corpus analysis (1998). In section 7.4 and 7.5, we describe the Bigram Expansion method and Regression method respectively. Section 7.6 presents some experiments on TREC benchmarks to evaluate our methods. Section 7.7 concludes this chapter and suggests some avenues for future work.

7.2 Related Work

Many stemmers have been implemented and used as standard processing in IR. Among them, the Porter stemmer (Porter, 1980) is the most widely used. It strips term suffixes step-by-step according to a set of morphological rules, such as the words ended with "-es" or "-ing" will be stripped. However, the Porter stemmer sometimes wrongly transforms a term into an unrelated root. For example, it will unify "news" and "new", "execute" and "executive". On the other hand, it may miss some conflations, such as "mice" and "mouse", "europe" and "european". Krovetz (1993) developed another stemmer, which uses a machine-readable dictionary, to improve the Porter stemmer. It avoids some of the Porter stemmer's wrong stripping, but does not produce consistent improvement in IR experiments (Kraaij and Pohlmann, 1996).

Both stemmers use generic rules for English to strip each word in isolation. In practice, the required stemming may vary from one text collection to another. Therefore, attempts

have been made to use corpus analysis to improve existing rule-based stemmers. Xu and Croft (1998) create equivalence clusters of words which are morphologically similar and occur in similar contexts.

As we stated earlier, the stemming-based IR approaches are not well suited to Web search. Query expansion has been used as an alternative (Peng et al. 2007). To limit the number of expansion terms, and thus the query traffic, Peng et al. only use alterations for some of the query words: They segment each query into phrases and only the head word in each phrase is expanded. The assumptions they made are: 1) Queries issued in Web search often consist of noun phrases. 2) Only the head word in the noun phrase varies in form and needs to be expanded. However, both assumptions may be questionable. Their experiments did not show that the two assumptions hold.

Stemming is related to query expansion or query reformulation (Jones et al., 2006; Anick, 2003; Xu and Croft, 1996), although the latter is not limited to word variants. If the expansion terms used are those that are variant forms of a word, then query expansion can produce the same effect as word stemming. However, if we add all possible word alterations, query expansion/reformulation will run the risk of adding many unrelated terms to the original query, which may result in both heavy traffic and topic drift. Therefore, we need a way to select the most appropriate expansion terms. In (Peng et al. 2007), a bigram language model is used to determine the alteration of the head word that best fits the query. In this chapter, one of the proposed methods will also use a bigram language model of the query to determine the appropriate alteration candidates. However, in our approach, alterations are not limited to head words. In addition, we will also propose a supervised learning method to predict if an alteration will have a positive impact on retrieval effectiveness. To our knowledge, no previous method uses the same approach.

The basic idea we use for selecting word alterations is similar to that used in chapter 6 for selecting good expansion terms from feedback documents. However, there have different expansion term candidates. Although we try to create a similar effect to query expansion with term stemming or term alteration, the candidates for expansion are

restricted to be variants of the one of the original query terms; while the general query expansion considers all related terms, no matter whether they share the same root form or not.

In the following sections, we will describe our approach, which consists of two steps: the generation of alteration candidates, and the selection of appropriate alterations for a query. The first step is query-independent using corpus analysis, while the second step is query-dependent. The selected word alterations will be *OR*-ed with the original query words.

7.3 Generating Alteration Candidates

Our method to generate alteration candidates can be described as follows. First, we do word clustering using a Porter stemmer. All words in the vocabulary sharing the same root form are grouped together. Then we do corpus analysis to filter out the words which are clustered incorrectly, according to word distributional similarity, following (Xu and Croft, 1998; Lin 1998). The rationale behind this is that words sharing the same meaning tend to occur in the same contexts.

The context of each word in the vocabulary is represented by a vector containing the frequencies of the context words which co-occur with the word within a predefined window in a training corpus. The window size is set empirically at 3 words and the training corpus is about 1/10 of the GOV2 corpus (see section 7.5 for details about the collection). Similarity is measured by the cosine distance between two vectors. For each word, we select at most 5 similar words as alteration candidates.

In the next sections, we will further consider ways to select appropriate alterations according to the query.

7.4 Bigram Expansion Model for Alteration Selection

In this section, we try to select the most suitable alterations according to the query context. The query context is modeled by a bigram language model as in (Peng et al. 2007).

Given a query described by a sequence of words, we consider each of the query word as representing a concept c_i. In addition to the given word form, c_i can also be expressed by other alternative forms. However, the appropriate alterations do not only depend on the original word of c_i, but also on other query words or their alterations.

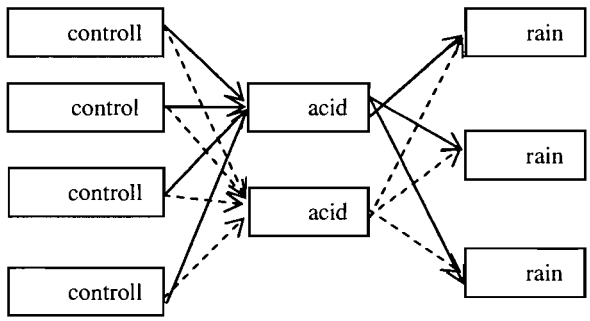

Figure 10. Considering all Combinations to Calculate the Plausibility of Alterations

Accordingly, a confidence weight is determined for each alteration candidate. For example, in the query "Steve Jobs at Apple", the alteration "job" for "jobs" should have a low confidence; while in the query "finding jobs in Apple", it should have a high confidence.

One way to measure the confidence of an alteration is the plausibility of its appearing in the query. Since each concept may be expressed by several alterations, we consider all the alterations of context concepts when calculating the plausibility of a given word. Suppose we have the query "controlling acid rain". The second concept has two alterations - "acidify" and "acidic". For each of the alterations, our method will consider all the combinations with other words, as illustrated in figure 10, where each combination

is shown as a path. More precisely, for a query of n words (or their corresponding concepts), let $e_{i,j} \in c_i$, $j=1,2,\ldots,|c_i|$ be the alterations of concept c_i. Then we have:

$$P(e_{ij}) = \sum_{1,j_1=1}^{|c_1|} \sum_{2,j_2=1}^{|c_2|} \cdots \sum_{i-1,j_{i-1}=1}^{|c_{i-1}|} \sum_{i+1,j_{i+1}=1}^{|c_{i+1}|} \cdots \sum_{n,j_n=1}^{|c_n|} P(e_{1,j_1}, e_{2,j_2}, \ldots, e_{i,j_i}, \ldots, e_{n,j_n}) \quad (7.1)$$

In equation 7.1, $e_{1,j_1}, e_{2,j_2}, \ldots, e_{i,j_i}, \ldots, e_{n,j_n}$ is a path passing through $e_{i,j}$. For simplicity, we abbreviate it as $e_1 e_2 \ldots e_i \ldots e_n$. In this work, we used bigram language model to calculate the probability of each path. Then we have:

$$P(e_1, e_2, \ldots, e_i, \ldots, e_n) = P(e_1) \prod_{k=2}^{n} P(e_k \mid e_{k-1}) \quad (7.2)$$

$P(e_k|e_{k-1})$ is estimated with a back-off bigram language model (Goodman, 2001). In the experiments with TREC6&7&8, we train the model with all text collections; while in the experiments with Gov2 data, we only used about 1/10 of the GOV2 data to train the bigram model because the whole Gov2 collection is too large.

Directly calculating $P(e_{ij})$ by summing the probabilities of all paths passing through e_{ij} is an *NP* problem (Rabiner, 1989), and is intractable if the query is long. Therefore, we use the forward-backward algorithm (Rabiner, 1989; Bishop, 2006) to calculate $P(e_{ij})$ in a more efficient way. After calculating $P(e_{ij})$ for each c_i, we select one alteration which has the highest probability. We limit the number of additional alterations to 1 in order to limit query traffic. Our experiments will show that this is often sufficient.

7.5 Regression Model for Alteration Selection

None of the previous selection methods considers how well an alteration would perform in retrieval. The Bigram Expansion model assumes that the query replaced with better alterations should have a higher likelihood. This approach belongs to the family of unsupervised learning. In this section, we introduce a method belonging to supervised learning family. This method develops a regression model from a set of training data, and it is capable of predicting the expected change in performance when the original query is

augmented by this alteration. The performance change is measured by the difference in the Mean Average Precision (MAP) between the augmented and the original query. The training instances are defined by the original query string, an original query term under consideration and one alteration to the query term. A set of features will be used, which will be defined later in this section.

7.5.1 Linear Regression Model

The goal of the regression model is to predict the performance change when a query term is augmented with an alteration. There are several regression models, ranging from the simplest linear regression model to non-linear alternatives, such as a neural network (Duda et al., 2001), a Regression SVM (Bishop, 2006). For simplicity, we use linear regression model here. We denote an instance in the feature space as X, and the weights of features are denoted as W. Then the linear regression model is defined as:

$$f(X)=W^T X \tag{7.3}$$

where W^T is the transpose of W. However, we will have a technical problem if we set the target value to the performance change directly: The range of values of $f(X)$ is $(-\infty,+\infty)$, while the range of performance change is [-1,1]. The two value ranges do not match. This inconsistency may result in severe problems when the scales of feature values vary dramatically (Duda et al., 2001). To solve this problem, we do a simple transformation on the performance change. Let the change be $y \in [-1,1]$, then the transformed performance change is:

$$\varphi(y) = \log\frac{1+y+\gamma}{1-y+\gamma} \quad y \in [-1,1] \tag{7.4}$$

where γ is a very small positive real number (set to be $1e$-37 in the experiments), which acts as a smoothing factor. The value of $\varphi(y)$ can be an arbitrary real number. $\varphi(y)$ is a monotonic function defined in the range of [-1,1]. Moreover, the fixed point of $\varphi(y)$ is 0,

i.e., $\varphi(y) = y$ when y=*0*. This property is nice; it means that the expansion brings positive improvement if and only if *f(X)>0,* which makes it easy to determine which alteration is better.

We train the regression model by minimizing the mean square error. Suppose there are training instances $X_1, X_2, \ldots, X_m$, and the corresponding performance change is y_i, $i=1,2,\ldots,m$. We calculate the mean square error with the following equation:

$$err(W) = \sum_{i=1}^{m} (W^T X_i - \varphi(y_i))^2 \tag{7.5}$$

Then the optimal weight is defined as:

$$\begin{aligned} W^* &= \arg\min_W err(W) \\ &= \arg\min_W \sum_{i=1}^{m} (W^T X_i - \varphi(y_i))^2 \end{aligned} \tag{7.6}$$

Because *err(W)* is a convex function of *W*, it has a global minimum and obtains its minimum when the gradient is zero (Bazaraa et al., 2006). Then we have:

$$\frac{\partial err(W^*)}{\partial W^*} = \sum_{i=1}^{m} (W^T X_i - \varphi(y_i)) X_i^T = 0$$

So,

$$W^{*T} \sum_{i=1}^{m} X_i X_i^T = \sum_{i=1}^{m} \varphi(y_i) X_i^T$$

In fact, $\sum_{i=1}^{m} X_i X_i^T$ is a square matrix, we denote it as XX^T. Then we have:

$$W^* = (XX^T)^{-1} \left[\sum_{i=1}^{m} \varphi(y_i) X_i \right] \tag{7.7}$$

The matrix XX^T is an $l \times l$ square matrix, where l is the number of features. In our experiments, we only use three features. Therefore the optimal weights can be calculated efficiently even we have a large number of training instances.

7.5.2 Constructing Training Data

As a supervised learning method, the regression model is trained with a set of training data. We illustrate here the procedure to generate training instances with an example.

Given a query "controlling acid rain", we obtain the MAP of the original query at first. Then we augment the query with an alteration to the original term (one term at a time) at each time. We retain the MAP of the augmented query and compare it with the original query to obtain the performance change. For this query, we expand "controlling" by "control" and get an augmented query "*(controlling OR control) acid rain*". We can obtain the difference between the MAP of the augmented query and that of the original query. By doing this, we can generate a series of training instances consisting of the original query string, the original query term under consideration, its alteration and the performance change, for example:

<controlling acid rain, controlling, control, 0.05>

Note that we use MAP to measure performance, but we could well use other metrics such as NDCG (Peng et al., 2007) or *P@N* (precision at *top-N* documents).

7.5.3 Features Used for Regression Model

Three features are used. The first feature reflects to what degree an alteration is coherent with the other terms. For example, for the query "controlling acid rain", the coherence of the alteration "acidic" is measured by the logarithm of its co-occurrence with the other query terms within a predefined window (90 words) in the corpus. That is:

$$log(count(controlling...acidic...rain|window)+0.5)$$

where "..." means there may be some words between two query terms. Word order is ignored.

The second feature is an extension to point-wise mutual information (Rijsbergen, 1979), defined as follows:

$$\log\left(\frac{P(controlling...acidic...rain \mid window)}{P(controlling)P(acidic)P(rain)}\right)$$

where *P(controlling...acidic...rain|window)* is the co-occurrence probability of the trigram containing acidic within a predefined window (50 words). *P(controlling), p(acidic), P(rain)* are probabilities of the three words in the collection. The three words are defined as: the term under consideration, the first term to the left of that term, and the first term to the right. If a query contains less than 3 terms or the term under consideration is the beginning/ending term in the query, we will set the probability of the missed term/terms to be 1. Therefore, it becomes point-wise mutual information when the query contains only two terms. In fact, this feature is supplemental to the first feature. When the query is very long and the first feature always obtains a value of *log(0.5)*, so it does not have any discriminative ability. On the other hand, the second feature helps because it can capture some co-occurrence information no matter how long the query is.

The last feature is the bias, whose value is always set to be 1.0.

As shown in table 31, we use two TREC collections to evaluate the proposed methods. Each collection has 150 queries. We divide the queries into 3 groups, each with 50 queries. The regression model is trained in a leave-one-out cross-validation manner on three groups of queries; each of them is used in turn as a test collection while the two others are used for training. For each incoming query, the regression model predicts the expected performance change when one alteration is used. For each query term, we only select the alteration with the largest positive performance change. If none of its alterations produce a positive performance change, we do not expand the query term. This selection is therefore more restrictive than the Bigram Expansion Model. Nevertheless, our experiments show that it improves retrieval effectiveness further.

7.6 Experiments

7.6.1 Experiment Settings

Table 30. Overview of Test Collections

Name	Description	Size (GB)	#Doc	Query
TREC6&7&8	TREC disk4&5, Newpapers	1.7	500,447	301-450
Gov2	2004 crawl of entire .gov domain	427	25,205,179	701-850

In this section, our aim is to evaluate the two context-sensitive word alteration selection methods. The ideal evaluation corpus should be composed of some Web data. Unfortunately, such data are not publicly available and the results also could not be compared with other published results. Therefore, we use two TREC collections. The first one is the ad-hoc retrieval test collections used for TREC6&7& 8. This collection is relative small and homogeneous. The second one is the Gov2 data. It is obtained by crawling the entire .gov domain and has been used for three TREC Terabyte tracks (TREC2004-2006). Table 31 shows some statistics of the two collections. For each collection, we use 150 queries. Since the Regression model needs some data for training, we divided the queries into three parts, each containing 50 queries. We then use leave-one-out cross-validation. The evaluation metrics shown below are the average value of the three-fold cross-validation. Because the queries in Web are usually very short, we use only the title field of each query.

To correspond to Web search practice, both documents and queries are not stemmed. We do not filter the stop words either.

Two main metrics are used: the Mean Average Precision (MAP) for the top 1000 documents to measure retrieval effectiveness, and the number of terms in the query to reflect query traffic. In addition, we also provide precision for the top 30 documents (P@30) to show the impact on top ranked documents. We also conducted t-tests to determine whether the improvement is statistically significant.

The Indri 2.6 search engine (Strohman et al., 2004) is used as our basic retrieval system. It provides for a rich query language allowing disjunctive combinations of words in queries.

7.6.2 Experimental Results

The first baseline method we compare with only uses the original query, which is named *Original*. In addition to this, we also compare with the following methods:

***Naïve Exp*:** The Naïve expansion model expands each query term with all terms in the vocabulary sharing the same root with it. This model is equivalent to the traditional stemming method.

***UMASS*:** This is the result reported in (Metzler et al., 2006) using Porter stemming for both document and query terms. This reflects a state-of-the-art result using Porter stemming. We report it here for comparison.

***Similarity*:** We select the alterations (at most 5) with the highest similarity to the original term. This is the method described in section 3.

The two methods we propose in this paper are the following ones:

***Bigram Exp*:** the alteration is chosen by a Bigram Expansion model.

***Regression*:** the alteration is chosen by a Regression model.

Table 31. Results of Query 701-750 Over Gov2 Data

Model	P@30	#term	MAP	Imp.
Original	0.4701	158	0.2440	----
UMASS	-------	-------	0.2666	9.26
Naïve Exp	0.4714	1345	0.2653	8.73
Similarity	0.4900	303	0.2689	10.20*
Bigram Exp	0.5007	303	0.2751	12.75**
Regression	0.5054	237	0.2773	13.65**

Table 32. Results of Query 751-800 over Gov2 Data

Model	P@30	#term	MAP	Imp.
Original	0.4907	158	0.2738	----
UMASS	-------	-------	0.3251	18.73
Naive Exp	0.5213	1167	0.3224	17.75**
Similarity	0.5140	290	0.3043	11.14**
Bigram Exp.	0.5153	290	0.3107	13.47**
Regression	0.5140	256	0.3144	14.82**

Table 33. Results of Query 801-850 over Gov2 Data

Model	P@30	#term	MAP	Imp.
Original	0.4710	154	0.2887	----
UMASS	-------	-------	0.2996	3.78
Naïve Exp	0.4633	1225	0.2999	3.87
Similarity	0.4710	288	0.2976	3.08
Bigram Exp	0.4730	288	0.3137	8.66**
Regression	0.4748	237	0.3118	8.00*

Table 34. Results of Query 301-350 over TREC6&7&8

Model	P@30	#term	MAP	Imp.
Original	0.2673	137	0.1669	----
Naïve Exp	0.3053	783	0.2146	28.57**
Similarity	0.3007	255	0.2020	21.03**
Bigram Exp	0.3033	255	0.2091	25.28**
Regression	0.3113	224	0.2161	29.48**

Table 35. Results of Query 351-400 over TREC6&7&8

Model	P@30	#term	MAP	Imp.
Original	0.2820	126	0.1639	-----
Naive Exp	0.2787	736	0.1665	1.59
Similarity	0.2867	244	0.1650	0.67
Bigram Exp.	0.2800	244	0.1641	0.12
Regression	0.2867	214	0.1664	1.53

Table 36. Results of Query 401-450 over TREC6&7&8

Model	P@30	#term	MAP	Imp.
Original	0.2833	124	0.1759	-----
Naïve Exp	0.3167	685	0.2138	21.55**
Similarity	0.3080	240	0.2066	17.45**
Bigram Exp	0.3133	240	0.2080	18.25**
Regression	0.3220	187	0.2144	21.88**

Tables 32, 33, 34 show the results of Gov2 data while table 35, 36, 37 show the results of the TREC6&7&8 collection. In the tables, the * mark indicates that the improvement over the original model is statistically significant with p-value<0.05, and ** means the p-values<0.01.

From the tables, we see that both word stemming (UMASS) and expansion with word alterations can improve MAP for all six tasks. In most cases (except in table 34 and 36), it also improve the precision of top ranked documents. This shows the usefulness of word stemming or word alteration expansion for IR.

We can make several additional observations:

1). Stemming Vs Expansion. UMASS uses document and query stemming while *Naive Exp* uses expansion by word alteration. We stated that both approaches are equivalent. The equivalence is confirmed by our experiment results: for all Gov2 collections, these approaches perform equivalently.

2). The Similarity model performs very well. Compared with the Naïve Expansion model, it produces quite similar retrieval effectiveness, while the query traffic is dramatically reduced. This approach is similar to the work of Xu and Croft (1998), and can be considered as another state-of-the-art result.

3). In comparison, the Bigram Expansion model performs better than the Similarity model. This shows that it is useful to consider query context in selecting word alterations.

4). The Regression model performs the best of all the models. Compared with the Original query, it adds fewer than 2 alterations for each query on average (since each group has 50 queries). Nevertheless we obtained improvements on all the six collections. Moreover, the improvements on five collections are statistically significant. It also performs slightly better than the Similarity and Bigram Expansion methods, but with fewer alterations. This shows that the supervised learning approach, if used in the correct way, is superior to an unsupervised approach. Another advantage over the two other models is that the Regression model can reduce the number of alterations further. Because the Regression model selects alterations according to their expected improvement, the improvement of the alterations to one query term can be compared with that of the alterations to other query terms. Therefore, we can select at most one optimal alteration for the whole query. However, with the Similarity or Bigram Expansion models, the selection value, either similarity or query likelihood, cannot be compared across the query terms. As a consequence, more alterations need to be selected, leading to heavier query traffic.

7.7 Summary of This Chapter

Traditional IR approaches stem terms in both documents and queries. This approach is appropriate for general purpose IR, but is ill-suited for the specific retrieval needs in Web search such as quoted queries or queries with a specific word form that should not be stemmed. The current practice in Web search is not to stem words in index, but rather to perform a form of expansion using word alteration.

However, a naïve expansion will result in many alterations and this will increase the query traffic. This chapter has proposed two alternative methods to select precise alterations by considering the query context. We seek to produce similar or better improvements in retrieval effectiveness, while limiting the query traffic.

In the first method proposed – the Bigram Expansion model, query context is modeled by a bigram language model. For each query term, the selected alteration is the one which maximizes the query likelihood. This method is quite similar to the one used in (Peng et al., 2007), Therefore, it can be treated as a state-of-the-art baseline. In the second method - Regression model, we fit a regression model to calculate the expected improvement when the original query is expanded by an alteration. Only the alteration that is expected to yield the largest improvement to retrieval effectiveness is added. The second method can also be viewed as an extension of the idea adopted in chapter 6, where we addressed the term selection problem in pseudo-relevance feedback.

The proposed methods were evaluated on two TREC benchmarks: the ad-hoc retrieval test collection for TREC6&7&8 and the Gov2 data. Our experimental results show that both proposed methods perform significantly better than the original queries. Compared with traditional word stemming or the naïve expansion approach, our methods can not only improve retrieval effectiveness, but also greatly reduce the query traffic.

This work shows that query expansion with word alterations is a reasonable alternative to word stemming. It is possible to limit the query traffic by a query-dependent selection of word alterations. Our work shows that both unsupervised and supervised learning can be used to perform alteration selection.

Our methods can be further improved in several aspects. For example, we could integrate other features (i.e., word relationships) in the regression model, and use other non-linear regression models, such as Bayesian regression models (e.g. Gaussian Process regression) (Rasmussen and Williams, 2006). The additional advantage of these models is that we can not only obtain the expected improvement in retrieval effectiveness for an alteration, but also the probability of obtaining an improvement (i.e. the robustness of the alteration).

Finally, it would be interesting to test the approaches using real Web data.

Chapter 8

Conclusion and Future Work

Traditional information retrieval models are confronted with some serious problems: 1). The user query is usually very short and ambiguous, which can hardly express user's information need in a precise and complete way. 2). Analogical to the query side, the internal representation of document is also full of ambiguities. 3). The notion of relevance is not well defined, and relevant documents are usually retrieved by "exact term match" strategy. As a consequence, only the relevant document containing at least one query term can be retrieved. Obviously, the strategy overlooks the synonym and polysemy problem. The consequence of it is that many irrelevant documents can be retrieved while a lot of relevant documents can be missed.

The main reason of this situation is that the traditional IR models adopted the independence assumption. The independence assumption between different query terms and between different document terms results in the ambiguity in query and document representations. The independence assumption between a query term and a different document term leads to the "exact term match" strategy for relevant document retrieval. Therefore, the key to resolve the aforementioned problems in the current IR models is to relax the independence assumption.

In this thesis, we proposed approaches to relax the independence assumption by exploiting word relationships. We tried to identify the relationships between query terms or between document terms to alleviate the ambiguity problem: The dependency between terms within a document or within a query is considered to some extent. For example, in the query "Java program", "Java" refers to a common used programming language, while

it means a island in the query "Java tourism". More particularly, our work has focused on the relationships between a query term and document term in order to retrieve relevant documents which do not contain any query term.

Previous studies also exploited word relationships, but they have some limitations compared with our approaches: 1). Most of the studies focus on how to extract word relationships, and few studies examine how to make use of the relationships; 2). Most of them consider a specific relationship such as co-occurrence relationship; 3). The relationships are often integrated into their retrieval framework in a heuristic way.

In this thesis, we exploited the word relationships in the following ways.

- **Document Expansion**

  Document expansion is to use some related additional terms to represent the document. There are at least two effects of document expansion. The first one is reducing ambiguity in document representation. Actually, the main topic of a document can be emphasized if more terms related to the topic are added into the document representation. The second effect is to avoid "exact term match". In chapter 3, we extended Berger and Lafferty's (1999) translation model for document expansion.

- **Query Expansion**

  Query expansion is a similar technique with document expansion. The only difference is that query expansion affects the query side. In fact, we can consider them as two inference process with reverse direction, one from document side to query and another from query to document. Analogically, query expansion aims to reduce the query ambiguity and avoid "exact term matching". In chapter 4, we proposed a Markov Chain based model to expand a query by exploiting word relationships, and achieved substantial improvement over the competitive baseline. This model is even extended to address cross-lingual query expansion in chapter 5. In fact, CLIR can be viewed as a special case of query expansion: instead of expanding a

query by terms in the same language, we replace the original query by terms in the target language. However, the same techniques can be used in both cases. This is demonstrated by the successful re-utilization of the same technique as query expansion for query translation in Chapter 5.

- **Language Modeling Framework**

  We used language model approach as the basic framework. We built a probabilistic model for document and query respectively. The negative KL-divergence between the two models is used to rank the relevant documents. Therefore, we can unify document and query expansion. They are considered as improving the estimation of document and query model respectively. When combining multiple word relationships, the language model framework gives us extra flexibility to adjust the role of individual relationship in a principled way. In this thesis, the weight of each relationship is estimated automatically.

In this thesis, our focus is to extract and exploiting word relationships. We adopted the basic idea to deal with various applications, such as ad-hoc retrieval, cross-lingual information retrieval, pseudo-relevance feedback and query term stemming. For each task, we proposed a model to realize the basic idea. These models are evaluated extensively with a set of TREC or NTCIR test beds. Experimental results show that our models can improve the retrieval effectiveness consistently and significantly. In particular, the thesis made the following contributions.

- We proposed a method to combine complementary word relationships.
- We used Markov Chain model to exploit multi-step relationships between terms, and we showed that this is feasible and useful.
- With Markov Chain models, we extended query translation in cross-lingual information retrieval to query expansion, which unifies query translation and query expansion.
- Finally, we proposed methods to select terms for query expansion or query alteration, according to the potential usefulness of the expansion terms. This

selection can not only further improve retrieval effectiveness, but also reduce query traffic.

Although the proposed models perform very well empirically, much work remains.

In this thesis, we used two methods to extract word relationships. The first one is according term co-occurrence, i.e., terms co-occurring frequently are considered to be related. The second one is based on manually created thesauri (WordNet in this thesis). In fact, the first one is an unsupervised learning approach and the second is a supervised learning approach (the relationships are labeled). Both approaches have pros and cons. The first one has high coverage but low accuracy and second one has high accuracy but low coverage. In the future, we would like to use semi-supervised learning approaches to extract term relationships, such as bootstrapping (Yarowsky, 1995). We can use a small set of terms whose relationships are labeled manually as a seed. Then we bootstrap the seed to include more terms whose relationships are extracted. This approach is expected to be less expensive than manually labeling data and more accurate than unsupervised learning approaches.

In this thesis, we exploited multiple word relationships, i.e., statistical relationships such as co-occurrence and proximity, and semantic relationships defined in WordNet, such as synonym, hypernym and hyponym. We have not used syntactic relationships. In our future work, we would like to integrate them.

Another limitation of our work is that we only evaluated the models with TREC or NTCIR data. Although they are benchmarks used on IR community, they are different from the real Web data. It is interesting to evaluate these models with some Web data.

Bibliographies

Anick, P. (2003) Using Terminological Feedback for Web Search Refinement: a Log-based Study. In SIGIR, pp. 88-95.

Baeza-Yates, R., and Ribeiro-Neto, B. 1999. Modern Information Retrieval. Addison Wesley Press.

Bai, J. Nie, J., Bouchard, H. and Cao, G. Using query contexts in information retrieval. In the Proceedings of SIGIR'2007, Armsterdam, Netherlands, 2007.

Bai, J., Song, D., Bruza, P., Nie, J.-Y. and Cao, G. (2005). Query Expansion Using Term Relationships in Language Models for Information Retrieval. In Proceedings Proceedings of the 14th CIKM, pp. 688-695.

Ballesteros, L. and Croft, W. (1998). Resolving Ambiguity for Cross-language Retrieval. In Proceedings of ACM SIGIR 1998 Conference, pages 64-71.

Bazaraa, M., Sherali, H., and Shett, C. (2006). Nonlinear Programming, Theory and Algorithms. John Wiley & Sons Inc.

Berger, A. and Lafferty, J. (1999). Information retrieval as statistical translation. In Proceedings of the 1999 ACM SIGIR Conference on Research and Development in Information Retrieval, pages 222-229.

Bergman, M. (2001). The deep web: surfacing hidden value. Available at http://www.brightplanet.com/images/stories/pdf/deepwebwhitepaper.pdf

Bishop, C. (2006). Pattern Recognition and Machine Learning. Springer.

Brémaud, P. (1999) Markov chains: Gibbs fields, monte carlo simulations, and queues. Springer-Verlag.

Brin, S., and Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. WWW7/Computer Networks and ISDN Systems, 20, pp. 107-117.

Brown, P., Della Pietra, S., Della Pietra, V., and Mercer, R. (1993). The mathematics of statistical machine translation: Parameter estimation. Computational Linguistics, 19(2), pp. 263-311.

Burges, C. A Tutorial on Support Vector Machines for Pattern Recognition. Data Mining and Knowledge Discovery, Vol.2, Number 2, p.121-167, Kluwer Academic Publishers, 1998.

Burgess, C., Livesay, K., and Lund, K. (1998). Explorations in Context Space: Words, Sentences, Discourse. Discourse Processes, 25(2&3), 211-257

Callan, J. Passage level evidence in document retrieval. In the Proceedings of SIGIR1994.

Cao, G., Nie, J.Y., and Bai, J. (2005). Integrating word relationships into language modeling. In Proceedings of the 2005 SIGIR pp. 298-305

Cao, G., Nie, J.Y., and Bai, J. Constructing better document and query models with Markov Chains. In the Proceedings of CIKM, pp.800-801, 2006.

Charniak, E. (2001). Immediate-head parsing for language models. In Proceedings of ACL' 2001.

Chelba, C. and Jelinek, F. (1998). Exploiting syntactic structure for language modeling. In Proceedings of COLING-ACL 1998.

Chen, S.F. and Goodman, J. (1998). An empirical study of smoothing techniques for language modeling. Tech. Rep. TR-10-98, Harvard University.

Collins-Thompson, K, and Callan, J. (2005). Query Expansion Using Random Walk Models. In Proceedings of CIKM, pp.704-711.

Cover, T., Thomas, J.A. 1991. Elements of Information Theory.

Cristianini, N., and Shawe-Taylor, J. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods. Cambridge University Press, 2000.

Croft, W. B. (1993). Knowledge-based and statistical approaches to text retrieval.

Croft, W.B. and Lafferty, J. (2003). Language Models for Information Retrieval. Kluwer Int. Series on Information Retrieval, Vol. 13, Kluwer Academic Publishers.

Davis, M.W., and Ogden, W.C. (1997). Free resources and advanced alignment for cross-language text retrieval. In the Proceedings of TREC6. NIST, Gaithersburg, MD.

Deerwester, S., Dumains, S.T., Furnas, G.W., Landauer, T.K., and Harshman, R. (1990). Indexing by Latent Semantic Analysis. Journal of the Society for Information Science, 41(6), 391-407

Dempster, A. , Laird, N. and Rubin, D. Maximum likelihood from incomplete data via the EM algorithm. Journal of the Royal Statistical Society. Series B. 39(1):1-38, 1977

Diligenti, M., Gori, M., and Maggini, M. (2005). Learning web page scores by error back-propagation. In the Proceedings of IJCAI. pp. 684-689.

Duda, R., Hart, P., and Stork, D. (2000) Pattern Classification (2nd Edition). John Wiley and Sons, 2000, ISBN 0471056693

Dunning, T. 1993. Accurate methods for the statistics of surprise and coincidence. Computational Linguistics. 19: 61-74.

Foo, S. and Li, H. Chinese word segmentation and its effect on information retrieval. Information Processing and Management, Vol 41(1), pp. 161-190.

Fagan, J. Automatic phrase indexing for document retrieval: An examination of syntactic and non-syntactic methods. In Proc. tenth Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval (1987), pp. 91–101.

Gao, J., Nie, J.Y., Zhang, J., Xun, E., Zhou, M., and Huang, C.N. (2001). Improving query translation for CLIR using statistical Models. In:Conference on Research and Development in Information Retrieval, ACM SIGIR'01, New Orleans, Louisiana, USA. September 9-12, 2001.

Gao, J., Goodman, J., Cao, G. and Li, L. Exploring asymmetric clustering for statistical language modeling. ACL2002, University of Pennsylvania, Philadelphia, PA, USA. July 6-12, 2002

Gao, J., Nie, J.Y. (2006). A Study of Statistical Models for Query Translation: Find a Good Unit of Translation. In Proceedings of ACM SIGIR, pp. 194-201

Gao, J., Nie, J.-Y., Wu, G., and Cao, G. (2004). Dependence Language Model for Information Retrieval. In Proceedings of the 2004 SIGIR , pp. 170-177.

Gao, J., Qi, H., Xia, X., and Nie, J.-Y. (2005). Linear discriminative model for information retrieval. In Proceedings of the 2005 SIGIR, pp. 290-297

Goodman, J. (2001). A Bit of Progress in Language Modeling. Technical report.

Grefenstette, G. (1999). The World Wide Web as a resource for example-based machine translation tasks, In Proc. ASLIB translating and the computer 21 conference.

He, H., Gao, J. (2001). NTCIR-3 CLIR Experiments at MSRA In the Proceedings of NTCIR3.

Hedlund, T., Airio, E., Keskustalo, H. Pirkola, A., Jarvelin, K. (2004) Dictionary-based Cross Language Information Retrieval: Learning Experiences from CLEF 2000-2002. Information Retrieval, 7: 99-119.

Hiemstra, D. 2000. Using Language Models for Information Retrieval. PhD dissertation. Twenty-one University

Hiemstra, D. and Kraaij, W. (1998). Twenty-one at trec-7: Ad-hoc and cross-language track. In Proceedings of Seventh Text Retrieval Conference (TREC-7).

Hofmann, T. (1999): Probabilistic Latent Semantic Analysis. In the Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval, Berkeley, California, pp.50-57, ACM Press.

Hsu, C. Chang, C. and Lin, C. A practical guide to support vector classification. Technical Report, National Taiwan University.

Hull, D. and Grefenstette, G. (1996). Querying across languages: A dictionary-based approach to multilingual information retrieval. In Proceedings of ACM SIGIR, pp.49-57.

Jelinek, F. (1998). Statistical Methods for Speech Recognition. MIT Press, Cambridge, MA.

Jin, Rong, Hauptmann, A.G., and Zhai, CX. (2002). Title Language Model for Information Retrieval. In Proceedings of the 2002 SIGIR, pages 42-48

Joachims, T. Text categorization with support vector machines: learning with features. In ECML, pp.137-142, 1998.

Jones, R., Rey, B., Madani, O., and Greiner, W. (2006). Generating Query Substitutions. In WWW2006, pp. 387-396

Kirkpatrick, S., Gelatt C., and Vecchi M., 1983. Optimization by Simulated Annealing. Science, 220(4598): 671-680.

Kleinberg, J. (1998). Authoritative sources in a hyper-linked environment. In the Proceedings of 9th ACM-SIAM Symposium on Discrete Algorithms.

Kraaij, W. and Pohlmann, R. (1996) Viewing Stemming as Recall Enhancement. Proc. SIGIR, pp. 40-48.

Kraaij, W., Nie, J.Y., and Simard, M. (2003). Embedding Web-Based Statistical Translation Models in Cross-Language Information Retrieval. Computational Linguistics, 29(3): 381-420.

Kraaij, W., Westerveld, T., and Hiemstra, D. 2002. The importance of prior probabilities for entry page search. In Proceedings of SIGIR2002, pp.27-34

Kraft, D. H. and Buell, D. A. (1983). Fuzzy sets and generalized Boolean retrieval systems. International Journal on Man-Machine Studies, 19: pp. 49-56.

Krovetz, R. (1993). Viewing Morphology as an Inference Process. Proc. ACM SIGIR, pp. 191-202.

Kurland, O. and Lee, L. (2004). Corpus Structure, language Models, and Ad Hoc Information Retrieval. In the Proceedings of SIGIR2004.

Kurland, O., and Lee, L. (2005). Pagerank without hyperlinks: Structural re-ranking using links induced by language models. In Proceedings of ACM SIGIR. pp. 306-313

Kwok, K.L, Grunfeld, L., Chan, K., THREC-8 ad-hoc, query and filtering track experiments using PIRCS, In TREC10, 2000.

Kwok, K.L. (2000). Exploiting a Chinese-English bilingual wordlist for English-Chinese cross language information retrieval. In the Proceedings of the Fifth International Workshop on Information Retrieval with Asian Languages, IRAL-2000. pp. 173-179.

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

Lavrenko, V. and Croft, W.B. Relevance-Based Language Models. In Proceedings of the 2001 SIGIR, pp. 120-127.

Lavrenko, V., Choquette, M. and Croft, B. Cross-lingual Relevance Model. In the Proceedings of SIGIR 2002, pp.175-182.

Lawrence, S. and Giles, C.L. (1999). Accessibility of information on the web. Nature, 400:107-109

Lee, C., Lee G.G. (2005). Dependency Structure Language Model for Information Retrieval. http://cir.dcs.vein.hu/cikkek/dslm_camera_ready.pdf

Lin, C.-Y., Cao, G.H., Gao, J.F., and Nie, J.-Y., (2006). An Informaiton-Theoretic Approach to Automatic Evaluation of Summaries. To appear in the Proceedings of HLT-NAACL 2006, New York, USA.

Lin, D. (1998). Automatic Retrieval and Clustering of Similar Words. In COLING-ACL, pp. 768-774.

Liu, S., Liu, F., Yu, C., and Meng, W., (2004). An Effective Approach to Document Retrieval via Utilizing WordNet and Recognizing Phrases. . In Proceedings of the 2004 ACM SIGIR Conference on Research and Development in Information Retrieval, pages 266-272.

Liu, S., Liu, F., Yu, C., and Meng, W., (2004). An Effective Approach to Document Retrieval via Utilizing WordNet and Recognizing Phrases. . In Proceedings of the 2004 ACM SIGIR Conference on Research and Development in Information Retrieval, pages 266-272.

Liu, X. and Croft, B. (2004). Cluster-based Retrieval Using Language Models. In the Proceedings of SIGIR 2004.

Mandala, R., Tokunaga, T., and Tanaka, H. (1998). Ad Hoc Retrieval Experiments Using WordNet and Automatically Constructed Theasuri. In Proceedings of the seventh Text REtreival Conference, pages 475-481.

Manning, C., Raghavan, P. And Schute, H. (2008). Introduction to Information Retrieval. Cambridge University Press, 2008.

McCarley, J. (1999). Should We Translate the Documents or the Queries in Cross-language Information Retrieval. In Proceedings of ACL 1999, pages 208-214

McNamee, P. and Mayfield, J. (2002). Comparing Cross-Language Query Expansion Techniques by Degrading Translation Resources. In the Proceedings of ACM SIGIR, pp. 159-166.

Metzler, D. and Croft, B. Latent Concept Expansion Using Markov Random Fields. In the Proceedings of SIGIR'2007, pp.311-318.

Metzler, D., Strohman, T. and Croft, B. (2006). Indri TREC Notebook 2006: Lessons learned from Three Terabyte Tracks. In the Proceedings of TREC 2006.

Metzler, D. Beyond bags of words: effectively modeling dependency and features in information retrieval. Ph.D. Thesis, University of Massachusetts, 2007.

Miller, D., Leek, T. and Schwartz, R. (1999). A hidden Markov model information retrieval system. In Proceedings of the 1999 SIGIR, pp. 214-222.

Minkov, E., Cohen, W., and Ng, A. (2006). A Graphical Framework for Contextual Search and Name Disambiguation in Email. In the Proceedings of ACM SIGIR, pp. 27-34.

Monz, C., and Dorr, B., (2005). Iterative translation disambiguation for cross-language information retrieval. In the Proceedings of ACM SIGIR, pp. 520-527.

Mooers, C.N. 1950. Information retrieval viewed as temporal signaling. In Proceedings of the International Congress of Mathematics, Volume 1, pp.572-573.

Morgan, W, Strohman, T., and Henderson, J. (2004). Dicrect maximization of average precision by hill-climbing with a comparison to a maximum entropy approach. Technical report. MITRE.

Nallapati, R. and Allan, J. (2002). Capturing Term Dependencies using a Language Model based on Sentence Trees. In Proceedings of the eleventh international conference on Information and knowledge management, McLean, Virginia, USA, 2002, pages 383-390.

Ng, A.Y., Zheng, A.X., and Jordan, M. (2001). Link analysis, eigenvectors, and stability. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01).

Ng, K. (1999). A Maximum Likelihood Ratio Information Retrieval Model. In TREC-8 Workshop notebook.

Nie, J.Y. (1988). An Outline of a General Model for Information Retrieval Systems. *In Proceedings of the 1988 SIGIR,* pp. 495-506

Nie, J.Y., Cao, G. and Bai, J. Inferential language models for information retrieval. ACM Transaction on Asian Language Processing (TALIP), 5(4):323-359, 2006

Nie, J.Y., Simard, M., Isabelle, P., Durand, R. "Cross-Language Information Retrieval based on Parallel Texts and Automatic Mining of Parallel Texts in the Web", 22ndACM-SIGIR, Berkeley, 1999, pp. 74-81

Nocedal, J. and Wright, S. Numerical optimization. Springer, 2006.

Oard, D. (1998). A Comparative Study of Query and Document Translation for Cross-language Information Retrieval. In Third Conference of the Association for Machine Translation in the Americas.

Och, F. (2003). Minimum error rate training in statistical machine translation. In Proceedings of ACL. pp. 160-67

Och, F., and Ney, H. (2000). Improved statistical alignment models. In Proceedings of ACL. pp. 440-447.

Ogilvie, P. and Callan, J. (2001). Experiments using the lemur toolkit. In Proceedings of the Tenth Text Retrieval Conference (TREC-10), pages 103–108.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1998). The PageRank citation ranking: Bringing order to the web. Technical Report, Computer Science department, Stanford University.

Peat, H. and Willett, P., The limitations of term co-occurrence data for query expansion in document retrieval systems. JASIS, 42(5): 378-383, 1991.

Peng, F., Ahmed, N., Li, X., and Lu, Y. (2007). Context Sensitive Stemming for Web Search. Proc. ACM SIGIR, pp. 629-636 .

Platt, J. Probabilities for SV Machines. Advances in large margin classifiers, pages 61-74, Cambridge, MA, 2000. MIT Press

Platt, J. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research, 1998.

Ponte, J. A language modeling approach to information retrieval. (1998b) Doctal thesis, Univ. of Mass. At Amherst.

Ponte, J. and Croft, W.B. (1998). A language modeling approach to information retrieval. In Proceedings of the 1998 ACM SIGIR Conference on Research and Development in Information Retrieval, pages 275-281.

Porter, M. (1980) An Algorithm for Suffix Stripping. Program, 14(3): 130-137.

Qiu, Y.G., and Frei, H.P. (1993). Concept query expansion. In the Proceedings of ACM SIGIR, pp. 160-169.

Rabiner, L. (1989). A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In Proceedings of IEEE Vol. 77(2), pp. 257-286.

Radecki, T. (1979). Fuzzy set theoretical approach to document retrieval. *Information Processing and Management,* 15: pp. 247-259.

Rijsbergen, V. (1979). Information Retrieval. Butterworths, second version.

Roark, B. (2001). Probabilistic top-down parsing and language modeling. Computational Linguistics, 27(2).

Robertson, S., Maron, M. and Cooper, W.S. Probability of relevance: a unification of two competing models for information retrieval. Information Technology - Research and Development. 1. 1-21 (1982)

Robertson, S. and Spark Jones, K. (1976). Relevance weighting of search terms. Journal of the American Society for Information Science, 27:129-146

Robertson, S., Walker, S., Hancock-Beaulieu, M., Gull, A., and Lau, A. 1992. Okapi at TREC. In the Proceedings of TREC1.

Robertson, S.E., On term selection for query expansion, Journal of Documentation, 46(4): 359-364. 1990.

Robertson, S.E., Van, Rijsbergen, C.J., and Poter, M.F., (1981). Probabilistic models of indexing and searching. In Information Retrieval Research, R.N. Odd et al, Eds. Butterworths, pages 35-56.

Rocchio, J. (1971) Relevance feedback in information retrieval. In the SMART Retrieval System: Experiments in Automatic Document Processing, pages 313-323. Prentice-Hall Inc., 1971.

Ross, S. Introduction to probability models (Eighth edition). Academic Press, 2003.

Salton, G. and Buckley, C. 1988. Term weighting approaches in automatic text retrieval. Information Processing and Management 24, 513-523

Salton, G. and McGill, M. 1983. Introduction to Modern Information Retrieval. McGraw-Hill

Salton, G., Fox, E. A. and Wu, H. (1983). Extended Boolean information retrieval. Communications of the ACM, 26(12): pp. 1022-1036.

Salton, G., Wong, A., and Yang, C.S. (1975). A vector space model for automatic indexing. Communications of the ACM, 18(11):613-620

Sherman, C. (2001). Google fires new salvo in search engine size wars. SearchDay, (157). http://searchenginewatch.com/

Silverstein, C., Henzinger, M., and Moricz, M. 1998. Analysis of a very large AltaVista qery log. SRC Tech. Note 1998-014, Compaq Systems Research Center, Palo Alto, CA. Website: http://www.research.compaq.com/SRC/publications

Singhal, A. 2001. Modern information retrieval: A brief overview. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering.

Smeaton, A. F. and Van Rijsbergen, C. J. The retrieval effects of query expansion on a feedback document retrieval system. Computer Journal, 26(3): 239-246. 1983.

Song, D. and Bruza, P.D. (2003). Towards Context-sensitive Information Inference. Journal of the American Society of Information Science and Technology (JASIST), Vol.54, 321-334.

Song, F. and Croft, B. (1999). A general language model for information retrieval. In Proceedings of SIGIR' 1999, pp.279-280.

Sparck Jones, K. 1971. Automatic keyword classification for information retrieval. Butterworths.

Sparck Jones, K., Walker, S., and Robertson, S.E. (2000). A probabilistic model of information retrieval: development and comparative experiments – part 1 and part 2. Information Processing and Management, 36(6):779-808 and 809-840.

Spink, A., Wolfram, D., Jansen, B.J., and Saracevic, T. 2001. Searching the Web: the public and their queries. Journal of American Society of Information Science and Technology. 52(3), 226-234

Srikanth, M. and Srikanth, R. (2002). Biterm language models for document retrieval. *In Proceedings of the 2002 ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 425-426.

Steinbach, M., Karypis, G., and Kumar, V. (2000). A comparison of document clustering technique. In *KDD Workshop on Text Ming.*

Stolcke, A. (2002). SRILM – An Extensible Language Modeling Toolkit. In Proceedings of International Conference Spoken Language Processing, vol.2, pp.901-904, Denver.

Strohman, T., Metzler, D. and Turtle, H., and Croft, B. (2004). Indri: A Language Model-based Search Engine for Complex Queries. In Proceedings of the International conference on Intelligence Analysis.

Sullivan, D. (2005). Search Engine Sizes. http://searchenginewatch.com/reports/

Tao, T. and Zhai, C. An exploration of proximity measures information retrieval. In the Proceedings of SIGIR'2007, pp.295-302, 2007.

Tao, T. and Zhai, C. Regularized estimation of mixture models for robust pseudo-relevance feedback. In the Proceedings of SIGIR'2006.

Toutanova, K., Manning, C. and Ng, A. (2004). Learning Random Walk Models for Inducing Word Dependency Distributions. In the Proceedings of the 21st International Machine Learning Conference, ACM Press, 2004

Vapnik, V. Statistical Learning Theory. New York: Wiley, 1998

Voorhees, E. (1994). Query Expansion Using Lexical-Semantic Relations In Proceedings of the 1994 ACM SIGIR Conference on Research and Development in Information Retrieval, pages 61-69.

Voorhees, E. M. (2000). The TREC-8 question answering track report. In the Proceedings of TREC-8.

Wang, J. and Oard, D. (2006). Combining Bidirectional Translation and Synonymy for Cross-Language Information Retrieval. In the Proceedings of ACM SIGIR. pp. 202-209.

Wang, M. and Si, L. Discriminative probabilistic models for passage based retrieval. In the Proceedings of SIGIR2008, pp. 419-426.

Wei, X. and Croft, W. B. , "Modeling Term Associations for Ad-hoc Retrieval Performance within Language Modeling Framework," Proceedings of the 29th European Conference on Information Retrieval (ECIR 2007), pp. 52-63

Xu, J. and Croft, B. (1996). Query Expansion Using Local and Global Document Analysis. Proc. ACM SIGIR, pp. 4-11.

Xu, J. and Croft, B. (1998). Corpus-based Stemming Using Co-occurrence of Word Variants. ACM TOIS, 16(1): 61-81.

Xu, J. and Weischedel, R. (2005). Empirical studies on the impact of lexical resources on CLIR performance. Information Processing and Management, 41, pp.475-487

Xu, J., and Weischedel, R. (2000). Cross-lingual information retrieval using Hidden Markov models. In the Proceedings of SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora. pp. 95-103.

Xu, J.X., Weischedel, R., and Nguyen, C. (2001). Evaluating a Probabilistic Model for Cross-lingual Information Retrieval. In Proceedings of ACM SIGIR, pp. 105-110.

Yarowsky, D. Unsupervised word sense disambiguation rivalling supervised methods. In the Proceedings of 33rd ACL conference, pp. 189-196, 1995.

Zhai, C., and Lafferty, J. (2001a). A Study of Smoothing Methods for Language Models Applied to Information Retrieval. In Proceedings of the 2001 ACM SIGIR Conference on Research and Development in Information Retrieval, pages 334-342.

Zhai, C. and Lafferty, J. (2001b). Model-based Feedback in the Language Modeling Approach to Information Retrieval. *In Proceedings Proceedings of the 10th CIKM,* pp. 403-410.

Zhai, C. and Lafferty, J. (2002). Two-stage language models for information retrieval. *In Proceedings of the 2002 ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49-56

Zhai, C. PhD dissertation. Carnegie Mellon University, 2003.

Zhai, C., Cohen, W. And Lafferty, J. Beyond independent relevance: Methods and evaluation metrics for subtopic retrieval. In the Proceedings of SIGIR2003, pp.10-17

Zhang, Y., Callan, J. And Minka, T. Novelty and redundancy detection in adaptive filtering. In the Proceedings of SIGIR2002, pp. 81-88.

Zhang, Y., Callan, J., The bias problem and language models in adaptive filtering. In the Proceedings of TREC11, pp.78-83, 2001

Zhang, Y., Xu, W. and Callan, J. Exact maximum likelihood estimation for word mixture. In the Proceedings of Text learning workshop in International Conference on Machine Learning, Sydney, Australia, 2002.