Université de Montréal

RELATING DEPENDENT TERMS IN INFORMATION RETRIEVAL

par

Lixin Shi

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

Novembre 2015

RÉSUMÉ

Les moteurs de recherche font partie de notre vie quotidienne. Actuellement, plus d’un tiers de la population mondiale utilise l’Internet. Les moteurs de recherche leur permettent de trouver rapidement les informations ou les produits qu'ils veulent.

La recherche d'information (IR) est le fondement de moteurs de recherche modernes. Les approches traditionnelles de recherche d'information supposent que les termes d'indexation sont indépendants. Pourtant, les termes qui apparaissent dans le même contexte sont souvent dépendants. L’absence de la prise en compte de ces dépendances est une des causes de l’introduction de bruit dans le résultat (résultat non pertinents). Certaines études ont proposé d’intégrer certains types de dépendance, tels que la proximité, la cooccurrence, la contiguïté et de la dépendance grammaticale. Dans la plupart des cas, les modèles de dépendance sont construits séparément et ensuite combinés avec le modèle traditionnel de mots avec une importance constante. Par conséquent, ils ne peuvent pas capturer correctement la dépendance variable et la force de dépendance. Par exemple, la dépendance entre les mots adjacents "Black Friday" est plus importante que celle entre les mots "road constructions".

Dans cette thèse, nous étudions différentes approches pour capturer les relations des termes et de leurs forces de dépendance. Nous avons proposé des méthodes suivantes:

- Nous réexaminons l'approche de combinaison en utilisant différentes unités d'indexation pour la RI monolingue en chinois et la RI translinguistique entre anglais et chinois. En plus d’utiliser des mots, nous étudions la possibilité d'utiliser bi-gramme et uni-gramme comme unité de traduction pour le chinois.

Plusieurs modèles de traduction sont construits pour traduire des mots anglais en uni-grammes, bi-grammes et mots chinois avec un corpus parallèle. Une requête en anglais est ensuite traduite de plusieurs façons, et un score classement est produit avec chaque traduction. Le score final de classement combine tous ces types de traduction.

- Nous considérons la dépendance entre les termes en utilisant la théorie d'évidence de Dempster-Shafer. Une occurrence d'un fragment de texte (de plusieurs mots) dans un document est considérée comme représentant l'ensemble de tous les termes constituants. La probabilité est assignée à un tel ensemble de termes plutôt qu'a chaque terme individuel. Au moment d'évaluation de requête, cette probabilité est redistribuée aux termes de la requête si ces derniers sont différents. Cette approche nous permet d'intégrer les relations de dépendance entre les termes.

- Nous proposons un modèle discriminant pour intégrer les différentes types de dépendance selon leur force et leur utilité pour la RI. Notamment, nous considérons la dépendance de contiguïté et de cooccurrence à de différentes distances, c'est-à-dire les bi-grammes et les paires de termes dans une fenêtre de 2, 4, 8 et 16 mots. Le poids d'un bi-gramme ou d'une paire de termes dépendants est déterminé selon un ensemble des caractères, en utilisant la régression SVM.

Toutes les méthodes proposées sont évaluées sur plusieurs collections en anglais et / ou chinois, et les résultats expérimentaux montrent que ces méthodes produisent des améliorations substantielles sur l'état de l'art.

Mots-clés: recherche d'information, modèle de langue, unité de traduction, recherche d'information translinguistique, la théorie de Dempster-Shafer, dépendance de termes, modèle discriminant, force de dépendance.

ABSTRACT

Search engine has become an integral part of our life. More than one-third of world populations are Internet users. Most users turn to a search engine as the quick way to finding the information or product they want.

Information retrieval (IR) is the foundation for modern search engines. Traditional information retrieval approaches assume that indexing terms are independent. However, terms occurring in the same context are often dependent. Failing to recognize the dependencies between terms leads to noise (irrelevant documents) in the result. Some studies have proposed to integrate term dependency of different types, such as proximity, co-occurrence, adjacency and grammatical dependency. In most cases, dependency models are constructed apart and then combined with the traditional word-based (unigram) model on a fixed importance proportion. Consequently, they cannot properly capture variable term dependency and its strength. For example, dependency between adjacent words “black Friday” is more important to consider than those of between “road constructions”.

In this thesis, we try to study different approaches to capture term relationships and their dependency strengths. We propose the following methods for monolingual IR and Cross-Language IR (CLIR):

- We re-examine the combination approach by using different indexing units for Chinese monolingual IR, then propose the similar method for CLIR. In addition to the traditional method based on words, we investigate the possibility of using Chinese bigrams and unigrams as translation units. Several translation models from English words to Chinese unigrams, bigrams and words are created based on a parallel corpus. An English query is then translated in several ways, each producing a ranking score. The final ranking score combines all these types of translations.

- We incorporate dependencies between terms in our model using Dempster-Shafer theory of evidence. Every occurrence of a text fragment in a document is represented as a set which includes all its implied terms. Probability is assigned to such a set of terms instead of individual terms. During query evaluation phase, the probability of the set can be transferred to those of the related query, allowing us to integrate language-dependent relations to IR.
- We propose a discriminative language model that integrates different term dependencies according to their strength and usefulness to IR. We consider the dependency of adjacency and co-occurrence within different distances, i.e. bigrams, pairs of terms within text window of size 2, 4, 8 and 16. The weight of bigram or a pair of dependent terms in the final model is learnt according to a set of features.

All the proposed methods are evaluated on several English and/or Chinese collections, and experimental results show these methods achieve substantial improvements over state-of-the-art baselines.

Keywords: Information retrieval, Language modeling, Translation unit, CLIR, Dempster-Shafer theory, Term dependency, Discriminative model, Dependency strength.

ACKNOWLEDGEMENT

I would like to express deepest gratitude to my advisor Professor Jian-Yun Nie for his full support, expert guidance, understanding and encouragement throughout my study and research. From his immense knowledge, expertise, hard work and keen insight, I learned how to do rigorous research. Without his incredible patients and timely wisdom and counsel, I could not enter this research area and complete my thesis work.

In addition, I express my appreciation to committee members: Prof. Jimmy Huang, Pascal Vincent, and Guy Lapalme. Their thoughtful questions and comments were valued greatly.

I would like to thank Prof. Philippe Langlais, Elliott Macklovitch, Fabrizio Gotti and the fellow students, Jing Bai, Youssef Kadri, Guihong Cao, Olga Feiguina, Wei Yuan for their kindness and helps when I was studying in RALI lib.

Also, a special thanks to Prof. Colin F. Davidson, the Cibât project team, Professor Francois Lepage, and Doctor Atefeh Farzindar for providing me the financial support and intern opportunities.

I would like to acknowledge Professor Xinming Ye and Guanglai Gao from Inner Mongolia University for their helps and continuous encouragement. I would like to thank everyone who has helped me during my doctoral studies.

Finally, a special thanks to my wife, my parents and my mother-in-law for their unconditional love and support during the past years. I would not have been able to complete this thesis without their sacrifices and support.

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	A probabilistic model for a document
θ_D	A probabilistic model for a query
$\arg\max f$	The argument for which f has its maximum value
$f \propto g$	f is proportional to g
$\binom{n}{r}$	Combination or binomial coefficient (#ways of choosing r objects from n)
$KL(p\|\|q)$	Kullback-Leibler (KL) divergence
(a, b)	The ordered pair of x and y
$\{a, b\}$	The un-ordered pair of x and y
$(abc)^*$	Term set. It includes all the terms formed from $a, b,$ and c
$\mathbb{R}$	The set of real numbers
$\overset{\text{rank}}{=}$	Rank equivalence (same ranking order)
$P(A\|B)$	The probability of A conditional of B

CHAPTER 1.
INTRODUCTION

1.1 Motivation and Problems

Information retrieval (IR) plays an increasingly important role in people's daily life and work. Nowadays, people tend to obtain information from information retrieval systems (search engines) rather than asking other people. According to Internet usages statistics[1], the current world Internet users are 3.08 billion, 42% of world population. From year 2000 to 2015, the growth of Internet user number is 753%.

As defined 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.*" Here we note the user's *need for information* (information need) as I, the *documents in storage* (document collection) as C, and the *actual list of citations* (ranked list) as L. To use an IR system, a user should represent his information need to a query (Q), which is usually a short natural language sentence, a Boolean expression or just some keywords. The information retrieval process can be expressed as follows:

$$I \xrightarrow{user} Q \xrightarrow{IR\ system} C \xrightarrow{user} L$$

[1] http://www.internetworldstats.com/stats.htm, 2015-07-29

The task of an IR system is to select documents that could be relevant to a user's information need and present his/her in a ranked list. In its traditional setting, IR faces the following problems:

- How to represent the documents and queries in a retrieval model;
- How to understand a user's query and *guess* the real information need from the query.

In traditional IR environment, we usually assume that the query provided by a user is the only description available for the information need. So the focus is on determining the relevance of a candidate document to the query. In modern search engines, more information is gathered from a user in addition to the query, such as previous queries the user issued in the same session, documents he/she clicked on, documents the other users have clicked on for the same query in the past, etc. The additional information has been proven very helpful to better guess the user's intent behind a query such as (Speretta & Gauch 2005), (Gao, He & Nie 2010). However, the new setting of search engine does not weaken the role of determining the relevance of a document to a query, i.e. the traditional document-query relationship. In search engines, the relationship between a document and a query is still the most important criterion to rank documents in the search result. In this thesis, we will work in the traditional setting, i.e. we assume that we only have a query without any peripheral information, and our task is to best rank documents according to the query.

In the most common setting of IR, both user's queries and documents in the collection are written in a natural language. To match documents and queries, we have to create an internal representation for them. The common approach is to use a set of independent words (or terms) to represent each of them. Such an approach is commonly called a "bag-of-words" approach. A score function is defined between a document and a query according to how much their "bags of words" overlap – there are different ways to define such a function, as we will see in Chapter 2.

A key limitation, which is widely recognized in IR, is that the bag-of-words representation is unable to cope with the dependencies between words in a natural language sentence. For example, if a query about "computer architecture" is represented by two independent words – *computer* and *architecture,* it can match incorrectly a document talking about "the use of *computer* in *architecture*". To cope with this problem, a large number of approaches have been proposed in the IR literature, ranging from the use of a phrase dictionary or phrase recognition rules (Evans & Zhai 1996), term proximity (Tao & Zhai 2007) (Zhao & Yun 2009), terms dependencies (Gao et al. 2004) (Turtle & Croft 1991) (Metzler & Croft 2007), and so on. We will provide a more detailed description of these methods in Chapter 2. Each of the methods can bring some improvement in retrieval effectiveness (i.e. the quality of the ranked list) compared to a bag-of-words approach. However, strong assumptions on the type of dependency between terms are usually assumed. For example, the approaches based on term proximity assume that if two query terms appear closely in a document, then its matching score should be increased. No special attention is paid on whether the closeness of occurrences of the two terms is necessary and useful for identifying relevant documents. In some cases, for example for the query "using database in commerce", it is unnecessary that two terms "database" and "commerce" should appear closely in relevant documents – the word "commerce" could appear at the beginning of a document to specify the commerce area, while "database" appears at several different places in the document to describe the technical details. Imposing proximity between the words will penalize unduly the document. We observe the same limitation for the other approaches, which use other assumptions.

In reality, dependencies between words vary from query to query. Two adjacent words in a query can be strongly dependent in a query, while completely independent in another. The usefulness of incorporating a dependency into the retrieval function also varies from query to query. Even if a strong semantic dependency is observed in a query, it may not always be the case that we have to favor documents in which the dependent words are connected. For example, it is quite obvious that there is a strong semantic dependency between words in the query "death due to cancer" (a query used in TREC experiments).

However, it is found that requiring “death” to appear closely to “cancer” does not improve the retrieval result; rather it hurts it[1]. As a matter of fact, when term proximity is imposed as a retrieval criterion, we obtain lower effectiveness than the bag-of-words approach. This example shows that an intuitively strong dependency may not be helpful for IR.

In summary, a simple assumption of term dependency and its usefulness to IR may not reflect the complex nature of them. Dependencies are variable, so is their usefulness in IR. It is the problem that we will address in this thesis: to cope with the variable dependencies that may exist between query terms.

1.2 Our Approaches: Relating Dependent Terms

In the thesis, we focus on relating terms in the representations of documents and queries and in IR models (i.e. matching functions). We assume that words (terms) in a query may be dependent (thus different from a bag-of-words approach) in some way. From a linguistic point of view, two words may be grammatically, e.g. a noun may depend on a verb. They can be semantically dependent as in “death due to cancer”, in which “cancer” is a *cause* of death.

In the history of IR, there have been a number of attempts to incorporate grammatical and semantic dependencies (Lafferty, Sleator & Temperle 1992), (Rio 2009). For semantic dependencies, the most critical aspect is the difficulty to determine the precise semantic relation between words. Up to now, there is still no reliable tool capable of determine such relations in large domains. The existing tools can usually determine a small number of semantic relations in limited domains (Salton, Yang & Yu 1975), (Miller 1995). The use of such tools for IR is premature.

[1] It is easy to understand the problem by imagining a potential relevant document for the query. In such a document, one can talk about the problems of cancers, including a paragraph providing statistics of death. However, words "death" and "cancer" may appear at some distance (i.e. not at proximity).

Even if we want to know the exact semantic relation between words in the ideal case, in many applications such as IR, we do not need to do it. In most cases, it suffices that we know if two words in a query are dependent. If they are, one can favor documents in which the two words appear to be in relation. In other words, we do not need to explicitly create a semantic interpretation for the dependent words, but merely to determine the likelihood that two words may be dependent. To do this, the use of grammatical dependencies may seem a reasonable approach. If a grammatical dependency is detected between a pair of words, one may assume that there is a possible semantic dependency between them. Therefore, it could be required that the same (or similar) dependency appears in a retrieved document.

However, the above approach has not been successful in IR. In an early study, (Fagan 1987) showed that it is not effective to use noun phrases in IR. In other words, the grammatical dependencies within noun phrases do not bring any significant improvement in retrieval effectiveness, compared to a bag-of-words approach. On the other hand, he showed that statistical dependencies (or phrases) can improve retrieval effectiveness. By statistical dependency, we mean groups of words that appear together often in the collection. This type of "phrase" could be a correct noun phrase (such as "computer science"), but can also be ungrammatical group of words. For example, "Xbox NBA" extracted from "Xbox NBA game sale" is not a grammatically correct phrase, but it can be useful for IR. The reason of this is that a document containing the "phrase" (even ungrammatical) has a higher chance to be relevant to the query. For example, it may contain "Xbox NBA download", which is a sign that the document can be highly relevant. This example shows that the relevance of a document does not require the words in a query to be connected by the same grammatical relation in the document as in the query.

Statistical dependencies focus on frequent co-occurrences of terms. If two terms co-occur often in a collection (or in a language), there is a chance that they form a well-defined concept together, and should be considered as a phrase. This type of dependency, compared to linguistically motivated ones, has the advantage that it has a broader coverage (all possible co-occurrences are candidates), less prone to errors of the tools, and requires less complex processing. Its effectiveness has also been proven in IR. As a matter of fact, most recent dependency-based retrieval models rely on statistical dependencies. Therefore, we will limit ourselves to statistical dependencies in this thesis. Throughout this thesis, we will use *dependency* to mean a certain statistical relationship between terms. In particular, we will consider the two following dependencies:

- The dependency between words in a fixed expression (such as "black Monday"), which require the words to appear together and in the same order. We will call it *bigram* dependency. Note that one can extend this type of dependency to longer n-grams, but this is at the cost of a much higher complexity. So, our investigation will be limited mainly to bigrams.
- The proximity dependency. It requires the terms to appear at proximity, i.e. within a small text window. This type of dependency covers a much wider range of relations such as variants of expression ("house construction" vs. "construction of houses") or contextual dependency (e.g. between "program" and "Java" in "a program for sorting words in Java"). We will call this type of dependency *co-occurrence* dependency.

These two types of dependencies cover most of the attempts in IR. Our assumption in this thesis is that if two terms *a* and *b* appear in a query, then we have three situations:

1. *a* and *b* are not dependent, and they can be used in a bag of words.
2. *a* and *b* are strongly dependent and they form a fixed expression. In this case, they should be considered as a bigram.
3. *a* and *b* has a loose contextual dependency that require them to appear at proximity.

Each of the cases creates a different type of index. Case 1 corresponds to the traditional bag-of-words index. Case 2 uses word bigrams (or n-grams) as indexing units. Case 3 uses free groups of 2 words as indexing units.

The problem we investigate is to define an appropriate retrieval model to capture the three above cases. We will describe three different ways to do it, which differ on the following aspects:

- whether to create separate indexes or a unique index for different types of units,
- whether to determine the strength of a dependency according to the query.

The research path we will describe in this thesis is as follows:

Combining different indexes

We first assume that different types of indexing units have been extracted from a document and a query (e.g. words and phrases), and attempt to use different indexes to produce a combined ranking function. The idea of this attempt is to see if a document and a query can be represented in multiple ways, and a ranking score based on multiple representations is better than using a single representation.

We will show that this is the case. The experiments will be carried out on Chinese. This choice is made because of the more critical aspect of indexing units in Chinese. In most European languages, it is shown that a bag-of-words provides a decent level of effectiveness (even though it is improvable). The use of more complex units such as phrase is often perceived as an optional add-on. The situation in Chinese is different: Chinese text is a sequence of characters and it has no a native notion of words. Words have to be determined by an automatic segmenter, which produces one, but not the unique, sequence of words for a sentence. There is an acute need to take into account words as well as the constituent characters. So, to combine different indexing units (segmented word and n-gram characters) in Chinese language is more important for overcoming the word segmentation problem.

A representation incorporating dependent units

In our second approach, rather than creating different indexes using different types of indexing units, we create a single representation for document and for query, which integrate different types of indexing units. In the representation, the indexing units are dependent. For example, a phrase is considered dependent on its constituent words. This approach is more principled than the former. We use Dempster-Shafer theory of evidence as the basis of our representation: A phrase and its constituent words are grouped into the same set of elements within which some dependencies are assumed. For example, when an occurrence of *computer architecture* is observed, we consider it as representing three possible terms: *computer*, *architecture* and *computer-architecture*, which form a set. Probability is assigned to the whole term set instead of to each term. This solves an important problem of probability assignment when they are considered independent: the probability mass assigned to *computer* should overlap with that of *computer-architecture.* To determine the score of a document facing a query, we will consider the possible relations between a term set of the document and a term set of the query. Several transfer function are defined to estimate the match between term sets.

Modeling variable term dependences according to their utilities in IR

The question that remains unanswered in the second approach is how to determine the dependency between different elements in a representation. In the second approach, we use heuristics to define it. In our third approach, we explicitly incorporate different types of dependencies, and we measure their strength according to their potential contribution to retrieve relevant documents.

Another problem with Dempster-Shafer model is that it can only capture relations of terms within a term set, and does not allow terms in different sets to be dependent. For example the word sequence a b c d e are grouped into two term sets: $(a\ b\ c)^*$ and $(d\ e)^*$, the model cannot capture the relations between a and d or between c and d etc., which could be useful for IR (contextual dependencies).

We propose a more flexible approach in which we explicitly capture two types of dependencies: bigram dependency and co-occurrence dependency. In addition, we assume that a specific dependency between a pair of terms may have a degree of contribution to the matching function, depending on the query and on the terms in the dependency. Therefore, we use a machine learning approach (regression) to learn the weigh importance of a dependency based on a set of features.

This final approach has been tested on both English and Chinese collections.

The three approaches are described in the three following chapters, which are composed of published papers and a submission to a journal (under way):

- Chapter 3: Using Unigram and Bigram Language Models for Monolingual and Cross-language IR, *InfoScale* (Shi, Nie & Bai 2007)
- Chapter 4: Relating Dependent Indexes Using Dempster-Shafer Theory, *CIKM* (Shi, Nie & Cao 2008).
- Chapter 5: Using Various Term Dependencies According to Their Utilities, *CIKM* (Shi & Nie 2010A), Modeling Variable Dependencies between Characters in Chinese Information Retrieval, *AIRS* (Shi & Nie 2010B), Coping with Different Types of Term Dependencies in Information Retrieval (under way).

1.3 Contributions

In this thesis, the central problem we address is the representation of documents and queries beyond bag of words. We consider that words in a document and a query can be dependent, which requires them to be used connected in some way in the matching process. This is a central problem in IR. In this thesis, we propose a series of approaches, which brings some original solutions to the problem. The contributions of this thesis are as follows:

- We show that an IR system that combines with different indexes works better than with a single index.
- We propose an integrated representation based on Dempster-Shafer theory of evidence that includes different types of indexing units. This is a new type of document representation.
- We define an IR model in which dependencies are explicitly represented and weighted according to their possible contribution to the retrieval process. This approach is a significant extension of the existing retrieval models.

This series of approaches and experiments will clearly show the importance of taking term dependencies into account. They also show that dependencies should not be used in a uniform way, rather they should be incorporated into the retrieval process according to their possible impact on the latter.

1.4 Organization of the Thesis

This is a thesis composed by articles. Therefore, the three chapters that describe the proposed approaches form the main part of the thesis. To make the thesis understandable for people not familiar to IR, we will provide some introductory material. In particular, before the three chapters, we will briefly describe the area of IR and the main approaches used currently.

Although we will include the articles in the same form as they are, we will add some introduction before and some discussion after it.

Finally, a chapter of conclusion will contain discussions on the series of approaches and experiments presented in the thesis.

The thesis is organized as follows:

In the Chapter 2, we will first introduce the basic concepts and processing of information retrieval; then, we describe the traditional IR models and related works.

In Chapter 3, we will define and test a simple way to combine different indexes in IR. The experiments are conducted on Chinese monolingual IR and cross-language IR between English-Chinese.

In Chapter 4, we relate dependent indexing units using Dempster-Shafer theory.

In Chapter 5, we propose a model based on discriminative model framework to integrate different types of term dependencies and to weight them according to their potential impact on retrieval effectiveness.

The general discussions and finally conclusions will be presented in Chapter 6.

CHAPTER 2.
STATE-OF-THE-ART OF IR

Information retrieval can be defined as follows (Manning & Schütze 1999):

Information retrieval (IR) is finding material (usually documents) of an unstructured nature (usually text) that satisfies an information need from within large collections (usually stored on computers).

To enable the system finding relevant materials (documents), we have to create a set of basic operations on documents and queries, and to define a matching score between them. In this section, we will first define the basic operations needed in all IR systems. Some evaluation measures used in the area will be defined to quantify the quality of a system. Then a set of retrieval models related to our work will be described.

2.1 The Basic Processing of IR

A typical IR system is shown in Figure 2-1. The main processes aim to create the representation of the document, the representation of the query, and the definition of a matching score between them.

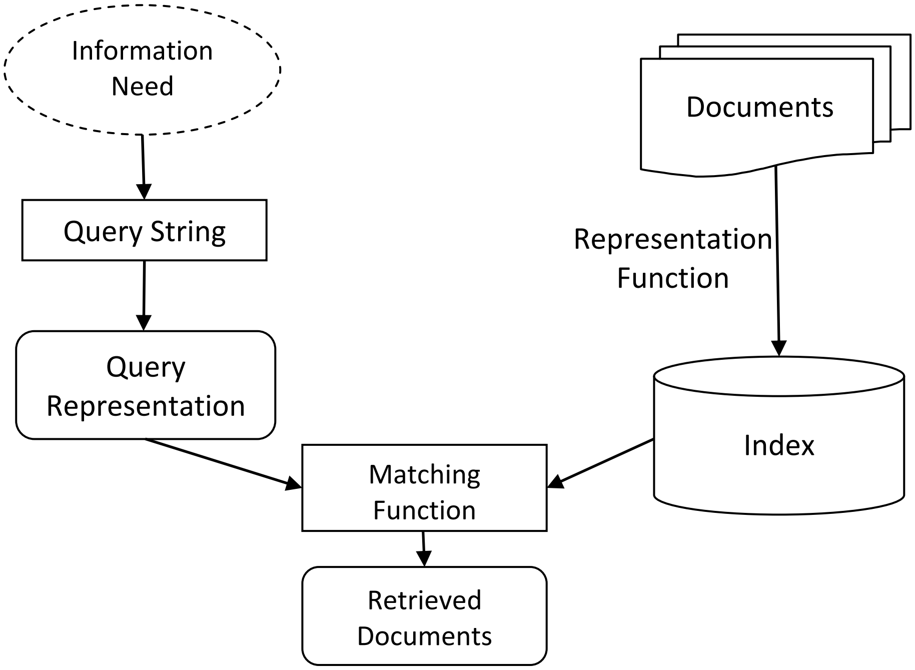

Figure 2-1. Information retrieval processes

The document representation is called *indexing* process. In this process, we need to determine the indexing units first, which are words (or word stems) for most European languages. For some East-Asian languages such as Chinese and Japanese, the indexing units could be segmented words or character n-grams (we will see more details on the processing of Chinese texts in Chapter 3).

A document written in a natural language has to undergo the following processes:

- Removing common terms (called stopwords) that do not bear specific meaning. These words usually correspond to functional words in a language, such as prepositions, articles, etc.
- Word stemming or lemmatization. The goal of this step is to discard the small differences in word form that do not change much the meaning. For example, the plural and singular forms of a word have the same meaning. Stemming removes some suffixes of words and only the word *stem* is kept. For example, the words

"information", "informing", "informed" can be transformed to the stem "inform". Two stemming algorithms are widely used in IR: Porter stemming (Porter 1980) and Krovetz stemming (Krovetz 1993). Lemmatization tries to convert a word form to its citation form (Hajič & Hladká 1998), (Kanis & Müller 2005). For example, the word "informed" and "informing" are transformed to their citation verb form "inform", while "information" to its root form "information". This requires that the grammatical category of the word to be known, thus a part-of-speech (POS) tagging. In practice, no clear difference is observed between the two processes. Morphological analysis only produces at most very modest benefits for retrieval (Manning, Raghavan & Schute 2008). So the simpler stemming is usually used in IR. A processed word is called *term* in IR.

- The last step for document processing is to create an index. Once a set of terms has been identified in the previous processes, one can create a set of terms to represent the document. For the sake of retrieval efficiency, one usually creates an *inverted index*, which maps a term to a set of documents that contain it. Using the inverted index, the retrieval operation with a query can be implemented as finding the corresponding sets of documents, then merging them.

Document indexing also involves term weighting – to associate a weight to each term in the index. The weight of a term will influence the matching score. The best known weighting schema in IR is $tf\text{-}idf$ weighting (see Section 2.3.1). As term weighting is dependent on the retrieval model used, we will describe it in IR models.

Once a matching score is obtained for each document, with respect to a query, the documents are ranked in the reverse order of the score and presented to the user (e.g. organized in pages of 10 results in search engines).

2.2 Information Retrieval System Evaluation

A crucial problem in IR is to know if the ranked list of documents correspond to what the user is looking for. We need to define some measures to reflect the quality of a retrieval system or method.

Usually two aspects are used to compare IR systems: 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. In most cases, we focus on effectiveness in IR, as this is the aspect the most difficult to improve.

To evaluate an IR system in effectiveness, we need a test collection on which the system is run:

- a document collection;
- a set of retrieval queries; and
- the relevance judgments, telling if (and possibly how much) a document is relevant to a query.

The basic evaluation measures are precision and recall, defined as follows:

- Precision (P) is the fraction of retrieved documents that are relevant:
$$Precision = \frac{\#(relevant\ items\ retrieved)}{\#(retrieved\ items)} = P(relevant|retrieved)$$
- Recall (R) is the fraction of relevant documents that are retrieved:
$$Recall = \frac{\#(relevant\ items\ retrieved)}{\#(relevant\ items)} = P(retrieved|relevant)$$

A good IR system should have both high precision and recall. However, the system with high precision usually has low precision, while a system with high recall usually has low precision. A single measure F-measure is defined to trade off precision versus recall:

$$F = \frac{(\beta^2 + 1)P \cdot R}{\beta^2 P + R}$$

when $\beta = 1$, we have the following common form of F_1 measure:

$$F_1 = F_{\beta=1} = \frac{2\,P \cdot R}{P + R}$$

As the result of an IR system is a ranking list, we do not have fixed set of retrieved documents (thus fixed precision and recall). To evaluate a ranked list, we use a form of average precision.

11-Point Average Precision (Salton & McGill 1983)

The idea is to go through the ranked documents one by one. At each point, the documents included up to that point are used to obtain a value of precision and recall. We then have a set of points that define a curve of precision and recall. We then determine the corresponding precisions at 11 recall points: 0.0, 0.1, …, 1.0. To do this, some interpolations are required (as the determined points are not necessarily on these recall values). At the end, the average of the 11 precision values is calculated.

Precision@N and Mean Average Precision (MAP) (Buckley & Voorhees 2000), (Kraaij, Nie & Simard 2003)

Precision@*N* corresponds to precision at ranking point *N*. This is often used to reflect the quality of an IR system among the top results, for example precision@5, precision@10. Average Precision (AP, also called un-interpolated average precision) is calculated by averaging the precision at each point of retrieved document rank.

For a set of queries, we use the mean of the average precision scores for each query. It is defined as follows, where M is the number of test queries, N_j the number of relevant documents for the query j, and $r_{i,j}$ is the total relevant items in top i retrieval of query j:

$$MAP = \frac{1}{M}\sum_{j=1}^{M}\frac{1}{N_j}\sum_{i=1}^{N_j}\frac{r_{i,j}}{i}$$

MAP is calculated using all the points in the ranking list, where we find a relevant document. The average of the precision at all these points is calculated. This is the most common measure used in TREC experiments, in which binary relevance judgments are made (relevant or irrelevant). This will be our main evaluation measure in this thesis.

DCG (discounted cumulative gain) and NDCG (normalized DCG) (Järvelin & Kekäläinen 2000) (Järvelin & Kekäläinen 2002)

When graded relevant judgments are given, for example, 4 (perfect), 3 (very good), 2 (good), 1 (fair) and 0 (bad), we use DCG (Discounted Cumulative Gain) or its normalized form $NDCG$ to evaluate the result. This measure is commonly used in search engines. The basic ideal of DCG is:

(1) highly relevant documents are more valuable
(2) the greater is the ranked position of a relevant document, the less valuable is it for the user

Given a set of relevant judgment values $rel_1, rel_2, \ldots, rel_N$ at different rank positions $1, 2, \ldots, N$, DCG is calculated as:

$$DCG = rel_1 + \sum_{i=2}^{N}\frac{rel_i}{log_2\, i}$$

Usually, we do not consider the full list of ranked documents and only consider the top n documents. Then the $DCG@n$ only considers the relevant documents among the top-n results.

$NDCG$ (Normalized DCG) is a normalized measure using the DCG of the ideal ranked list (i.e. the best ranking list achievable for the query) as the normalization factor, i.e. $NDCG@n = DCG@n \,/\, Ideal - DCG@n$.

2.3 Information Retrieval Models

A large number of retrieval models have been developed in the literature: Boolean models (Lancaster & Fayen 1973) (Baeza-Yates & Ribeiro-Neto 1999) (Kraft & Buell 1983), vector space models (Salton, Wong & Yang 1975) (Salton & McGill 1983), probabilistic models (Robertson & Sparck Jones 1976) (Salton, Fox & Wu 1983) (Turtle & Croft 1990) (Robertson & Walker 1994) (de Campos, Fernández-Luna & Huete 2000), language models (Ponte & Croft 1998) (Miller, Leek & Schwartz 1999) (Song & Croft 1999) (Zhai & Lafferty 2001), and the variations based on above models. We do not intend to provide a complete description of all these models in this section. Rather we will describe only vector space model (which is widely used in IR), language model (which are the basic models we use in our work), Markov random field model and proximity model (which are as our baseline models), and few recent dependency models.

2.3.1 **Vector Space Model**

The vector space model (VSM) (Salton, Wong & Yang 1975) (Salton & McGill 1983) was first introduced and used in SMART system by G. Salton in late 1960s (Salton & Lesk 1965). It is the most popular and widely used model in information retrieval. In VSM, the query and documents are presented as vectors $\vec{Q}$ and $\overrightarrow{D_j}$, where the $\vec{Q} = \{w_{1,q}, w_{2,q}, \dots, w_{t,q}\}$, $\vec{D_j} = \{w_{1,j}, w_{2,j}, \dots, w_{t,j}\}$, and $w_{i,j}$ is the term weight. The degree of the document D_j related to the query Q is as the correlation between the vectors $\vec{D_j}$ and $\vec{Q}$, such as the angle formed by the vectors as:

$$Score(D_j, Q) = cos\theta = \frac{\vec{D_j} \cdot \vec{Q}}{|\vec{D_j}||\vec{Q}|} = \frac{\sum_i w_{i,j} w_{i,q}}{\sqrt{\sum_i w_{i,j}^2} \sqrt{\sum_i w_{i,q}^2}}$$

There are many term weighting schemes (Manning, Raghavan & Schute 2008) to measure the term importance in the vectors. The most common and effective one is tf-idf weighting, i.e. $w_{ij} = tf_{i,j} \cdot idf_i$, where $tf_{i,j} = c(t_i; D_j)$ is term frequency of term t_i in document j and $idf_i = \log \frac{N}{df_{t_i}}$ is inverse document frequency of term t_i in the whole collection. df_{t_i} is the number of documents which include the term t_i and N is the total number of documents in the collection. Both tf and idf have many variants in calculation, such as $tf_{i,j} = 0.5 + 0.5 \frac{c(t_i;D_j)}{\max_t c(t;D_j)}$, $tf_{i,j} = \sqrt{c(t_i; D_j)}$ and $idf_i = max\left(0, \frac{log\left(N - df_{t_i}\right)}{df_{t_i}}\right)$, $idf\_i = 1 + \log(\frac{N}{1+df_{t_i}})$.

2.3.2 **Basic Language Models**

A statistical language model (LM) is a probability distribution over a sequence of words that attempts to reflect how this sequence occurs as a sentence of a natural language. LMs have been successfully used in many fields of natural language processing such as speech recognition, machine translation, handwriting recognition, as well as information retrieval. The language modeling approach for IR was first introduced by (Ponte & Croft 1998) and successfully applied to many information retrieval problems (Miller, Leek & Schwartz 1999), (Song & Croft 1999), (Zhai & Lafferty 2001), (Bai et al. 2005).

The basic language modeling approach builds a probabilistic language model from each document D, and documents ranking based on the probability of the model generating the query. It also called query likelihood scoring method.

$$Score(Q,D) = P(Q|\theta_D) = \prod_{i=1}^{m} P(q_i|\theta_D)$$

Another approach of using LM in IR is that we can make a language model from both the document and query, and then ranking the documents according to the difference of these two language models. The Kullback-Leibler (KL) divergence method is commonly used for measure this difference:

$$Score(Q,D) = KL(\theta_D||\theta_Q) = \sum_{t\in V} P(t|\theta_D) \cdot \log \frac{P(t|\theta_Q)}{P(t|\theta_D)}$$

In both above language models, smoothing play a very important role. The unsmoothed model is the maximum likelihood estimate by relative counts as

$$P_{ML}(t|\theta_D) = \frac{count(t;D)}{\sum_{t'\in V} count(t';D)}$$

where V is the set of all terms in the vocabulary. A term which does not occur in a document will be assigned zero probability. Consequently, all documents which contain only partial query terms will get the equal result as zero.

Smoothing is the technique which adjusts of the maximum likelihood estimator by taking off some probabilities from presented words and assigning it (a small probability) to the absent terms. Not only the smoothing methods generally prevent zero probabilities, but they also attempt to improve the accuracy of the model as a whole. As listed in (Chen & Goodman 1999), numerous smoothing algorithms are studied in many nature language processing tasks such as Jelinek-Mercer smoothing, Katz smoothing, Witten-Bell smoothing, Absolute discounting, Church-Gale smoothing, Dirichlet prior smoothing etc.

Recent studies of smoothing methods in information retrieval show that smoothing play a very important role for IR and the retrieval performance is highly sensitive to the setting of smoothing parameters. The following three smoothing methods are commonly used for IR, which generally perform well (Zhai & Lafferty 2001):

- Absolution discounting:

$$P_{abs}(t|\theta_D) = \frac{\max(c(t;D) - \delta, 0)}{|D|} + \delta \frac{|D|_{uniq}}{|D|} P_{ML}(t|\theta_{Coll})$$

- Jelinek-Mercer:

$$P_{JM}(t|\theta_D) = (1 - \lambda)P_{ML}(t|\theta_D) + \lambda P_{ML}(t|\theta_{Coll})$$

- Dirichlet smoothing:

$$P_{Dir}(t|\theta_D) = \frac{c(t;D) + \mu P_{ML}(t|\theta_{Coll})}{|D| + \mu}$$

where θ_{Coll} is collection model, $c(t;D)$ is the number of term t in document, $|D|$ is the total number of terms in document, $|D|_{uniq}$ is the number of unique term in document, and δ, λ, μ is the empirical parameters of absolution discounting, Jelinek-Mercer, and Dirichlet smoothing respectively.

2.3.3 **Proximity Models**

Both vector space model and unigram language model assume terms are statistically independent, as well as the order in which the terms appear in the document is lost. Recently, some studies have been conducted to capture the terms dependence. One of approaches is using proximity which represents the closeness or compactness of the query terms appearing in a document. The studies (Hawking & Thistlewaite 1995), (Rasolofo & Savoy 2003), (Tao & Zhai 2007), (Zhao & Yun 2009) already show incorporating proximity factor can improve the effectiveness and performance of IR.

Basically, two approaches are used to measure the proximity: span-based and pairwise-based proximity distance measures. The proximity factors are combined with or integrated into the general IR models such as Okapi BM25 (Robertson & Walker 1994) and LM (Ponte & Croft 1998). The span-based approach measures the length of shortest document segment that covers all the query terms. While, the pairwise-based approach defines the pairwise distances between individual term occurrences, and then aggregates them to an overall proximity value.

Tao and Zhai (Tao & Zhai 2007) compared several proximity measures including span-base measure, minimum pair distance $MinDist = \min_{q_1,q_2 \in Q \cap D} Dis(q_1, q_2; D)$, average pair distance $AvdDist = \frac{2}{n(n-1)} \sum_{q_1,q_2 \in Q \cap D} Dis(q_1, q_2; D)$, and maximum pair distance $MaxDist = \max_{q_1,q_2 \in Q \cap D} Dis(q_1, q_2; D)$. It shows the $MinDist$ measure performs best and the proximity model significantly improves the retrieval performance over LM and Okapi BM25.

Zhao and Yun (Zhao & Yun 2009) proposed a proximity language model which integrated the proximity into the KL divergence language modeling framework based on Dirichlet prior smoothing. The document model is defined as:

$$P_{prox}(t|Q, \theta_D) = \frac{c(t; D) + \lambda Prox(t, Q) + \mu P_{ML}(t|\theta_{Coll})}{|D| + \sum_{i=1}^{|V|} \lambda Prox(t_i, Q) + \mu}$$

where λ is proximity weight parameter and $Prox(t, Q)$ is term proximity defined according to pairwise distance measures. Their result shows an empirical better performance than the basic language model and combination approaches of using proximity. In their results, the proximity based on summed of pair distance ($P\_SumProx$) and on minimum pair distance ($P\_MinProx$) performs much better than on average pair distance ($P\_AvgProx$), and the $P\_SumProx$ performs a little better than $P\_MinProx$.

This is one of our baseline models in this thesis.

2.3.4 **Markov Random Field Models**

Markov Random Field models are developed to extend the classical language models so that some term dependencies can be taken into account. As we will see in more details in this section, these models integrate three components: the traditional unigram model, a model considering the sequence of words, and a model considering the co-occurrences of terms within a text window. By adding the two latter components, the models are capable of strengthening the matching score of a document if it contains the same expression (sequence of words) as in the query, or if the query terms appear at proximity.

A Markov random field, also called Markov network or directed graphical model, is a graphical model in which a set of random variables have a Markov property described by an undirected graph. Nodes in the graph represent random variables, and edges define dependencies between the random variables. Formally, a Markov network consists of:

- An undirected graph $G = (V, E)$, where each vertex $v \in V$represents a random variable in V and each edge $\{u, v\} \in E$ represents a dependency between the random variables *u* and *v*.
- A set of functions f_k (also called *factors* or *clique factors* and sometimes *features*), where each f_k has the domain of some clique (or subclique) *k* in *G*. Each f_k is a mapping from possible joint assignments (to the elements of *k*) to non-negative real number.

The joint distribution (or Gibbs measure) represented by a Markov network is given by:

$$P(X = x) = \frac{1}{Z} \prod_{k} f_k(x_{\{k\}})$$

where $x\{k\}$ is the state of the random variables in the k-th clique, and the product runs over all the cliques in the graph. Here, Z is the partition function, so that

$$Z = \sum_{x \in X} \prod_{k} f_k(x_{\{k\}})$$

In practice, a Markov network is often conveniently expressed as a *log-linear model* by means of introducing feature functions ϕ_k, given by

$$f_k = \exp\left(w_k \phi_k\left(x_{\{k\}}\right)\right)$$

so that

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_{k} w_k \phi_k\left(x_{\{k\}}\right)\right)$$

Metzler and Croft (Metzler & Croft 2005) proposed a Markov random field model for IR. They try to capture term dependencies by integrating ordered and unordered term groups into the model. As shown in Figure 2-2, a graph *G* consists of query nodes q_i (each representing a term) and a document node *D*.

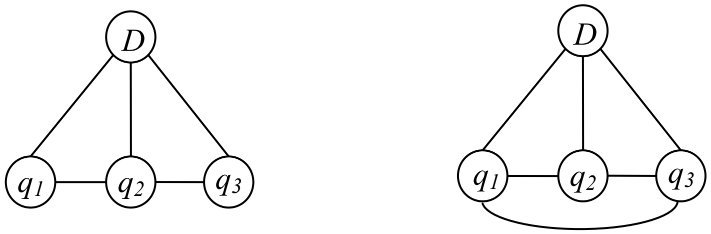

Figure 2-2. Markov Random Field models: Sequential Dependence Model (MRF-SD, left) and Full Dependence Model (MRF-FD, right)

They defined three types of potential functions:

- on clique of single terms, T (each clique contains a single term and the document *D*),
- on ordered term clique, O (a clique containing contiguous terms in Q and D), and

- on unordered term clique, U (a clique containing a non-contiguous set of query terms and D).

The ranking function is defined as following:

$$P(D|Q) \overset{\text{rank}}{=} \sum_{c \in C(G)} \lambda_c f(c) = \sum_{c \in T} \lambda_T f_T(c) + \sum_{c \in O} \lambda_O f_O(c) + \sum_{c \in U} \lambda_U f_U(c)$$

A MRF model requires the setting of three parameters: λ_c, λ_o and λ_U. It is usually done through cross validation – a set of judged queries is used to set the parameters, which are then used to test on new queries. In practice, it is shown that the setting of the parameter respectively at 0.85, 0.1 and 0.05 usually produce good results on different test collections.

MRF model is another baseline model we compare to in the thesis.

2.3.5 **Other Dependency Models**

Some new dependency models are proposed recent years (Park, Croft & Smith 2011), (Zhao, Huang & He 2011), (Bendersky & Croft 2012), (Hou et al. 2013), (Zhao & Huang 2014), (Zhao, Huang & Ye 2014). The methods of capturing the dependency vary from term proximity, concept hypergraph, information geometry to quasi-synchronous.

Although most successful attempts to consider term dependencies do not consider syntactic dependencies, there are some attempts trying to take advantage of syntactic structure. Park et al. (Park, Croft & Smith 2011) propose a term dependence model by using a quasi-synchronous stochastic process. Both query and documents are represented as syntactic dependency trees. The IR score function integrates a measure based on the distance between a query tree T_C and a document tree T_D. Four types of syntactic dependencies are considered: parent-child, ascendant-descendant, siblings, and c-commanding. Their final model is a linear interpolated model of the quasi-synchronous model and the sequential dependence Markov Random Field model – SDM (Metzler & Croft 2005). Their results show that the syntactic tree structure can significantly improve over the baseline SDM.

In addition to the dependencies between a pair of terms (sometimes called first-order dependencies), researchers have also been interested in using higher-order dependencies, i.e. more complex dependencies among more than 2 terms.

Bendersky and Croft (Bendersky & Croft 2012) propose a representation of dependencies using hypergraph. A vertex in a query hypergraph corresponds to an individual query concept, and a dependency between a subset of these vertices is modeled through a hyperedge. The importance of a concept is determined by features derived from frequencies in collections. There experimental results show that for verbose natural language queries (description field of the TREC topics), the proposed retrieval framework significantly improves the retrieval effectiveness of several state-of-the-art retrieval methods.

Hou et al. (Hou et al. 2013) proposed another approach to cope with pure high-order dependencies using information geometry. Pure high-order dependencies are those that cannot be reduced to first-order dependencies. These high-order dependencies (a set of terms) are incorporated into the $MRF\text{-}FD$ model (Metzler & Croft 2005). Their experimental results show that the orders of dependencies of 2 and 3 are main contributors to the improvement over the unigram model. The order n greater than 3 do not bring benefit to the IR system.

Zhao, Huang and He (Zhao, Huang & He 2011) (Zhao, Huang & Ye 2014) proposed a different way, called Cross Term model (CRTER), to cope with term dependencies. They consider dependent terms to form a Cross Term. The more the terms appear close to each other, the stronger the cross term is weighted (according to several decaying functions such as Gaussian function, triangle function, etc.). Such weight is incorporated into the traditional $BM25$ weighting scheme, by considering a cross term as a new type of term.

The final score function is a combination between the traditional $BM25$ score of words and the new $BM25$ score based on cross-terms. When cross-terms of size 2 are used, this corresponds to:

$$CRTER(D) = (1-\lambda)\sum_{i} w(q_i, D) + \lambda \sum_{i,j} w_2(q_{i,j}, D)$$

where $q_{i,j}$ is a cross-term, w is the score by a traditional model (e.g. $BM25$) and w_2 is that with cross-terms, and λ a combination parameter. In addition to cross-terms of size 2 (i.e. first-order dependencies), N-gram Cross-Term $q_{i_1,i_2,\ldots,i_n}$ (i.e. cross-terms of larger size) is also considered in (Zhao, Huang & Ye 2014). Such a cross-term is weighted by a distance metric and a kernel function as: $Kernel(\frac{1}{2}dist\left(pos_{k_1,i_i}, pos_{k_2,i_2}, \ldots pos_{k_n,i_n}\right))$. The final model is defined recursively as below:

$$CRTER_n(D) = (1-\lambda_n)CRTEM_{n-1}(D) + \lambda_n w_n(Crter_n, D)$$

$$CRTER_2(D) = (1-\lambda_2)w(term, D) + \lambda_2 w_2(Crter_2, D)$$

where $\lambda_2 \ldots \lambda_n$are combination parameters (between 0 to 0.2 in their experiments).

In their implementation, the score functions w and w_i are either based on BM25 or language models. Their experiments show that $CETER_2^{BM25}$ with BM25 weighting produces significant improvements over the traditional $BM25$ model with words, and is comparable to the state-of-the-art probabilistic proximity approaches; the $CETER_2^{LM}$ leads to an improvement over basic Dirichlet LM (in some collections are significant) and is comparable to MRF model (Metzler & Croft 2005) and PLM model (Lv & Zhai 2009).

The $CRTER_n$ model, in particular $CRTEM_3$ (trigram cross-term model) improves $CRTER_2$ slightly in most cases, but the improvements of $CRTER_3$ over $CRTER_2$ is lower than the improvements of $CRTER_2$ over $BM25$. On the other hand, with the increase of n in $CRTER_n$ model, the computational complexity grows exponentially. Therefore, the practical gain using cross-terms of size larger than 2 is limited.

In (Zhao & Huang 2014), a weight of term proximity $f(q_i, q_j)$ is also incorporated:

$$RelProx(D) = (1 - \delta) \sum_i w(q_i, D) + \delta \cdot f(q_i, q_j) \sum_{i,j} w_2(q_{i,j}, D)$$

where $f\left(q_i, q_j\right) = \frac{1}{|topDoc|} \sum_{q_i, q_j} \sum_{D \in topDoc} Rel(q_i, q_j, D)$ is the average contextual relevance value of term proximity between q_i and q_j among the top ranked documents (pseudo relevant documents). This additional weighting captures the importance of the dependency between a pair of term, which goes into the same direction as our model presented in Chapter 5.

The above dependency models proposed new ways to incorporate term dependencies in IR models and extends the previous models to high-order term dependencies. The experiments generally confirm that the most useful dependencies for IR are first-order dependencies. Therefore, in our study, we will focus first-order dependencies only.

2.3.6 **Discussions**

As we can see through the description in this chapter, existing models either do not consider dependencies between terms, or consider them in a simplistic manner.

Proximity models assume that any pair of terms appearing in a query is required to appear closely. Although this requirement is reasonable for many queries, there are still many cases where two query terms do not have to appear in a relevant document. For example, for the query “download acrobat reader”, the word “download” can appear quite far away from “acrobat reader” in a relevant document[1] – it is often a “download” button

[1] An example is the official website of Acrobat downloading: http://get.adobe.com/reader/otherversions/

in a different sub-window. Considering the proximity of “download” and the two other words will penalize this document. On the other hand, in this same example, it is reasonable to consider the proximity between “acrobat” and “reader”. This example shows that the requirement to consider term similarity in a document varies according to the terms.

The MRF models have similar limitations, as dependencies are assumed uniformly between all terms in a query. The parameters that we generally use to combine the three components are set to maximize the average effectiveness on different queries. However, one can easily imagine that one query may require a strong emphasis on dependency components, while another query does not require them at all.

The above problem is what we aim to solve in this thesis (Chapter 5) – to capture various dependencies between terms and to incorporate them into a retrieval model according to their usefulness for IR.

In the following chapters, we will describe a series of attempts in this direction.

In Chapter 3, we describe a naïve integration of different types of indexing units and test the approach in Chinese IR. The goal is to show that if we rely on multiple types of indexes among which phrases are represented, we can reach higher retrieval effectiveness. We choose to test the approach on Chinese IR because the problem is more pervasive in Chinese than in other languages, as Chinese language lacking a set of standard words.

In Chapter 4, we attempt to integrate multiple types of indexing units in the same representation, in order to cope with the possible relations among them.

In Chapter 5, we describe an approach to extend the MRF models by incorporating a measure to assess the importance of a dependency for a query.

CHAPTER 3.
USING DIFFERENT UNITS FOR CHINESE MONOLINGUAL AND CROSS-LANGUAGE IR

Introduction to the chapter

In this chapter, we consider one way to capture the possible dependencies between terms. The basic idea is to create several indexes using different types of indexing units. For example, in English, if we use both single word and multiple-word phrase, then we can create two separate indexes, one for words and one for phrases. Two ranking scores for a document are determined using the different indexes. These scores are then combined to produce the final score.

This is a simple way to cope with term dependencies, namely the dependencies within a phrase are considered, as a phrase is considered as a single unit in the phrase index. We consider here IR in Chinese, in which we are faced with a big problem of "phrase", as Chinese is not written as separated words but as a continuous string of characters (or ideograms). A crucial problem is to decide what indexing units to use. The problem of coping with different indexing units in Chinese is more pervasive than in most European languages. In the latter, even if we do not consider phrases and only use words, we can usually obtain quite good results (the bag-of-words approaches are still considered the state-of-the-art). In Chinese, however, if we only use single characters, the result is not necessarily good. In this chapter, we use Chinese as the support language to show the necessity of combining different types of indexing units.

This chapter is a paper published at the *Proceedings of the 2nd International Conference on Scalable Information Systems:* (Shi, Nie & Bai 2007).

3.1 Introduction

Traditional information retrieval (IR) approaches usually assume that index terms are independent. Both documents and queries are represented by a set of independent terms. For example, “computer architecture” is represented by two independent terms – *computer* and *architecture*. It is obvious that in this simple representation, the strong dependency between terms vanished and the meaning of “computer architecture” is not represented precisely. The same assumption is also used for cross language information retrieval (CLIR). A more often used method of CLIR is using a statistical translation model (TM) for query translation. It trains a TM for each term from parallel corpus first. The query term is then represented by the top translation words in TM individually. While we do this kind of query translation, the dependency of query terms is lost.

For the Chinese language IR, this problem is more serious. A Chinese text consists of a sequence of Chinese characters without natural word boundaries. Although a single Chinese character has a meaning and can be a single-character word, more often it is combined with other characters in a multi-character word. Another way to capture the relation of Chinese characters is to using bigram or trigram. Therefore, in the current Chinese IR models, two general families of approaches have been proposed to cut Chinese text into indexing units: using characters (mainly character unigrams and bigrams) and using words, such as in (Chien 1995), (Liang, Lee & Yang 1996) and (Kwok 1997). Words and bigrams are representations of relating Chinese characters. When a Chinese text forms to words and bigrams, the relationship of component characters has been captured. Several studies have compared the effectiveness of these two types of indexing units in Chinese IR (Luk, Wong & Kwok 2002), (Nie, Brisebois & Ren 1996), (Nie et al. 2000). They all show that words and bigrams can achieve comparable performances, and have produced higher retrieval effectiveness than unigrams.

However, both words and bigrams may encounter the problems of segmentation ambiguity and failure of match the slight different word. For example, for the sequence 发展中国家 (developing country), it is well possible that it is segmented inconsistently into 发展 (development) 中 (middle) 国家 (country) or 发展 (development) 中国 (China) 家 (family), depending on the segmentation method used and the context. Moreover, two different words do not always have different meanings. They can be related, especially when the words share some common characters such as 办公室 (office) and 办公楼 (office building). To avoid these problems, unigrams are usually considered as the index. In this case, documents can match when they share characters with a query.

The previous studies have been carried out using different retrieval models: vector space model, probabilistic model, etc. No comparison has been made using language modeling (LM). In this study, we will re-examine the problem of indexing units for Chinese IR within the LM framework, and investigate the combination approach for different units.

For Chinese CLIR, only words have been used as translation units. No study has investigated the possibility of using n-grams of Chinese characters as translation units or their combination with words. The main focus in this chapter is to investigate the impact of using different Chinese units in CLIR. We will compare different approaches to query translation using different translation units.

Our experiments on several large (NTCIR and TREC) test collections will show that in both Chinese monolingual and cross-language IR, it is much better to combine words (bigrams) with unigrams. The combination mode benefits from both independent character model (unigram) and dependent character models (word and bigram). For CLIR, consider co-occurrence term in TM, we can get future improvement.

The remaining of the chapter is organized as follows. In Section 3.2, we will describe the background of our study. Some related work will be described. Section 3.3 will describe our approaches using different index and translation units for Chinese IR and CLIR. Section 3.4 describes the experimental setting and results. Conclusions and future work will be given in Section 3.5.

3.2 Background

3.2.1 Chinese Word Segmentation

Unlike most Indo-European languages, a Chinese text is written as a continuous sequence of Chinese characters without natural delimiters such as spaces. Therefore, before further linguistic analysis on a Chinese text, the text has to be first segmented into a sequence of words. The main difficulties of word segmentation are word boundaries ambiguities (a sentence can be segmented into several different sequences of legitimate words) and unknown words. Many segmentation approaches have been proposed and most of them are published a decade ago. Basically, they fall into the category of dictionary-based method, statistically-based method, or the hybrid of these two methods.

The dictionary-based approach uses a lexical dictionary and the greedy longest match algorithm to segment the text. This approach is simple and efficient. The segmentation quality often depends on the completeness of the dictionary. However, we are hardly supposed to have a truly complete Chinese dictionary. Some studies (Liang & Zhen 1991), (Yao, Zhang & Wu 1990) (Nie, Jin & Hannan 1994) try to improve the quality by adding a set of heuristic rules, such as rules to deal with numbers, dates and proper names. These rules are incorporated into the segmentation process to detect the out-of-vocabulary words (unknown words). However, the longest match algorithm cannot solve ambiguities. For example, the sentence "太阳能发光" (The Sun can shine) has two segmentations: 太阳能 (solar energy) / 发光 (shine) and 太阳(the Sun) / 能(can) / 发光

(shine). The algorithm always chooses the result with the longest words (the former), which is wrong in this case.

Statistical approaches do not require a dictionary. Instead, they need a great amount training data. Various statistical models are proposed for segmentation from n-gram language model (Teahan et al. 2000), hidden Markov model (Sproat & Shih 1990) (Zhang et al. 2003), maximum entropy model (Xue 2003), conditional random fields model (Peng, Feng & McCallum 2004) to self-supervised EM model (Peng & Schuurmans 2001) (Huang et al. 2003) and pragmatic mathematical framework model (Gao et al. 2005). Both supervised learning (Teahan et al. 2000) and unsupervised learning (Peng & Schuurmans 2001) have been used.

More specifically, given a sequence of Chinese characters $C = c_1c_2 \dots c_n$, we wish to segment the characters sequence into words $W = w_1w_2 \dots w_m$. There can be different ways to segment the sequence, corresponding to different word sequences $W^i = w_1^iw_2^i \dots w_{m_i}^i$. The goal of segmentation model is to find the most likely word sequence $\widehat{W}$ among all possible candidates W^i:

$$\widehat{W} = \underset{W^i}{\arg\max} P(W^i|C)$$

The statistical segmentation model in (Chiang et al. 1992) is define as

$$\begin{aligned} P\left(W^i|C\right) &= \prod_{k=1}^{m_i} P_i(l_k, w_k|l_1l_2..l_{k-1}, w_1w_2..w_{k-1}, m_i, C, n)P_i(m|C, n) \\ &\approx \prod_k P_i(w_k|l_{k-1})P_i(m|n) \end{aligned}$$

where l_k denotes the k-th possible word length.

In the hidden Markov model of (Zhang et al. 2003), classes of segmented words are introduced. We denote the sequence of classes by $T^i = t_1^it_2^i..t_{m_i}^i$, where t_k^i is a corresponding class of w_k^i (9 classes are defined according to lexicon). The final decision is made by:

$$\widehat{W} = \arg\max_{W^i} P\left(W^i, T^i \middle| C\right) = \arg\max_{W^i} P\left(W^i \middle| T^i\right) P\left(T^i\right)$$
$$= \arg\max_{W^i} \prod_{k=1}^{m_i} P(w_k|t_k) P(t_k|t_{k-1})$$

Peng, Feng and McCallum (Peng, Feng & McCallum 2004) defined a conditional random fields model for Chinese word segmentation. The probability $P(W^i|C)$ is defined as a set of feature functions:

$$P\left(W^i \middle| C\right) = \frac{1}{Z} \exp\left(\sum_{t=1}^{n} \sum_{k} \lambda_k f_k(w_{t-1,} w_t, C, t)\right)$$

where λ_k is the learning weight of feature f_k.

In the self-supervised EM model of (Huang et al. 2003), two lexicons are used: core lexicon V_1 (may be empty at the beginning) and candidate lexicon V_2 which contains all other candidate words that are not in the core lexicon. The probability distribution $\Theta = \{\theta_j | \theta_j = P(w_j), j = 1, \dots |V_1|\}$ and $\Phi = \{\phi_j | \phi_j = P(w_j), j = 1, \dots |V_2|\}$ are defined over above lexicons. Then, the segmentation becomes:

$$\widehat{W} = \arg\max_{W^i} P\left(W^i \middle| C; \Theta, \Phi\right) = \arg\max_{W^i} P(W^i, C|\Theta, \Phi)$$

The Joint likelihood is defined as:

$$P\left(W^i \middle| C|\Theta, \Phi\right) = \prod_{j_1=1}^{M_1} \lambda P\left(w_{j_1}\right) \prod_{j_2=1}^{M_2} (1-\lambda) P\left(w_{j_2}\right)$$

where $M_1$and $M_2$are number of words in V_1, and V_2 and λ is the weight of the core lexicon.

The probability distribution Θ and Φ are learned from training corpus by EM algorithm and update Q function is given by

$$Q(k, k+1) = \sum_{W} P(W|C; \Theta^{\mathrm{k}}, \Phi^{\mathrm{k}}) \log(\mathrm{P}(\mathrm{W}, \mathrm{C} | \Theta^{\mathrm{k+1}}, \Phi^{\mathrm{k+1}}))$$

First, run EM process on training C_1 until it stabilizes. Then, repeat forward selection (move M highest probability words from V_2 to V_1) and backward selection (move M lowest probability words from $V_1$to V_2) on validation corpus C_2 to get the best accuracy. This process is iterated until certain accuracy threshold is reached.

Previous studies on Chinese word segmentation showed that segmentation accuracy in Chinese is usually higher than 90% (Chen & Kiu 1992), (Li et al. 1991), (Yao, Zhang & Wu 1990). This accuracy is shown to be satisfactory for IR (Nie, Brisebois & Ren 1996). In addition, (Peng et al. 2002) and (Huang et al. 2003) showed a non-monotonic relationship between retrieval performance and segmentation accuracy. In the experiment of (Huang et al. 2003), a segmentation accuracy of 70%-80% can archive best IR performance and a higher segmentation accuracy leads to decreases in the IR performance. The reason is that the high accuracy segmentation may identify longer words, which are less useful than shorter words (Wu 2003) (Gao et al. 2005). Our experiment will also confirm this.

Therefore, in our study, we will not strive to increase Chinese word segmentation accuracy. We will simply choose a common word segmentation method, and we will use other means to improve Chinese IR effectiveness: Besides using word segmentation, we also use n-grams. This has been proven effective in previous studies (Huang et al. 2000).

Later in this Chapter, we will investigate how different types of Chinese indexing units can impact the IR performance and how they can be combined.

3.2.2 **Cut Chinese Text into Index Unit**

Chinese IR has been studied for more than one decade. The difference from IR in English and in Chinese lies in the fact that word boundaries are not marked in Chinese. In order to index a Chinese text, the latter has to be cut into indexing units. The simplest method is to use single characters (unigrams) or all adjacent overlapping character pairs (bigrams), such as in (Chien 1995), (Liang, Lee & Yang 1996). Another method is to segment Chinese sentences into words, as in (Kwok 1997).

A Chinese word is composed of one, two, or more Chinese characters. Nie et al. (Nie et al. 2000) shows that the average length of Chinese words is 1.59 characters. It means that most Chinese words have only one or two characters. So, by considering bigrams, most Chinese words can be correctly covered. Although some longer words cannot be represented accurately by bigrams, the extension from bigrams to longer n-grams has a cost: there will be much more n-grams to be stored as indexes, and the complexity both in space and retrieval time will increase substantially. Therefore, limiting n-grams to length 2 is a reasonable compromise. So, besides words, we will consider only unigrams and bigrams.

Using a word segmentation method, a sentence can be transformed into a sequence of words. Then the same word-based method used for European languages can also be used for Chinese. For example, the sentence “国企增加研发投资” (National enterprises increase the investment in R&D) can be segmented into: “国企 / 增加 / 研发 / 投资”.

However, this example also shows an important problem: the same meaning can be expressed in multiple ways. For example, 研发 （R&D）can be expressed as 研究和开发 (research and development). If only 研发 is used as index, then it will not be able to match against 研究和开发. This problem is similar to that of abbreviation in European languages (such as “R&D”). We argue here that the phenomenon is more frequent in Chinese. Very often new abbreviations are easily created. For example, 国营企业

(national enterprises) can be abbreviated to 国企 (as in our example). In addition, Chinese also has a large number of similar words to express the same meaning. For example, 增大, 猛增, 递增, 加大, etc. can all express the same (or a similar) meaning as (to) 增加 (increase). A strategy that only uses words as indexing units will very likely miss the corresponding words.

We notice in the above example of "increase" that many similar Chinese words share some common characters. Therefore, a natural extension to word-based indexing of documents and queries is to add characters as additional indexing units. By adding 国, 企, 增, 加, 研, 发 as additional indexes, we will create partial matches with other words expression "national enterprises", "increase" and "R&D", thereby increase recall. Although this approach is unable to cover all the alternative expressions, it has been shown to be effective for Chinese IR (Luk, Wong & Kwok 2002), (Nie et al. 2000).

An alternative to word segmentation is to cut a Chinese sentence into overlapping bigrams such as: 国企 / 企增 / 增加 / 加研 / 研发 / 发投 / 投资. Compared to word segmentation, this approach has the advantage that no linguistic resource (such as dictionary) is required. In addition, new words can be better represented. For example, suppose 新译林 is a new word (possibly the name of a magazine), which is not stored in the dictionary. Then it is likely segmented into three separate characters 新 / 译 / 林 using a word segmentation approach. If we use bigrams, the sequence 新译 / 译林 will be generated. These latter can better reflect the sequence 新译林 than the three separate characters.

A possible problem with bigrams is that many of them do not correspond to valid semantics. In the earlier example, 企增, 加研 and 发投 do not correspond to any valid meaning. However, it can be expected that their frequency of occurrences in documents will be much lower than the valid parts 国企, 增加, 研发 and 投资. Therefore, there is a natural selection of valid bigrams by the corpus statistics.

The above observation has been made in several previous studies (Luk, Wong & Kwok 2002), (Nie et al. 2000). However, words and bigrams have often been used as two competitive approaches instead of combining them. In (Nie et al. 2000), it is found that the most effective approach is to segment sentences into words but also add the characters. For example, the sequence 国企增加研发投资 is segmented into 国企 / 增加 / 研发 / 投资 / 国 / 企 / 增 / 加 / 研 / 发 / 投 / 资. The addition of single characters (or unigrams) allows us to extend the words to related ones.

Several studies have compared the effectiveness of these two types of indexing units in Chinese IR (Luk, Wong & Kwok 2002), (Nie, Brisebois & Ren 1996), (Nie et al. 2000). In this chapter, we will re-examine the problem of indexing units for Chinese IR within the LM framework, and investigate the combination approach for different units.

3.2.3 **Using Parallel Corpus for CLIR**

Cross-language information retrieval (CLIR) is becoming increasingly important due to the rapid development of the Web. As the query and the documents are written in different languages, the main problem of CLIR is the automatic translation between query and document languages. The basic approach is to translate the query from a source language to a target language. There are three main techniques for query translation: using a machine translation (MT) system, using a bilingual dictionary, and using a statistical model trained on parallel texts. It has been shown that when used correctly, these approaches can lead to comparable retrieval effectiveness (Gao et al. 2001), (Gao et al. 2002), (Jin & Chai 2005), (Kraaij, Nie & Simard 2003), (Nie et al. 1999). However, for CLIR involving Chinese, words are usually used as translation units. Although n-grams of characters have been found to be reasonable alternatives to words in indexing (Luk, Wong & Kwok 2002), (Nie, Brisebois & Ren 1996), no previous study has investigated the possibility of using Chinese character n-grams as translation units. In this study, we will investigate into this issue. Our investigation will make use of a parallel corpus.

Parallel texts are texts in one language accompanied by their translations in another language. Parallel corpora containing such texts have been used for CLIR in different manners.

A simple method is used in (Davis & Ogden 1997), (Yang et al. 1998): a source language query is first used to retrieve source language documents in the parallel corpus; then the parallel texts in target language corresponding to the top retrieval results are used to extract some target language words; these latter are considered as a "translation" of the query. This method works in a way similar to "pseudo-relevance feedback" in information retrieval.

A more often used method trains a statistical translation model (TM) from a parallel corpus. (Nie et al. 1999) is among the first ones to use this method for CLIR. They build a probabilistic translation model from a parallel corpus. The top translation words proposed by the TM are kept as the translation of a query. This study showed that the retrieval effectiveness obtained is very close to that using a good MT system (Systran). A series of other papers, such as (Gao et al. 2001), (Gao et al. 2002), (Jin & Chai 2005), follow the same direction to integrate TM to CLIR. In particular, (Kraaij, Nie & Simard 2003) has tested the integration of query translation into a global language model. They showed that this integrated approach outperforms the existing machine translation system (Systran).

A translation model is a mathematical model, which gives the conditional probability $P(T|S)$, i.e. the likelihood of translation a source language string S into a target language string T. Different TMs use different methods to align words between source and target languages. The main single-word-based alignment methods are IBM 1 to 5 (Brown et al. 1993) and Hidden-Markov alignment model (Vogel, Ney & Tillmann 1996). These models use words as the basic translation units. For Chinese, it is assumed that a sentence is segmented into words. Then the same approach can be used for Chinese. Word-based translation approach has been used in all the previous studies on Chinese translation using parallel corpora. However, as shown in monolingual IR, a Chinese sentence can also be segmented into n-grams of characters (unigrams or bigrams). Therefore, an alternative query translation method is to use n-grams of Chinese characters as translation units. This possibility has not been studied previously. This is the focus of this chapter.

3.2.4 **Language Modeling Approach**

Statistical language modeling is an approach widely used in current IR research. Compared to other approaches (e.g. vector space model), it has the advantage that different factors of IR can be integrated in a principled way. For example, unlike in vector space model, term weighting becomes an integral part of the retrieval model in language modeling. In addition, LM can also integrate easily query translation, as well as considering multiple indexing units in Chinese. Therefore, we will use an LM approach in this chapter.

The basic approach of language modeling to IR is to build a statistical language model for each document, and then determine the likelihood that the document model generates the query (Croft 2003), (Ponte & Croft 1998). An alternative is to build a language model for each document as well as for the query. A score over document is determined by the difference between them. A common score function is defined by the negative Kullback-Leibler divergence or relative entropy as follows:

$$\begin{aligned} Score(D,Q) &= -KL(\theta_Q || \theta_D) \\ &= -\sum_{w \in V} P(w|\theta_Q) \log \frac{P(w|\theta_Q)}{P(w|\theta_D)} \\ &\propto \sum_{w} P(w|\theta_Q) \log P(w|\theta_D) \end{aligned} \tag{3-1}$$

where θ_Q and θ_D are the parameters of language model for query Q and document D respectively, V is the vocabulary of the language. The simplest way to compute query model $P(w|\theta_Q)$ is estimating probability by the maximum likelihood according to query text. For the document model, it is necessary to use a certain smoothing method, such as absolute discounting, Jelinek-Mercer, Dirichlet prior, etc., to deal with the problem of zero-probability for the missing words in the document (Zhai & Lafferty 2001).

In CLIR, words in Q and D are in different languages. Query translation can be integrated into the query model $P(w|\theta_Q)$ formulas follows:

$$\begin{aligned} P(t_i|\theta_{Q_i}) &= \sum_{s_j} P(s_j, t_i|\theta_{Q_i}) \\ &= \sum_{s_j} t(t_i|s_j, \theta_{Q_i}) P\left(s_j\middle|\theta_{Q_j}\right) \\ &\approx \sum_{s_j} t(t_i|s_j) P\left(s_j\middle|\theta_{Q_j}\right) \end{aligned} \tag{3-2}$$

where s_j is a word in source language, t_i is a word in target language, $t(t_i|s_j)$ is a translation probability between s_j and t_i. This probability is provided by a translation model trained on a parallel corpus. In our case, we use IBM model 1 (Brown et al. 1993) trained using GIZA++ toolkit[1]. We will provide some details about the model in Section 3.4. A similar approach has been used in (Kraaij, Nie & Simard 2003) for CLIR between European languages, in which s_j and t_i are words.

For CLIR with Chinese (as the target language), t_i can either be words or n-grams. Therefore, we are faced with an additional problem of choosing between, or combining, different indexing units.

[1] http://www.fjoch.com/GIZA++.html

3.3 Using Different Indexing and Translation Units for Chinese IR and CLIR

3.3.1 Combination Model for Information Retrieval

Several studies have compared utilizations of words and n-grams as indexing units for Chinese IR (Nie et al. 2000), (Luk, Wong & Kwok 2002). Most of them have been done in models other than language modeling. Here, we first re-examine the impact of different indexing units within the language modeling framework. Then, we test using different units together by a combination model.

As we discussed above, we have the following index units for Chinese text: segmented words, unigram of Chinese characters, bigrams, words with unigram characters, and bigrams with unigram characters. The latter two index units able to combine words (bigrams) and n-grams naturally.

An alternative approach is more flexible to combine index units. We can create several indexes for the same document: using words, unigrams and bigrams separately. Then during the retrieval process, these indexes are combined to produce a single ranking function. In LM framework, this means that we build several language models for the same document and query. Each type of the model determines a score *Score$_i$*. The final score is a combination of these scores. So, in general, we define the final score as follows:

$$Scroe(D,Q) = \sum_i \lambda_i Score_i(D,Q) \tag{3-3}$$

where *Score$_i$* is the score determined by a type of model (in our case, either unigram, bigram or word model) and λ_i is importance in the combination (with $\sum_i \lambda_i = 1$). In particular, we can have the following possible basic indexing strategies:

- W (Word): segment sentences into words, and only use the word model for retrieval
- U (Unigram): segment sentences into unigrams (single characters), and only use unigram model for retrieval.
- B (Bigram): segment sentences into overlapping bigrams of characters.
- WU (Word+Unigram): segment sentences into both words and unigrams, as in (Nie et al. 2000).
- BU (Bigram+Unigram): segment sentences into both overlapping bigrams of characters and unigrams.

These strategies can then be combined according to Formula (3-4). For example, we can combine word and unigram models, bigram and unigram models, or word, bigram and unigram models, which we denote respectively by W+U, B+U and W+B+U as follows ($0 < \lambda < 1$):

$$Score_{w+u}(D,Q) = \lambda \cdot Score_w + (1-\lambda) \cdot Score_u$$

$$Score_{b+u}(D,Q) = \lambda \cdot Score_b + (1-\lambda) \cdot Score_u \qquad (3\text{-}4)$$

$$Score_{b+w+u}(D,Q) = \lambda_b \cdot Score_b + \lambda_w \cdot Score_w + (1-\lambda_b-\lambda_w) \cdot Score_u$$

3.3.2 **Creating Different Translation Models for CLIR**

For CLIR, we use a TM to translate query Q_s from source language to target language. Here, we use maximum likelihood estimation to estimate the source terms in the query, that is: $P(s_j|\theta_Q) = \frac{c(s_j;Q_s)}{|Q_s|}$. The query model in Formula (3-2) becomes:

$$P(t_i|\theta_{Q_s}) = \sum_{s_j \in Q_s} T(t_i|s_j) \frac{c(s_j;Q_s)}{|Q_s|} \qquad (3\text{-}5)$$

where $c(s_j;Q_s)$ is occurrence of term s_j in query Q_s, and $|Q_s|$ is the number of terms in Q_s.

The simplest TM is English-Chinese word-to-word translation model, which can be trained from English-Chinese parallel corpus (in which Chinese sentences are segmented into words). If only words are used, then we will have a TM translating English words into Chinese words. We denote this translation approach by W. To improve the retrieval coverage (recall) in CLIR, we can use the same method as in monolingual IR: we expand each Chinese word sequence in the parallel texts by adding the unigrams. The resulting translation model will suggest both Chinese words and characters as translations of English words. We denote this translation model by WU. The addition of single characters into parallel sentences aims to deal with the same problem as in monolingual Chinese IR. For example, if only 国家 (country) is segmented as a word in a parallel sentence, then this word will be suggested as the only translation candidate for "country". In fact, 国 (country) is another reasonable alternative for the same meaning. Therefore, by adding single characters into the training sentence, the TM can also suggest 国 as another translation candidate to "country". This approach is simple. We only need to perform the following transformation of each parallel sentence:

$$e_1 e_2 \dots e_n || w_1 w_2 \dots w_m \Rightarrow e_1 e_2 \dots e_n || w_1 w_2 \dots w_m c_1 c_2 \dots c_k$$

where e_i is an English word, w_i is a Chinese word, c_i is a Chinese character included in $w_1 \dots w_m$. GIZA++ is then used to create an IBM 1 model. Now, the word "country" is translated into not only 国家 (country): 0.2216, but also 国 (country): 0.2501, 家 (home): 0.1871, etc.

In the same way, if we append characters to bigrams, the resulting TM will translate an English word to Chinese bigrams and unigrams.

Now we show how these TM are used for CLIR. Firstly, we notice that the translation candidates with low probabilities usually are not strongly related to the query. They are more noise than useful terms. So, we remove them by setting a threshold δ: we filter out the items t_i with $T(t_i|s_j) < \delta$. Then, the probabilities of the remaining translation candidates are re-normalized so that $\sum_{t_i} T(t_i|s_j) = 1$.

Then, we calculate the query model by Formula (3-5). To further reduce the noise, we use one of the following two methods to select translations:

(1) For each source term s_j, we select the top N best translations.

(2) We sort of the translation candidates by $P(t_i|\theta_Q)$ according to Formula (3-2) and select the top $N \cdot |Q_s|$ terms as translation. Here N is a fixed parameter that we can tune manually.

3.3.3 **Using Co-Occurrence Terms**

Translation models are created for word translation. That is, the translation of a word only depends on the source word in isolation. In many cases, a single word is ambiguous. For example, the word "intelligence" has several meanings. It can be translated into Chinese as 智能, 情报, etc. In order to solve the ambiguities, several studies have exploited the context words to determine the most appropriate translation candidates. For example, Gao et al (Gao et al. 2002) uses a cohesion measure between the translation candidates for different source words to select the ones with the highest cohesion. Ballesteros and Croft (Ballesteros & Croft 1998) uses co-occurrence statistics for translation disambiguation.

However, all these studies focus on the selection ambiguous translations in the target language afterwards. In (Bai, Nie & Cao 2006), a different approach has been proposed to suggest related words for query expansion according to more than one query word at each time. For example, instead of using ambiguous term relations "Java→programming" and "Java→island", we include more than one term in the condition: "(Java, computer) → programming", where "(Java, computer)" means that the two words co-occur in some window. By adding more terms into the condition, the derived term is more strongly related to the query, and it is context-dependent.

In this study, we use the same idea but for query translation: In order to determine a target language translation, we make use of more than one source language word. For example, if “java” co-occurs with “computer”, then the probability of translating it into 程序 (program) and Java 语言 (Java language) will be much higher than into 爪哇岛 (Java island), i.e., $T(\text{程序}|\,java, computer) \gg T(\text{岛}|\,java, computer)$.

In order to obtain such context-dependent translation relations, we perform a co-occurrence analysis on the parallel texts. As in (Bai, Nie & Cao 2006), we also limit the condition part of the translation relations to two words.

The first question is what pair of words can be considered as meaningful pairs for translation. A meaningful pair of words is the one that brings more information than the two words separately. Several statistical measures have been proposed to determine such pairs (Thanopoulos, Fakotakis & Kokkinakis 2002), including t-score, Pearson’s χ^2, log-likelihood ratio, pointwise mutual information and mutual dependency. The results show that log-frequency biased mutual dependency ($LFMD$) and log-likelihood ratio (LLR) outperform the other methods. Therefore, we choose the LLR method for identifying meaningful co-occurrence words. LLR of words w_1 and w_2 is determined in as follows (Dunning 1993):

$$LLR(w_1, w_2) = -2\log\frac{L(H_0)}{L(H_1)} = -2\log\frac{L(c_{12}, c_1, p)\cdot L(c_2 - c_{12}\,, N - c_1, p)}{L(c_{12}, c_1, p_1)\cdot L(c_2 - c_{12}, N - c_1, p_2)}$$

where H_0 is the hypothesis of $P(w_2|w_1) = p = P(w_2|\neg w_1)$, and H_1 is $P(w_2|w_1) = p1 \neq p_2 = P(w_2|\neg w_1)$; $L(k, n, x) = x^k(1-x)^{n-k}$; c_1, c_2, and c_{12} are the occurrences of w_1, w_2 and w_1w_2 respectively; $p = c_2/N$, $p_1 = c_{12}/c_1$, $p_2 = (c_2 - c_{12})/(N - c_1)$. Usually, the co-occurrence of words should be limited within the same context (paragraph or sentence) and not far away from each other. We also limit word co-occurrences in the same sentence and within a fixed size of window: *win_size*. We apply a threshold to filter out word pairs with low LLR values, and keep the remaining word pairs (a list of meaningful word pairs).

Now, we can extend the source sentences of parallel corpus. For all words e_i and e_j, if the distance between them is less than *win_size* and they are in the list of meaningful word pairs, we add the pair $e_{i}\_e_j$ into the source sentence as follows:

Original sentence pair: $e_1 \dots e_n \,||\, w_1 \dots w_n$

Transformed pair: $e_1 \dots e_n\ e_{i}\_e_j\ \dots\ ||w_1 \dots w_n$

With the word pairs added, we train a translation model (IBM model 1), which include two types of translations: one is from English word to Chinese words, TM_0; another is from English word pair to Chinese words, TM_{co}.

The above approach can be viewed as a way to integrate the translation of compound terms. However, this approach is more flexible than that using compound terms – the determination of compounds usually require stricter syntactic constraints between compounds, while in our method words can freely group to form word pairs provided that they appear together often. Not only this method has a larger coverage, but also it can consider the influence of any useful context word in translation of a word without requiring them to form a compound term.

The final question is how these translation relations can be used for query translation. The basic idea is adjusting the probabilities of TM_0 according to TM_{co} in the sentence context. The translation probabilities (in TM_0) should be boosted if the translations are also proposed by the co-occurrence translation model (TM_{co}), and decreased otherwise. The translation model in Formula (3-2) is then defined as follows:

$$T(c|e_i,\theta_Q) = \sum_{e_i \in Q} (1-\alpha_{e_i e_j}) T_0(c|e_i) + \alpha_{e_i e_j} T_{co}(c|e_{i}\_e_j) \tag{3-6}$$

where the parameter $\alpha_{e_i e_j} \propto LLR(e_i, e_j)$, which is a value within the range [0,1], is a confidence factor measuring how strong the two words are related in the query. The final translation probability for each e_i is then normalized so that $\sum_c T(c|e_i,\theta_Q) = 1$.

3.4 Experiments

3.4.1 The Experiment of Chinese Monolingual IR

We use Lemur toolkit[1] with KL-divergence and Dirichlet prior smoothing method. We evaluated the monolingual IR and CLIR using two TREC collections and three NTCIR collections: TREC5, 6, and NTCIR3, 4, 5. The statistics are described in Table 3-1.

Table 3-1. Collection and query topic description

Query&Coll.	Description	Size(MB)	#Doc	#Topic
TREC5	Peoples Daily & Xinhua news agency	173	165K	28
TREC6	Peoples Daily & Xinhua news agency	173	165K	26
NTCIR3	CIRB011&CIRB020	543	381K	50
NTCIR4	CIRB011&CIRB020	543	381K	60
NTCIR5	CIRB040	1106	901K	50

Table 3-2 gives the retrieval results measured in MAP (Means Average Precision), where for each of collection, we obtain two results: one with "title" of each topic as the query, another with "title+description" as query. We use different index and retrieval units described in Section 3.3: word segmentation (W), bigrams (B), Unigrams (U), mixture of words and unigrams (WU), mixture of bigrams and unigrams (BU). In addition, we also tested several combinations of these indexing methods, by combining their ranking scores. Namely, we combined W and U indexes (W+U) as well as B and U indexes (B+U). We vary the combination factor of Formula (3-4) from 0.1, 0.2,…, to 0.9, and results show that when we attribute around 0.3 to W or to B and 0.7 to U, we obtain the best performances. When combining W, B, and U (W+B+U), we tune the parameters manually by try λs from 0 to 1 by 0.1. On average, $\lambda_u = 0.6$, $\lambda_w = \lambda_b = 0.2$ gives best results.

[1] http://www.lemurproject.org

We can observe that using words (W) or using bigrams (B) as indexing units, we obtain quite similar results. This is consistent with the observations in previous studies. What is surprising in our experiments is that using unigrams alone (U), we can also obtain very good results, which are even better than W and B. In some previous studies, unigrams have not been found to be as effective as bigrams (Nie et al. 2000). We believe that the difference may be due to the use of different retrieval models: we use language modeling approach which is different from previous ones. The language modeling may have a capacity to extract discriminative unigrams higher than the other models. Even if characters are not always meaningful, their probabilities are assigned in LM in such a way that more meaningful characters are attributed more different probabilities in different documents. These characters will make more difference between documents, thus affect document ranking more. This capability of LM to consider discrimination values of indexes is analyzed in (Zhai & Lafferty 2001).

Table 3-2. Comparing Chinese monolingual IR results

Chinese Monolingual IR (Query: Title)								
Query&Coll.	**W**	**B**	**U**	**WU**	**BU**	**.3W+.7U**	**.3B+.7U**	**W+B+U**
TREC5	0.2585	0.2698	0.3012	0.3298	0.3074	0.3123	0.3262	0.3273
TREC6	0.3861	0.3628	0.3580	0.4220	0.3897	0.4090	0.3880	0.4068
NTCIR3	0.2609	0.2492	0.2496	0.2606	0.2820	0.2754	0.2840	0.2862
NTCIR4	0.1996	0.2164	0.2371	0.2254	0.2350	0.2431	0.2429	0.2387
NTCIR5	0.2974	0.3151	0.3390	0.3118	0.3246	0.3452	0.3508	0.3470
Average	**0.2805**	**0.2827**	**0.2970**	**0.3099**	**0.3077**	**0.3170**	**0.3184**	**0.3212**
(Query: Title + Description)								
TREC5	0.3240	0.3496	0.3433	0.3553	0.3553	0.3581	0.3693	0.3668
TREC6	0.4909	0.5068	0.4709	0.5095	0.5165	0.5165	0.5116	0.5269
NTCIR3	0.2822	0.2692	0.2672	0.2788	0.2766	0.3118	0.3080	0.3167
NTCIR4	0.2122	0.2074	0.2390	0.2195	0.2170	0.2464	0.2443	0.2449
NTCIR5	0.3386	0.3490	0.3741	0.3421	0.3516	0.3858	0.3942	0.3869
Average	**0.3296**	**0.3364**	**0.3389**	**0.3410**	**0.3434**	**0.3637**	**0.3654**	**0.3684**

When we mix up two types of indexing units in the segmentation step – W with U (WU) and B with U (BU), we can see that the results are generally better than when only one type of index is used. This observation is consistent with (Nie et al. 2000)However, the best methods are those that create separate indexes for each type, and then combine the ranking score according to Formula (3-4). The result of combining word, bigram and unigram together shows that this approach can produce slightly better results than W+U and B+U, but the improvements are marginal. A possible reason is that words are usually formed with two characters. So there is a large overlap between words and bigrams. As a consequence, once words have been used, bigrams do not bring much new information, and vice versa.

Our statistical hypothesis test shows that the improvements of mixed and combined approaches over W, B, and U are statistically significant only for some results; others are marginal. It means that different topics may benefit from different indexing units.

Overall, comparing W to B, we obtain comparable effectiveness, either when they are used alone or they are combined each with U. Therefore, we can conclude that bigrams are reasonable alternative to words as indexing units. The combination between them does not seem to be interesting. This shows that both types of indexing units captures about the same information. On the other hand, unigrams are complementary to them and it is useful to combine unigrams with either bigrams or words.

3.4.2 **Using Different Chinese Translation Units for CLIR**

Our model requires a set of parallel texts to train a TM. We have implemented an automatic mining tool to mine Chinese-English parallel texts from Web using a similar approach to (Chen & Nie 2000). Parallel texts are mined from six websites, which are located in United Nations, Hong Kong, Taiwan, and Mainland of China (Chinese pages encode in GB2312, Big5, and Unicode). It contains about 4000 pairs of pages and includes some noise (non-parallel texts).

After converting the HTML texts to plain text and mark the paragraph and sentence boundaries, we use a sentence alignment algorithm to align the parallel text to sentence pairs. Our sentence alignment algorithm is an extension of the length-based method, which also considers the known lexical-translation according to a bilingual dictionary. The idea is that if a pair of sentences contains many words that are mutual translations in the dictionary, then their alignment score is increased. Here we use CEDICT[1], which includes 28,000 Chinese words/phrases. After sentence alignment, we obtain 281,000 parallel sentence pairs. Another extension we made to the traditional TM training is to use sentence alignment score during TM training. A pair of sentences with a higher score is considered more important in the training process than a pair with lower score. This factor can be easily incorporated into the GIZA++ tool. Our previous experiments showed that these measures result in better translation models and higher CLIR effectiveness (Shi & Nie 2006). In this study, we use the same approach for TM training.

For English, we use a simple morphological analyzer[2] to remove the English language suffixes, such as *-s*, *-ed*, *-en*, *ase*, *-yl*, *-ide*, etc. For Chinese word segmentation, we use an existing segmentation tool[3]. The segmenter uses a version of the maximal matching algorithm based on a lexicon.

Once the parallel corpus has been pre-processed as above, GIZA++ is used to train translation models – IBM model 1.

When preprocessing Chinese texts in the parallel corpus, different Chinese units have been created separately. We therefore obtain several types of translation models:

- *W*: English word to Chinese words;
- *B*: English word to Chinese bigrams;
- *U*: English word to Chinese unigrams (single characters);
- *WU*: English word to Chinese words and unigrams;
- *BU*: English word to Chinese bigrams and unigrams.

[1] http://www.mandarintools.com/cedict.html
[2] http://web.media.mit.edu/~hugo/montylingua/
[3] http://www.mandarintools.com/download/segment.zip

In our experiments, we set N=10 and use the second method introduced in Section 3.2, i.e. keep top $10 \cdot |Q|$ target words in query model. This method is slightly better than the first one. As for monolingual IR, when two function scores are combined using Formula (3-4), we set $\lambda = 0.3$ for either W or B models. The CLIR results (measured in MAP) are shown Table 3-3.

We can observe that in general, CLIR effectiveness is much lower than monolingual effectiveness. This is normal and consistent with previous studies. Although we can expect a quite high effectiveness for CLIR between European languages, in general, the CLIR effectiveness between English and Chinese is much lower than monolingual effectiveness. So, the drop we observe here is not an exception.

What is important to observe is the comparison between different translation approaches.

As for monolingual IR, we see that using W or B as translation units, we can obtain similar results. Using U as translation units, we obtain generally better effectiveness. This result is also new compared to the previous studies. This shows that Chinese characters can be reasonable indexing and translation units for Chinese.

When we mix up Chinese units in TM (WU and BU), we can obtain further improvements. On the other hand, although it is still an interesting approach to translate the query into different units with different TMs and then combine their ranking scores by Formula (3-4), we do not observe any significant increase using this last approach over WU and BU, contrarily to monolingual IR.

Table 3-3. CLIR results using different translation models

English→Chinese CLIR (Query: Title)							
Query&Coll.	**W**	**B**	**U**	**WU**	**BU**	**.3W+.7U**	**.3B+.7U**
TREC5	0.1904	0.2003	0.1922	0.2448	0.2277	0.2158	0.2251
TREC6	0.2047	0.2293	0.2602	0.2670	0.2772	0.2672	0.2822
NTCIR3	0.1288	0.1017	0.1536	0.1628	0.1504	0.1619	0.1495
NTCIR4	0.0956	0.0953	0.1382	0.1410	0.1308	0.1337	0.1286
NTCIR5	0.1158	0.1323	0.1762	0.1532	0.1462	0.1682	0.1602
Average	**0.1470**	**0.1518**	**0.1841**	**0.1938**	**0.1865**	**0.1894**	**0.1891**
(Query: Title + Description)							
TREC5	0.2433	0.2637	0.2674	0.2984	0.2897	0.2848	0.2906
TREC6	0.2910	0.3355	0.3624	0.3745	0.3866	0.3641	0.3793
NTCIR3	0.1401	0.1189	0.1741	0.1878	0.1748	0.1977	0.1731
NTCIR4	0.1021	0.0992	0.1463	0.1493	0.1390	0.1443	0.1395
NTCIR5	0.1315	0.1430	0.2252	0.1851	0.1731	0.2051	0.2053
Average	**0.1816**	**0.1921**	**0.2351**	**0.2390**	**0.2326**	**0.2392**	**0.2376**

3.4.3 **Using English Word Pair for Translation**

To determine meaningful English word pairs, we use the monolingual English corpus, Associate Press (AP88-90). We filtered out the word pair with a LLR less than 100, and kept 828,750 pairs.

The new translation method is compared to the translation method WU, which proved to be the most effective. Here, in addition to segmenting Chinese sentences into both words and unigrams, we also group English words to form an additional term. Finally, we trained a TM (TM_{co}) from English to Chinese that also contains translations of English word pairs. Using Formula (3-6), we can get the new model that we denote by WU_{co} in the following table.

Table 3-4. Comparing different translation approach (Documents are indexed by WU in both cases)

	Query: Title			Query: Title + Description		
Query& Collection	WU	WU_{co}		WU	WU_{co}	
	MAP	MAP	%WU	MAP	MAP	%WU
TREC5	0.2448	**0.2463**	+0.6	**0.2984**	0.2910	-2.5
TREC6	0.2670	**0.2912**	+9.1	0.3745	**0.3883**	+3.7
NTCIR3	0.1628	**0.1656**	+1.7	**0.1878**	0.1869	-0.5
NTCIR4	0.1410	**0.1448**	+2.7	0.1493	**0.1536**	+2.9
NTCIR5	0.1532	0.1586	+3.5	0.1851	0.2008	+8.5

We can see that when meaningful English word pairs are considered in the translation model, the resulting retrieval effectiveness is slightly higher than the WU translation model. However, the improvements are not consistent in all cases.

For some queries, we observe that this new translation model can produce better translation. For example, for TREC6 topic CH45, The MAP of WU is 0.2514 and that of WU_{co} is 0.6439. The English title is "*China red cross*". By the WU translation model this topic is translated to "红:0.5388 中国: 0.3842 中:0.3427 国:0.2650 两:0.1336 两岸:0.0837 跨:0.0760 十:0.0720 岸:0.0718 …" The underlined Chinese words are correct translations. Once we combine TM_0 and TM_{co} by Formula (3-6), the translation becomes "中国:0.3842 中:0.3427 红:0.3007 国:0.2650 十: 0.2362 字:0.2292 红十字会:0.1662 两: 0.1025 会:0.0901 两岸 0.0642 …" We see that the translation is more related to the original query.

For some other queries, we observed decreases in effectiveness. This is the case for TREC6 topic CH24, for which the effectiveness drops from 0.3216 to 0.2437. The English title is "*Reaction to Lifting the Arms Embargo for Bosnian Muslims*". For this query, we have determined correctly "arm_embargo" as a word pair. Its translation should be "武器(weapon,arms) ／禁运(embargo)". However, due to the limitation of our parallel corpus, the translations of "arm_embargo" in TM_{co} are "运(transport):0.1045 安

全(safe):0.1025 安 (safe):0.0813 全 (complete):0.0734 禁 (forbid):0.0654 表:0.0576 禁运(embargo):0.0386 生:0.0348 发:0.0339…" We see that the meaning of "weapon" is completely lost and the meaning of "embargo" is only reflected by two low probability translations. Therefore, the result becomes worse. We believe that this decrease is largely due to the limited size of our parallel corpus and its coverage of Chinese and English words. With a larger parallel corpus, the translation model with word pairs should be able to produce larger improvements in retrieval effectiveness.

Another factor that strongly impacts this method is that we have normalized the influence of each translation component in Formula (3-6). That is, when an English word is contained in a word pair, both types of translations are combined. If a word is not part of a word pair, then only word-based translation is considered. In this case, the word-based translation will be attributed with a higher weight (because it is attributed the whole relative importance, or $\alpha_{e_ie_j} = 0$ in Formula (3-6)). This may raise some problem. Indeed, when a single word is translated, much ambiguity is introduced. Therefore, we should rather reduce our confidence on the translations from single English words. This is a problem that we will consider in our future research.

3.5 Conclusion and Future Work

Chinese words and bigrams have been considered to be two competitive indexing units for Chinese IR. In this study, we further compared these approaches and combined them with unigrams (characters). We have found that Chinese unigrams are actually more effective than either words or bigrams along – it is new in Chinese IR. In addition, by combining either words or bigrams with unigrams, we can get better retrieval effectiveness. This result is consistent with previous studies.

For CLIR with Chinese (as the target language), previous studies usually use words as translation units. In this chapter, we have tested the possibility of using bigrams and unigrams as alternative translation units. Our experiments showed that these translation units are as effective as words. In particular, unigrams have proven to be even more effective than words and bigrams.

Based on above results, we can see that Chinese characters are very meaningful units, which can be used as both indexing and translation units.

When an English query is translated into both unigrams and words or bigrams, we observed slightly higher retrieval effectiveness. However, the increase is marginal.

In order to reduce translation ambiguity, we also tested the possibility of determining Chinese translation from a pair of English words. For some queries, the results are very encouraging, but for some others, we observed rather a decrease. Therefore, the overall effectiveness is only marginally better. Still, we believe the proposed translation method can be further improved in following aspects:

- Using a large parallel corpus, we can derive more useful translation from English word pairs;
- Effectiveness can be further improved by translating both word pairs and words. In our current implementation, we only considered the strength of link between English words may not be sufficient. We have to define a better measure of confidence about the translations generated from single word or word pairs.

We will tackle these problems in our future research. It would be worthwhile to test our approaches also for other Asian languages such as Japanese and Korean.

CHAPTER 4.
RELATING DEPENDENT INDEXES USING DEMPSTER-SHAFER THEORY

Introduction to the chapter

In the previous chapter, we considered different types of indexing units. However, different units are considered to be independent: they are used separately to produce a matching score. In reality, different types of indexes are not independent. Let us use the combination of word and phrase indexes to illustrate the problem. Give the query "computer architecture", once we represent it as "computer-architecture" in a phrase index, and as "computer" and "architecture" in a word index, it is not reasonable to consider the two types of indexes independent. "computer-architecture" is strongly related to "computer" and "architecture" in the word index.

In this chapter, we try to address this problem. Our approach is to create a unique index in which both the phrase and the component words are represented, however, as dependent elements. As it is difficult to determine a priori the way that the phrase and the words are dependent, we use a flexible way to deal with it – Dempster-Shafer theory of evidence. The idea is to consider a phrase and the component words as forming a group of dependent elements. An expression such as "computer architecture" corresponds to such a group of elements: {computer-architecture, computer, architecture}. If the same expression is found in a query and a document, then the group is matched together. If, on the other hand, there is only a partial match (e.g. the document contains separate words "computer" and "architecture", then the group is discomposed to provide a partial matching score.

The use of Dempster-Shafer theory allows us to have a reasonable representation of dependent elements.

This chapter reproduces a paper published at the *Proceedings of the 17th ACM Conference on Information and Knowledge Management*: (Shi, Nie & Cao 2008).

4.1 Introduction

The problems related to the independence between terms assumption are well documented in IR literature (Evans & Zhai 1996), (Pôssas et al. 2002), (Wong, Ziarko & Wong 1985). To solve the problems, a common approach proposed in the literature is to create multiple types of indexes (Evans & Zhai 1996), using both single-word terms and multi-word terms. For example *computer architecture*, one would arrive at two possible representations: by *computer* and *architecture*, and by the compound term *computer-architecture*. In so doing, the document can match a query about *computer* or *architecture* due to the first type of index, and a query about *computer architecture* more strongly due to the second type of index.

In the previous chapter, we investigated the combination approaches for Chinese IR and English-Chinese CLIR. Although the approach offers a remedy to the practical problem of mismatch to some degree, it does not solve the fundamental problem concerning the relationship between different terms – the two types of indexes were still used as independent pieces of evidence. In reality, *computer* of the first type is strongly related to *computer-architecture* of the second type.

The importance to take into account the relationships between terms in IR is widely recognized, and a number of investigations have been carried out on it. The studies, such as (Metzler & Croft 2007), (Theophylactou & Lalmasy 1998), (Turtle & Croft 1991), use different frameworks ranging from language modeling, Bayesian network to Dempster-Shafer theory. We observe that any of the proposed methods encountered a crucial problem in implementation for the estimation of probabilities of different terms. In general, probabilities (or any other weights) are assigned to terms according to their frequencies of occurrences. However, when we observe an occurrence of the string “computer architecture” in a document, should we consider it as the one for the compound term *computer-architecture*, for the single terms *computer* and *architecture*, or for both?

Different methods have been used to deal with this problem. For example, (Evans & Zhai 1996), (Theophylactou & Lalmasy 1998) considered the occurrence simultaneously for both compound and single terms. This will in fact duplicate the occurrence, one for the compound term, and another for single terms. This obviously falsifies the final probability estimation and enhances compound terms unduly. For example, in the occurrence “computer architecture and network”, “computer architecture” will be assigned an overly enhanced importance compared to “network” due to the consideration of both compound and simple terms for it.

Another typical approach is to consider the occurrence only for single terms, and the probability of the compound term is estimated afterwards from those of the single terms (de Campos, Fernández-Luna & Huete 2003). In this case, however, it is implicitly assumed that an occurrence of “architecture” or “architecture” alone implies somehow *computer-architecture*. This is obviously not always true, and a generalized assumption can lead to a wrong probability estimation.

The above problem is fundamental in IR theory. However, it has not been paid due attention in previous research in European languages. Part of the reason is that the consideration of compound terms in addition to single terms is just an option, because words can already capture most of the document contents. The addition of compound terms can sometimes improve the retrieval effectiveness. However, in many East Asian languages (Chinese, Japanese), the above problem is omnipresent: these languages do not have a proper native definition of the notion of word. The current indexing process used in these languages is inherited from the IR studies in European languages, in which word is a clear linguistic notion. Forcing to use words as indexing units in Asian languages will hide the several important problems:

Word segmentation is often ambiguous. For the same sentence in Chinese, there may often be several different segmentation results, leading to inconsistent word sequences. The inconsistency between the segmentation of a document and that of a query will lead to mismatch.

Words in Asian languages strongly overlap. In Asian languages (in particular, Chinese), we can determine longer or shorter words from a sentence and they often overlap. A typical example is the case of single characters (or ideograms), which can represent a concept alone (e.g. 网 – network); but they can also be part of a longer word (e.g. 网络 – network) representing the same (or similar) concept. Then both 网 and 网络 can be used as terms. Obviously, as they overlap, they cannot be considered to be independent.

One may argue that these problems of ambiguous and overlapping terms also exist in European languages, e.g. when considering compound terms or when dealing with languages such as German. However, the difference is the extent to which the phenomena spread in these languages: In Chinese (and several other Asian languages), the phenomena are generalized – almost every word formed by two or more characters can be viewed as juxtaposition of two or more concepts. The consideration of multiple and dependent terms in Asian languages is thus a fundamental question rather than an option.

In previous studies on Chinese IR, remedial approaches similar to (Evans & Zhai 1996), (Theophylactou & Lalmasy 1998) have been used, e.g. (Nie et al. 2000), (Kwok 1997). For example, one can segment a sentence into all the possible (long and short) words and then count them independently. However, these approaches do not offer a radical solution to the problem.

In this chapter, we propose a different approach to consider strongly dependent terms. We will use Dempster-Shafer theory (Dempster 1968), (Shafer 1976) to group multiple terms implied in the same occurrence of a string. For example, when an occurrence of 网络 (network) is observed, we consider it as representing three possible terms: 网络, 网 and 络. This occurrence is represented as a set of terms, or *term set*, {网络, 网, 络}, as in Dempster-Shafer theory of evidence. Probability is assigned to the whole term set instead of to each of terms. This will avoid the problem of duplicating the occurrence for long terms. In this representation, terms in the same set are assumed to be dependent, but the dependency is not explicitly represented. The dependency will be considered in the query evaluation phase: To determine the score of a document facing a query, we will consider the possible relations between a term set of the document and a term set of the query. This will extend the belief and plausibility functions of Dempster-Shafer theory and allow us to define more appropriate evaluation functions for IR.

Our utilization of Dempster-Shafer theory aims to solve the fundamental problem of document and query representation by dependent terms. This is different from previous utilizations of the theory, which often exploited the theory for combining multiple pieces of evidence assumed to be independent (Plachouras & Ounis 2005), (Ruthven & Lalmas 2002), (Urban, Jose & Rijsbergen 2006).

This approach is tested on several Chinese test collections from TREC and NTCIR, and we obtained significantly better effectiveness than state-of-the-art approaches.

In the remaining of the chapter, we will first describe some related studies trying to deal with dependencies between terms. Then we will describe our approach using Dempster-Shafer theory. We will describe our experiments on several Chinese test collections. An analysis will be made on the results before drawing the conclusions.

4.2 Related Work

Single-word terms are found to be ambiguous in many cases. Therefore, compound terms or phrases have been used to complement the single-word terms. For example, one can first identifies compound terms or phrases using both statistical and linguistic analyses, then combine them with single-word terms (Evans & Zhai 1996). A typical approach is to define two retrieval scores, respectively from single-word terms and from multi-word terms. The two scores are then combined to produce a final score. This approach has shown some improvements in retrieval effectiveness in some cases, but the improvements are usually small and not consistent across studies.

One can observe that the above combination approach does not really consider the tight relationship between single-word terms and compound terms. In order to cope with the strong relationship between them, Campos et al. (de Campos, Fernández-Luna & Huete 2003) proposed a Bayesian network to represent the possible relationships between different terms. Each node in the network represents a term and a link between nodes represents their dependency. The model can be seen as an extension of the model proposed by Turtle and Croft (Turtle & Croft 1991), but (de Campos, Fernández-Luna & Huete 2003) tries to relax the constraint imposed in (Turtle & Croft 1991), that terms in the same layer of the network are independent. Although the model described in (de Campos, Fernández-Luna & Huete 2003) can integrate, in theory, any type of relation between terms in the network, the key issue is the difficulty to estimate the dependencies between terms and to assign probabilities to terms. In fact, to implement the model, Campos et al. had to heavily simplify the model and the probability of a compound term is simply estimated from those of the constituent terms considered alone. This implementation fails to reflect the initial idea of term dependencies.

The Bayesian network proposed by Tuttle and Croft (Turtle & Croft 1991) has also been used to consider both single-word terms and phrases (Croft, Turtle & Lewis 1991). In this case, they are considered to form two independent sets of terms.

In (Metzler & Croft 2007), Metzler and Croft considered term relationships within the language modeling framework. In his approach, different types of relations are assumed between terms in a set (e.g. a noun phrase), varying from strict order to more flexible proximity relations. However, the relationships are only considered between terms in a query, and when documents are indexed, single-word terms are still assumed to be independent when estimating their probabilities.

In (Nallapati & Allan 2002), term dependencies are integrated into a bigram language model, but the dependencies are restricted to a tree form, and they are estimated loosely from term co-occurrences in documents.

Several approaches have been developed using vector space model. Wong et al. (Wong, Ziarko & Wong 1985) proposed a generalized vector space model, which uses logical conjunctions of terms as new dimensions in a new vector space. However, this method will greatly increase the complexity of the model, making it intractable in practice. Pôssas et al. (Pôssas et al. 2002) followed a similar direction using term sets. Term sets group terms that co-occur frequently in documents. These sets are used to replace the traditional terms in vector space model. However, no relationship between term sets is considered.

The above review shows a critical problem in the current practice in IR: terms and term sets are usually considered to be independent. This is particularly apparent in the indexing process when terms are assigned probabilities within a document: occurrences within the document are counted separately. However, in reality, when a term occurs in a document, it often implies the occurrence of some other terms. This is particularly the case in Chinese, in which one can segment a sentence into sequences of long or short words. A longer term usually implies shorter constituent terms. For example, the sequence $abcd$ can be segmented into words ab and cd, but each of the Chinese characters a, b, c and d can also be a word. All these words are implied in the occurrence $abcd$. Then, how can we assign probabilities to these indexes, given the fact that they overlap in it? The previous approaches have suggested the following assignment schemas:

Two independent assignments: the sequence is indexed separately by characters a, b, c and $d,$ and by words ab and cd. The former are assigned a probability of 1/4 (assuming a simple uniform assignment here), and the latter 1/2. The two probability assignments are combined during query evaluation.

Mixed assignment: one can expand the occurrence by all the possible indexes implied and mix them up. Probabilities are assigned to them as if they are independent. In this case, the occurrence will be expanded to 6 terms. Then each of the indexes is assigned the probability 1/6 (again assuming a simple assignment here).

Both probability assignments are deficient. The indexes are indeed dependent, and the assignments ignore the dependencies. The reason that the above assignment schemas are used is due to the difficulty to take into account the dependences during the indexing phase, partly for efficiency reasons, but more often due to the lack of appropriate models.

The question we put forward in this chapter is: why should we force ourselves to assign a part of probability to the individual terms involved in an occurrence when we lack information for doing it? For example, when we observe the string $abcd$ in a document, we are unable to assign a precise probability to each of the terms implied because of their dependencies and the overlapping nature of their occurrences and probabilities.

The idea we propose in this chapter is to assign the probability just at the level we can, i.e. when ab is observed, we well assign the probability to the set of the implied terms $\{ab, a, b\}$, and no further assumption is made to force the assignment to each of the member terms. This is the idea used in Dempster-Shafer theory of evidence.

Dempster-Shafer theory has been used in several previous studies in IR (Plachouras & Ounis 2005), (Theophylactou & Lalmasy 1998), (Urban, Jose & Rijsbergen 2006). Theophylactou and Lalmasy considered a compound term as a set of single terms following Dempster-Shafer theory. However, the probability assigned to a set of terms was determined from the IDF values of each of the terms in the set, thereby losing all the inherent dependency between single and compound terms. Dempster-Shafer theory has also been used in (Plachouras & Ounis 2005) to combine content and link evidence in web IR, and in (Urban, Jose & Rijsbergen 2006) to combine textual and visual evidence for image retrieval. These studies are concerned with the combination of multiple pieces of evidence, which is not our focus here. In this chapter, we deal with the assignment of probability mass to terms or term sets using Dempster-Shafer theory.

In the next section, we will first describe briefly the Dempster-Shafer theory. Then we will describe how it is used in our model.

4.3 A New Model Based on Dempster-Shafer Theory

4.3.1 **Dempster-Shafer Theory**

Dempster-Shafer theory (Dempster 1968), (Shafer 1976) is developed in order to account for the lack of information, or the uncertainty. Different from the traditional probability theory, when there is lack of information to allow a precise assignment of probability to individual elements; Dempster-Shafer theory will just assign probability to sets of elements. The terms in a set will share the probability mass, however, in an undetermined way.

More specifically, let Θ be a set of basic elements under consideration. The power set, 2^{Θ}, denotes the set of all possible subsets of Θ including the empty set $\emptyset$. A function $m: 2^{\Theta} \rightarrow [0,1]$ is called a basic probability assignment (BPA), which assigns a probability mass to each of the subsets, and this function satisfies the following axioms:

1. The empty set is assigned the value 0: $m(\emptyset) = 0$
2. The sum of the probabilities assigned to all subsets of Θ is 1: $\sum_{X \in 2^{\Theta}} m(X) = 1$

The probability mass $m(A)$ assigned to a subset A expresses the proportion of all available evidence that supports the claim that the actual state belongs to A but to no particular subset of A.

From the mass assignments, we can determine two measures - plausibility (Pl) and belief (Bel), which are usually considered as the upper and lower bounds of a probability interval:

$$Bel(A) \geq P(A) \leq Pl(A)$$

The belief $Bel(A)$ for a set A is defined as the sum of all the masses of subsets of A:

$$Bel(A) = \sum_{B:B\subseteq A} m(B)$$

That is, $Bel(A)$ gathers all the evidence directly in support of A or of its subsets. The plausibility $Pl(A)$ is the sum of all the masses of the sets B that intersect A:

$$Pl(A) = \sum_{B:B\cap A\neq\emptyset} m(B)$$

That is, $Pl(A)$ gathers all evidence that *may* support A, or not in contradiction with A.

When m assigns a part of the evidence to a set A containing several elements, this is because we do not have the necessary information about the precise distribution of the probability mass to each of the members – this is the source of uncertainty. In the particular case where m assigns non-zero probabilities only to individual elements (or subsets containing only one element), Dempster-Shafer theory will correspond to the traditional probability theory, i.e. $Bel(A) = P(A) = Pl(A)$ for any A containing a single element.

4.3.2 New Indexing Method Based on Dempster-Shafer Theory

In most previous studies, e.g. using probabilistic models, terms are assumed to be independent; thus, the occurrence of one term does not affect that of another. Therefore, the probabilities of terms can be estimated simply according to their frequencies of occurrences. However, this independence assumption is more for calculation convenience than the reality, as our previous examples showed. Indeed, we do not know exactly how to assign an occurrence to the compound term and to its constituent shorter terms. This situation can be correctly accounted for using Dempster-Shafer theory.

Let us consider the case of Chinese. In Chinese, a compound term abc can often be further segmented into shorter words (suppose that we can recognize ab and bc[1]) and legitimate single-character terms a, b and c. Therefore, when abc is observed in a document, we should not segment it only as a compound term. This occurrence also implies all the other implied terms. However, as we are unable to distribute this occurrence precisely among them, an appropriate representation is to consider that the occurrence of abc indeed represents the set of terms $\{abc, ab, bc, a, b, c\}$ as in Dempster-Shafer theory. Graphically, this is illustrated in the following figure, where the large circle represents the occurrence of abc and the smaller circles those of the constituent terms:

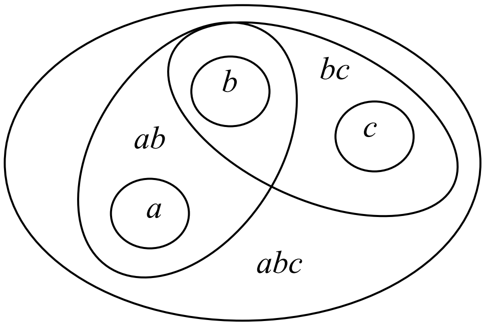

Figure 4-1. Illustration of overlapping terms

[1] In this study, we assume that ac is not a constituent term in this case, and we will only consider terms formed by consecutive character strings.

We can see that terms and their occurrences strongly overlap, but in this case, we do not further impose a way to share the occurrence (or probability) among them.

Now, for a document that contains much longer sequences of characters, we have two choices:

We can generalize the consideration of dependency between any sequence and its subsequences. For example, suppose a document containing the sequence $abcd$, in which we can recognize two legitimate compound words ab and cd. One could consider the relationships within ab and within cd, but also consider ab and cd to be dependent. This consideration of dependency can further spread to longer sequences. This generalized consideration of dependency possibly reflects the reality, as ab and cd could be somehow dependent. However, we are then faced with a serious problem of complexity, which makes the approach intractable.

Instead, we will take another option, in which we assume independence between compound terms ab and cd, which are determined using an existing segmentation process This assumption is made because in general, there are weaker relationships between segmented words than within these words (in Chinese). Therefore, in this chapter, we focus on the strong dependencies within segmented words only. The generalized consideration will be the subject of a future work.

The second option requires a word segmentation tool to chunk a text into non-overlapping segments. There are a number of such tools, and it has been shown that they can usually achieve an accuracy of over 95%. In fact, the real problem of Chinese IR is not in word segmentation accuracy, but rather the consideration of strongly overlapping terms, which is exactly the focus of this chapter.

Once a document is segmented into *non-overlapping* segments, we can use their frequency to estimate their probability as usual. However, the probability is assigned to the term set corresponding to the segment. Let us denote a segment by t, and the corresponding term set by t^*. Then we can arrive at the following mass assignment:

$$m(t^*|D) = \frac{count(t; D)}{|D|}$$

where *count*(*t*; *D*) is the frequency of occurrences of the segment t in a document D and $|D|$ is the size of the document (the number of segments).

Notice that the above assignment satisfies the conditions required for BPA of Dempster-Shafer theory.

4.3.3 **Retrieval Model**

Given a query Q, it can also be segmented into a sequence of segments $q_1\ q_2, \dots, q_n$. For example, the query 爱滋病防治 (prevention and treatment of AIDS) can be segmented into 爱滋病 (AIDS) and 防治 (prevention and treatment). The corresponding term sets are {爱滋病, 爱滋, 爱, 滋, 病} and {防治, 防, 治}. For a query, we assume a logical AND relation between different term sets. To simplify the notation, we will also represent the term sets of the query Q by $q_1^*, \dots, q_n^*$, and the term sets of terms $d_1, d_2, \dots$ in a document D by $d_1^*, d_2^*, \dots$. Then the correspondence between a document and this query can be determined by the following conditional probability using language modeling:

$$P(Q|D) = \prod_{i=1}^{n} P(q_i^*|D) \tag{4-1}$$

4.3.4 **Direct Application of Dempster-Shafer Theory**

Ideally, we would like to be able to define a precise probability function $P(q_i^*|d_j^*)$ to measure the relationship between each term set d_j^* appearing in the document and the term set q_i^* in the query. With such a function, the probability $P(Q|D)$ could be estimated as follows:

$$P(Q|D) = \prod_{i=1}^{n} P(q_i^*|D) = \prod_{i=1}^{n} \sum_{d_j \in D} P\left(q_i^* \middle| d_j^*\right) \cdot m(d_j^*|D) \tag{4-2}$$

However, the probability $P(q_i^*|d_j^*)$ cannot be estimated precisely due to the lack of information, as in Dempster-Shafer theory. Following this latter, we can nevertheless define the following lower and upper bounds for it:

$$\begin{aligned} Bel(q_i^*|D) &= \sum_{d_j \in D, d_j^* \subseteq q_i^*} m(d_j^*|D) \\ Pl(q_i^*|D) &= \sum_{d_j \in D, d_j^* \cap q_i^* \neq \emptyset} m(d_j^*|D) \end{aligned} \tag{4-3}$$

which gather all the direct support evidence and the possible support evidence, respectively, for q_i^*.

Equivalently, we can also consider that there is a *transfer* of probability mass from one term set to another in these functions. To correspond to the above functions, we can assume the transfer function $t(A|B) = 1$ between two term sets A and B respectively as follows:

$$Bel\colon t(A|B) = 1 \text{ iff } B \subseteq A;$$
$$Pl\colon t(A|B) = 1 \text{ iff } A \cap B \neq \emptyset.$$

The following figure illustrates the transfer in the two cases, where each arrow represents a transfer t=1 for each case:

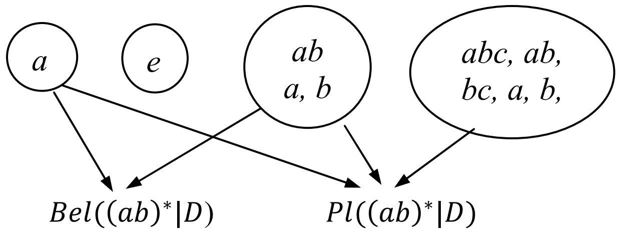

Figure 4-2. Illustration of probability transfer.

Consequently, we can also define the following bounds[1] for $P(Q|D)$:

$$Bel'(Q|D) = \prod_{i=1}^{n} Bel(q_i^*|D)$$

$$Pl'(Q|D) = \prod_{i=1}^{n} Pl(q_i^*|D)$$

Then the query likelihood can be bounded as follows:

$$Bel'(Q|D) \leq P(Q|D) = \prod_{n=1}^{n} \sum_{d_j \in D} P\left(q_i^* \middle| d_j^*\right) m\left(d_j^* \middle| D\right) \leq Pl'(Q|D)$$

[1] Notice that these functions do not correspond to *Bel* and *Pl* of Dempster-Shafer theory in the strict sense for the evaluation of a conjunction of two different subsets. In Dempster-Shafer theory, *Bel* and *Pl* are determined according to the mass assigned to the intersection of the subsets, which could be empty. So, we use *Bel'* and *Pl'* to denote our bounds.

4.3.5 **Modified Applications**

As we saw, the above application of Dempster-Shafer theory assumed a transfer of the whole probability mass from one term set to another, according to whether the former is a subset of, or intersects with, the latter. No particular knowledge about the language is used. In fact, the language can provide us with a better definition of the transfer function. For example, it can be generally admitted in Chinese that when the term ab is observed, the shorter terms a and b are also observed. So the latter are implied, and we can admit a strong transfer from the former subset $(ab)^*$ to the latter a^* and b^*. On the other hand, when a shorter term is observed, a longer term is also implied to some degree. For example, the term 爱滋 (AIDS) can imply 爱滋病 ([disease of] AIDS). Therefore, there is also some transfer from the former term set to the latter. However, this transfer strongly depends on cases. For example, the transfer from a very ambiguous and frequent character such as 人 (person) to a specific term such as 人权 (human rights) should be much lower than between 爱滋 and 爱滋病 (AIDS). The transfer degree from a term set to another strongly depends on how much the former overlaps with the latter and how frequent they are in the collection (the language). We will provide some intuitive criteria for the definition of transfer functions below. For now, let us assume such a language-dependent transfer function between term sets $t(A|B)$. This transfer function offers a flexible means to extend the original evaluation process in Dempster-Shafer theory, and it can be adapted to the particular characteristics of a language. Accordingly, we can define the following generalized form of evaluation function instead of Bel and Pl functions:

$$F(q_i^*|D) = \sum_{d_j \in D} t\left(q_i^*\middle|d_j^*\right) \cdot m(d_j^*|D)$$

$$Score(Q|D) = \prod_{i=1}^{n} F(q_i^*|D) = \prod_{i=1}^{n} \sum_{d_j \in D} t\left(q_i^*\middle|d_j^*\right) \cdot m(d_j^*|D) \tag{4-4}$$

Now, let us consider in more detail the desired transfer function for Chinese. Our definition is guided by the following general observations:

1. A longer Chinese term ab usually strongly implies a shorter term a or b;
2. A shorter term a is more ambiguous than a longer term ab, and the occurrence of a implies less strongly that of the longer term ab.
3. In addition to strictly inclusive term sets, two strongly overlapping term sets also have strong similarity. The more a term set A overlaps with another term set B, the stronger there is a transfer from A to B.

The following definitions comply with the above general observations:

1. Transfer according to word morphology: A simple method is to observe how much a term overlaps another, and define a transfer function accordingly. This corresponds to:

$$t_1(A|B) = \frac{|A \cap B|_c}{|A|_c} \tag{4-5}$$

where $|\cdot|_c$ means the length in character. The more common characters two terms sets share, the higher a transfer value will be assigned.

2. Transfer according to collection frequency:

$$t_2(A|B) = \begin{cases} 1 & if\ A \subseteq B \\ 0 & if\ A \cap B = \varnothing \\ \dfrac{count(A)}{count(A \cap B)} & otherwise \end{cases} \tag{4-6}$$

where $count(\cdot)$ is the count of occurrences of A or intersection of A and B in the document collection.

3. Transfer function according to document frequency:

$$t_3(A|B) = \begin{cases} 1 & if\ A \subseteq B \\ 0 & if\ A \cap B = \varnothing \\ \dfrac{df(A)}{df(A \cap B)} & otherwise \end{cases} \tag{4-7}$$

where $df(.)$ is the document frequency.

All these functions will be tested in our experiments. It turns out that the third definition results in the best effectiveness.

The above idea of probability transfer is related to the approach based on logical imaging (Crestani & van Rijsbergen 1995), in which the probability of a term assigned in the document is transferred to its nearest term, and the amount transferred to the query terms is used as the degree to which the document satisfies the query. Our approach is different from this approach in two respects: First, our transfer is defined between sets of terms; Second, and more importantly, our transfer is not towards one single nearest term (or set of terms), but the transfer can be made towards several term sets. This corresponds to the idea of generalized logical imaging (Kwok 1997), in which the probability mass of one term set can be transferred to several term sets close to it.

One may raise the concern about the complexity to estimate the transfer function. Although, theoretically, we will have $O(n^2)$ where n is the number of terms (including single-character words) in Chinese, in practice, we only have to estimate the transfer between term sets that share some common characters. Given the length of words is usually not more than 4 characters and the average length of words is around 2 characters, the practical time for the estimation can be much reduced. Another complexity is the exponential number of terms within a term set. However, in our case, several factors contribute to limit this complexity in practice: 1) we restrict the terms sets to those that correspond to known Chinese words (determined by a segmentation tool); 2) the length of the latter usually does not exceeds 4 characters; and 3) we only consider terms that are consecutive characters in the segment, i.e. for abc, we only consider the terms ab and bc, but not ac (which is much less likely to be a sub-term of abc in general). Therefore, the actual complexity in our calculation is strongly confined.

The transfer function is defined offline. It is then integrated with the Lemur toolkit, which we use as our basic retrieval tool.

4.3.6 **An Illustration Example**

Let us show a simple example to illustrate the process of indexing and retrieval with our method, in comparison with previous approaches.

Suppose a document containing the sequence 通讯网络图 (graph of communication network) and suppose 通讯 (communication) and 网络 (network) are legitimate compound words, and each of the characters can also be a word. Intuitively, the document is strongly related to the query 通讯网 (communication network), which can be segmented into 通讯 (communication) and 网 (network).

Traditional method 1:

If we use the traditional method with segmented words only, the document will be indexed by terms 通讯, 网络 and 图, with equal probability 1/3. However, the query term 网 will be considered to be independent from 网络. Therefore, the score of this document is 0. This evaluation is clearly deficient.

Traditional method 2:

If we index the document by both long and short terms, then we will have the following terms: 通讯, 通, 讯, 网络*,* 网, 络, and 图, and each of them is assigned a probability 1/7. The query is also segmented into: 通讯, 通, 讯, 网. The corresponding score is then $(1/7)^4$. Although this method can give some weak score to the query, no relationship between the overlapping terms is considered. The critical situation in method 1 is remedied by the fact that term occurrences are duplicated for shorter terms. However, the overlapping terms are assigned probabilities as if they occur independently in the document, leading to a deficient probability assignment.

Our method:

Using the approach we propose, the document is represented by the following term sets, each being assigned the probability 1/3 by *m*:

$d_1^* = \{通讯, 通, 讯\}$, $d_2^* = \{网络, 网, 络\}$, $d_3^* = \{图\}$

Given a query 通讯网 (communication network), the query can be segmented into 通讯 and 网, which are represented by two term sets:

$q_1^* = \{通讯, 通, 讯\}$, $q_2^* = \{网\}$

Using the first transfer function we defined, we have:

$t_1(q_1^*|d_1^*) = 1, t_1(q_2^*|d_2^*) = 1/2;$

Then $Score(Q|D) = P\left(\frac{1}{3} \cdot 1\right) P\left(\frac{1}{3} \cdot \frac{1}{2}\right) = \frac{1}{18}$.

Intuitively, this value seems more reasonable than the previous ones. We will further confirm it in our experiments.

Plausibility and Belief:

If we use *Bel'* and *Pl'* derived from Dempster-Shafer theory, we will have:

Bel'(*Q*|*D*) = 0 and *Pl'*(*Q*|*D*) = $(1/3)^2$ = 1/9.

Clearly, *Bel'* is too strict to be used as document score - many queries will have *Bel'* = 0. As to *Pl'*, it sum up all the probability masses whenever a term set intersects with the query term set. This transfer is too generous. It is problematic for Chinese because of the fact that very different words may share some character (e.g. 桌子 – table and 儿子 – son), and it is not reasonable to transfer the whole probability mass between the two term sets in this case.

In comparison, our method allows for partial probability transfer and the degree of transfer can be defined according to the characteristics of the terms.

4.4 Experiments

We test the proposed approach for Chinese IR. Several test collections are used. They come from the TREC and NTCIR experimentation campaigns. The characteristics of the collections are summarized in the following table. In our experiments, we choose to use topic titles as our queries.

Table 4-1. Characteristics of the test collections and query sets

Query&Coll.	#Doc	Size (MB)	Avg. Doc. Length	#Queries	Avg. Qry Len. (in word)
TREC5	164,778	166.9	158	28	4.7
TREC6				26	4.7
TREC9	127,938	86.2	205	25	3.7
NTCIR4	381,681	531.8	226	59	4.3
NTCIR5	901,446	1081.4	207	50	4.6
NTCIR6				50	3.9

4.4.1 **Preprocessing**

The test collections are in different coding schemas – GB and Big5. We converted all documents and queries to GB.

Chinese texts are segmented into words. Several word segmentation tools are available. We choose to use the segmentation program from LDC[1]. It uses dynamic programming to find the path which has the highest segmentation score.

Once the documents are segmented, we use Lemur[2] as our basic retrieval system to index them, i.e. to assign a probability value to each of the segments, or equivalently, to the corresponding term set.

4.4.2 **Compared Methods**

We will compare our method to several methods commonly used for Chinese IR:

- Indexing by segmented words: In this method, only the segmented words (usually the longest words) are used as indexes. We denote the method by W.
- Indexing by all words: This method is a relaxed method illustrated in section 3.4 – Traditional method 2. This method is identified by WU.
- Indexing by character unigrams and bigrams: These methods do not require to segment texts into words. Every character unigram and bigram is used as an index. We compare three versions: U – using unigrams only; B – using bigrams and BU – mixing both of them (as in WU).
- Linear combination of words and characters: This method determines two retrieval scores for a query, one according to words and another according to characters. Then the scores are combined linearly. In our experiment, we normalize the two scores by dividing them by the respective highest score. Then they are combined using 0.5 weight for each of them, which results in the best effectiveness for this method. This method is denoted by $W+U$.

[1] http://www.ldc.upenn.edu/Projects/Chinese/seg.zip

[2] http://www.lemurproject.org/

All the above methods are widely used in previous studies on Chinese IR (Kwok 1997), (Nie et al. 2000). Notice that these methods are the state-of-the-art approaches to Chinese IR. In particular, methods 3 and 4 often produced the best retrieval effectiveness in previous experiments (Shi, Nie & Bai 2007).

- Our method M_1: The transfer function is based on term overlapping – t_1 defined in section 4.3.5.
- Our method M_2: The transfer function is based on term frequency in the collection – t_2 defined in section 4.3.5.
- Our method M_3: The transfer function is based on document frequency – t_3 defined in section 4.3.5.

To avoid zero probability value, we use Dirichlet method to smoothing the basic m function and we use default Dirichlet prior 1000.

4.4.3 **Experimental Results**

Table 4-2 and Table 4-3 summarize our experimental results of baselines and our models. It shows the effectiveness (measured in Mean Average Precision – MAP) of all the traditional methods, as well as the two variants of our method.

First, we observe that results using existing methods are consistent with previous studies, except for the U method. In general, we can observe that the method B using character bigrams leads to quite equivalent effectiveness to the method W using word segmentation. What is not usual to observe in previous studies is the relatively high effectiveness obtained using character unigrams only (U). It is often higher than B (except in Trec6 and 9), which is different from previous experiments. We believe that the key reason is the different model we use: we use language modeling while previous experiments often used vector space model with $tf\text{-}idf$ weighting. The weighting schema in the language model may be more appropriate for characters, which are often of high document frequencies in

Chinese. The traditional idf factor may not be able to differentiate and weigh these characters effectively.

We also show that some combinations proposed in previous studies (BU, WU and $W+U$) can result in higher effectiveness than using a single type of index, which is consistent with previous studies. The effectiveness using these methods corresponds to current state-of-the-art in Chinese IR.

For our methods, we first tested the transfer function t_1. However, the results (M_1 in Table 4-2) are not better. This performance is clearly shows that the transfer function defined solely according to the overlapping of terms is not appropriate. Indeed, as we showed earlier, very different terms such as 桌子 – table and 儿子 – son can be assigned a quite high transfer degree, which is not reasonable. Therefore, we have to consider better criteria for the transfer function.

In Table 4-3: Our methods (‡ t.test<0.01, † t.test<0.05**)**, we can see that other 2 transfer functions can produce very competitive results, usually higher than state-of-the-art approaches. This shows that these transfer functions exploiting term distribution in the collection are more reasonable. In addition, we can see that the third transfer function defined using document frequency (M_3) leads to better results than the one defined using term frequency in collection (M_2).

For the method M_3, the t-test shows that most of the improvements over the traditional methods using a single type of index are statistically significant. Compared with the combined traditional methods, those improvements are not always statistically significant. However, we do observe general improvements on all the collections, except in one case – NTCIR4 compared to $W+U$.

The above results strongly suggest that the method we proposed is better suited to Chinese IR than state-of-the-art approaches. In particular, it can better take into account the overlapping nature of Chinese compound terms and simple terms, and account for their relationships during probability assignment. The fact that we obtained better results than traditional approaches also shows that this problem is crucial in Chinese IR and should be correctly dealt with.

In the next section, we will analyze some examples to show why our methods performed better.

Table 4-2: MAP of traditional methods (Baselines)

Query&Coll	B	U	W	BU	WU	W+U
Trec5	0.2649	0.2917	0.2773	0.3059	0.3274	0.3185
Trec6	0.3592	0.3524	0.3984	0.3794	0.4017	0.4009
Trec9	0.2109	0.2379	0.1964	0.2422	0.2287	0.2245
NTCIR4	0.2013	0.2305	0.2084	0.2263	0.2250	0.2393
NTCIR5	0.3293	0.3463	0.3758	0.3543	0.3783	0.4000
NTCIR6	0.2438	0.2664	0.2759	0.2884	0.2850	0.2973

Table 4-3: Our methods (‡ t.test<0.01, † t.test<0.05)

Query &Coll.	M_1	M_2	M_3						
	MAP	MAP	MAP	%U	%B	%W	%BU	%WU	%W+U
Trec5	0.2523	0.3221	**0.3306**	+13.3†	+24.8‡	+19.2‡	+8.1	+1.0	+3.8
Trec6	0.3278	0.4131	**0.4185**	+18.8‡	+16.5‡	+5.0†	+10.3†	+4.2	+4.4
Trec9	0.2356	0.2735	**0.2756**	+15.8†	+30.7†	+40.3†	+13.8	+20.5	+22.8†
NTCIR4	0.2106	0.2334	0.2357	+2.3	+17.1‡	+13.1‡	+4.2	+4.8	-1.5
NTCIR5	0.3175	0.4137	**0.4189**	+21.0‡	+27.2‡	+11.5‡	+18.2‡	+10.7‡	+4.7
NTCIR6	0.2528	0.2955	**0.3002**	+12.7‡	+23.1‡	+8.8‡	+4.1	+5.3	+1.0

4.4.4 **Analysis and Discussions**

A detailed analysis reveals several interesting facts.

The new method can consider various expressions of the same concept.

Some concepts have various writings in Chinese. This is the case for *AIDS*, which can be written as 爱滋病 or 艾滋病. One of test queries (CH73) used the second writing: 中国 (China) 的 (of) 艾滋病 (AIDS) – AIDS in China; while many relevant documents used the first one. As a consequence, the simple word-based approach resulted in an average precision close to 0.

On the other hand, the unigram-based method can take advantage of the common characters, and the result is very good: 0.3313 in average precision. The method combining words and unigrams resulted in 0.3983 average precision.

Using our model, the query can match both the whole exact term or partially match the alternative writing through the common characters. We obtained 0.4268 in average precision.

Compared to the traditional combination method, our method does not consider a fixed way to combine characters and terms. Instead, we try to determine the proportion of documents containing 滋 and 病 that also contain 艾滋病 (the query term). That is, in this case, the transfer function is defined as:

$$\begin{aligned} t_3\left(\left(\text{艾滋病}\right)^*\middle|\left(\text{爱滋病}\right)^*\right) &= df\left(\text{艾滋病}\right)^*/df\left(\left(\text{艾滋病}\right)^*\cap\left(\text{爱滋病}\right)^*\right)\\ &= df\left(\text{艾滋病}\right)^*/df(\{\text{滋},\text{病}\})) \end{aligned}$$

which is relatively high in this case.

Our method can exploit the implied shorter words.

When a long term implies a shorter term, our model can consider both of them, as well as their relationship.

For the query NTCIR5, query 18: 烟草商 (tobacco business) 诉讼 (accusation) 赔偿 (compensation), the first word contains a shorter word 烟草 (tobacco). The long term 烟

草商 is a much less frequent word than the shorter word 烟草 in Chinese. As a consequence, the word-based approach will also miss many relevant documents talking about 烟草. The average precision of this method is only 0.0998.

Using the mixture of words and unigrams, we will also consider the single character terms 烟 (tobacco, smoke), 草 (herb, grass), 商 (business, commerce, discuss). Although these characters are quite common and ambiguous, considering them still raised the average precision of this query to 0.2895.

In our approach, there is a strong transfer between the term set corresponding to 烟草 to the term set corresponding to 烟草商. This will enable the documents about 烟草 to have a strong correspondence with the query. For this query, our approach obtained an average precision of 0.4210.

Our method can weaken the influence of ambiguous single characters.

When character unigrams are used in a traditional approach, it participates in the matching process at an equal importance, i.e. any character is assigned a probability in a uniform manner. In fact, some characters are highly ambiguous, as we showed in the last example 烟草 (tobacco). When a compound term is discomposed into such ambiguous characters, we will almost lose all the specific meaning of the compound. Another example is 人权 (human rights), which is a quite specific term, but it is composed of two common characters: 人 (person) and 权 (right, power). If we rely on these characters in a fixed manner, many irrelevant documents will be retrieved.

On the other hand, through the estimation of the transfer function according to document frequency, our method will be able to estimate that a term set containing 人 will not transfer a large amount of probability to the term 人权, so is from 权 to 人权. As a consequence, the role of such ambiguous characters in the matching process will be diminished. This example also explains why the first transfer function (t_1) did not work well.

Our method can also introduce noise.

Determining a transfer from characters does not always produce better result. For example, CH27 of TREC5 asks for 中国 (China) 在 (in) 机器人 (robotics) 方面 (area) 的 (of) 研制 (research) - Robotics research in China. Using unigram-based method, the key term 机器人(Robot) is decomposed into 3 very common characters 机 (machine, engine) 器 (machine, utensil), 人(human, person), and it leads a low average precision (0.1319). For this query, word-based method obtains a high effectiveness (0.4302). When words and unigrams are combined (WU), we obtain 0.3734, lower than using the word alone. Our method achieves 0.3952 in average precision. Although this effectiveness is higher than the traditional WU method, it is still lower than W alone.

This example shows the potential risk of our method (indeed, of any method that combines different types of indexes): the transfer functions we defined can lead to additional noise in some cases. This also shows an aspect which we should improve in our future research – to define a more reasonable transfer function that better consider how the term set to which the transfer is made is ambiguous. Document frequency can partly account for term ambiguity, but better criteria should be found.

Despite this potential risk, the global effect of our method using transfer functions is positive. It allows matching slightly different terms, while the transfer rate is measured by the extent the two terms correspond.

4.5 Conclusions

In this chapter, we dealt with the fundamental problem by using different terms and considering the relationships between them. Different from previous approaches, our approach does not make strong independent assumption to force assigning probability to individual terms. Instead, we followed the principle of Dempster-Shafer theory and assigned probability to the whole set of terms instead of individual terms. This approach can better account for the phenomenon that term meanings strongly overlap in documents, and therefore, the same occurrence of a string can be simultaneously considered as that of a set of long term and shorter terms within it. The method to assign probability to the set of these terms allows us to avoid duplicating the occurrence artificially, resulting in a more principled way to estimate probability.

Dempster-Shafer theory includes two functions: belief and plausibility. It only considers cases that two term sets are overlapping or inclusive with no specific knowledge about the language being used. In our approach, we have extended these functions by considering the possible relationships between terms or term sets, which led to a more suitable matching function for IR, especially in Chinese.

In this chapter, we have focused on the general model and many aspects remain to be fully explored. For example, we have defined simple transfer functions, although, more complex functions can be defined by using better criteria, especially for the consideration of term ambiguity, as we discussed earlier. In addition, the transfer function can rely on richer linguistic knowledge rather than just character overlapping or term distributions. For example, one can take advantage of linguistic resources such as a thesaurus – if two term sets correspond to two related terms in the thesaurus, then a stronger transfer function can be defined. Term co-occurrences can be used as another resource. When we extend the approach in this direction, the transfer function can be used as a general mean to consider any type of relationship between terms. This will be worthwhile to do future work.

One may also have noticed that we did not explicitly represent dependencies between terms within a term set, but consider it through the utilization of transfer functions. So, term dependencies are indirectly considered.

The proposed approach can be applied beyond the limit of segmented words, i.e. we can also consider dependencies between segmented words in Chinese. The key question is how to consider relatively strong relations between segmented words, while avoiding considering terms that are not truly related in a sentence. This is an aspect to be studied in the future.

The approach has been tested only on Chinese. The general idea can also be used on other languages. For example, the same model can be used to consider the relationship between single words and compound terms in European languages. This is another direction to pursue in the future.

CHAPTER 5. MODELING VARIABLE TERM DEPENDENCIES ACCORDING TO THEIR UTILITIES IN IR

Introduction to the chapter

In the previous chapter, dependent units are grouped into a single set. Although the solution is attractive from a theoretical point of view, there are two important limitations:

- Dependencies are only allowed within a group (corresponding to a phrase), making it impossible to relate a term from a group to a term in another group. In IR, there are cases that such relation should be considered – a term can depend on another term that loosely specify its context, as in “hotel booking Java” where “hotel” provides a useful hint on the interpretation of “Java”, without forming a phrase with it.
- The relation between a long string and a shorter one with the same group is defined in a heuristic way according to their lengths. In reality, terms within a query can be dependent in different ways and we need to represent these dependencies more explicitly.

Therefore, in this chapter, we present a model that captures two specific types of dependencies: bigram and co-occurrence. This approach is an extension to the Markov

Random Field models. Rather than using MRF, we use a discriminative model so that we can select part of the binary dependencies without having to incur the high complexity of MRF models.

This chapter is a submission to the Information Retrieval Journal. Parts of this chapter have been published in SIGIR-2009: (Shi & Nie 2009), CIKM-2010: (Shi & Nie 2010A), and AIRS-2010 (Shi & Nie 2010B).

5.1 Introduction

Traditional bag-of-words IR models do not consider independence between terms, so they usually lead to unsatisfactory retrieval results. In the previous chapters, we already discussed the problem and proposed two approaches to capture the relationship between terms. The phrase-based or n-grams models try to capture the term dependency by using phrases or n-grams of terms, and they are combined with the unigram language model through model smoothing. Graphical modeling approaches have also been used to encode term dependencies, including Bayesian network (Turtle & Croft 1990), (de Campos, Fernández-Luna & Huete 2003), dependency tree (Nallapati & Allan 2002), (Gao et al. 2004) and Markov random field (Metzler & Croft 2005). More recently work includes our Dempster-Shafer model (Shi, Nie & Cao 2008) (Shi, Nie & Cao 2008), (Shi & Nie 2009) and term proximity models (Tao & Zhai 2007), (Lv & Zhai 2009), (Zhao & Yun 2009).

All the above studies have shown the usefulness of considering term dependencies of different types. However, these models have different kind limitations. The phrase and n-gram based combination models cannot explicitly represent the dependency. Dempster-Shafer model cannot capture the relation of distance terms, e.g. the distance terms fall into two terms set. We notice that most other approaches (especially within the language modeling framework) assume that all the term dependencies of the same type have the same contribution to the global model. Typically, different types of models (unigram model and dependency models) are combined via smoothing or linear combination, in which a unique weight is assigned to each component model. This is equivalent to say that a type of term dependency, say adjacency, has equal importance in the retrieval process regardless to the terms being considered. This is obviously untrue. The strength of dependency between adjacent terms changes largely. For example, the adjacency between “black” and “Monday” in the expression “black Monday” corresponds to a strong dependency, which is crucial to be captured in an IR model. On the other hand, the one in “computer game” is less critical – even if the dependency between “computer” and “game” is ignored, the retrieval effectiveness obtained using the unigram model would still be quite good. The difference between the two cases lies in the strength of the dependency as well as the utility of it for IR. Intuitively, a stronger dependency should play a more important role in the retrieval process. However, not all dependencies are necessary to be captured – if the meaning of the dependent terms together is compositional, then the omission to consider the dependency is not problematic. On the other hand, if the meaning is non-compositional (e.g. “black Monday”), then the consideration of the dependency is crucial. The above aspects have not been fully integrated in the models proposed in the literature: all the dependencies of the same type are treated with equal importance.

Another restriction in a number of previous studies is to consider adjacent words only (Bendersky, Metzler & Croft 2010). The reason is that of computational complexity. Depending on the model used, it could be difficult to extend dependency beyond adjacent words. However, dependencies can span over more distant terms. For example, in “processor specifically designed for laptop computers”, there is a strong dependency between “processor” and “laptop”. This dependency cannot be captured under this restriction. The dependency between more distant terms is not necessarily weaker than closer terms. For example, in “computer aided crime”, the relation between “computer” and “crime” is much stronger and useful for IR than the adjacent pairs “computer aided” and “aided crime”. Therefore, the strength of the dependency is not only a function of their distance. More criteria should be considered.

Term dependency is indeed of multiple types. In terms of distance, one can consider two adjacent terms, with or without their order. One can also consider dependency between terms at some distance. In terms of the nature of dependency, one can consider grammatical dependency (e.g. between subject and verb) or statistical co-occurrence dependency. In several previous studies, e.g. (Fagan 1987) and (Gao et al. 2005), it turns out that statistical dependencies are more useful than grammatical dependencies. In our study, we found that the terms dependency exists among all query terms in various strength, considering only natural phrases (even non-strict phrases) is not sufficient for IR. Our experiments show that using manually selected phrases in the dependency model usually worse than considering all adjacent/closer term pairs. So, in this study, we will consider statistical dependencies (although the model we propose can also integrate other types of dependencies). Statistical dependency can be of the following types. When a user issues a query including terms "ab", he/she may intend to retrieve documents in which:

- the terms a and b occurs independently at any position in the documents;
- the terms occur adjacently and in the same order in the documents, i.e. the bigram ab should occur; or
- the terms a and b are preferred to occur closely within a certain distance.

The first case applies to terms that are not strongly tied, and their separate occurrences in documents are sufficient. For example, the terms in the query "Ford Audi" about automobiles can be treated independently by a unigram model. The second case typically refers to a non-compositional compound expression formed by two terms, for example "black Monday". The separation between the terms would generate very different meanings. A large number of cases are in the third category. In many queries such as "laptop price", the terms are dependent to some extent, but it is too strong a constraint to require them to occur as a bigram in documents. The loose dependency between them can be captured, to some extent, by the fact that they occur relatively closely to each other in documents. This is the phenomenon considered in proximity-based models (Lv & Zhai

2009), (Tao & Zhai 2007). In this chapter, we will call such a case co-occurrence dependency within text windows.

This chapter considers all these dependencies. As we will see in the next section, a number of previous studies have considered these dependencies in some way. However, this study bears an important difference with them: in the previous studies, all the dependencies of the same type are assigned a unique importance, corresponding to the collective contribution of the dependencies of this type to the retrieval effectiveness. It is clear that individual dependencies of the same type between different terms can have very different impacts on IR, which cannot be reflected in the above method. Our model tries to integrate term dependencies according to their strength and possible impact on the retrieval effectiveness: for a query which can be treated correctly as unigrams (e.g. "Ford Audi"), the dependency between the terms will play little role. However, if the terms are strongly dependent (e.g. "black Monday"), then the dependency model will be assigned a larger importance. An important task in this chapter is to determine such a strength and impact. We will propose a learning process for it using a set of features.

This chapter will be organized as follows. In the next section, we review some related studies considering term dependencies. Then, we will describe our variable dependency discriminative model in Section 5.3, which integrates several types of dependencies. In Section 5.4, we provide more details on parameter learning. We present the experiments on TREC and NTCIR collections (in English and Chinese) in Section 5.5. Analysis, discussion and conclusions are given in Section 5.5.4.

5.2 Related Work

5.2.1 **Proximity Models**

To deal with dependencies between more distant terms, proximity model further considers the proximity of query terms in a document. Early studies (Keen 1992), (Rasolofo & Savoy 2003) used the proximity in Boolean retrieval models and BM25 models. More recently, (Tao & Zhai 2007) combined term proximity with KL-divergence language model and Okapi BM25 model, and showed significant improvement. Lv and Zhai (Lv & Zhai 2009) use a proximity-based density function (a non-increasing function of the distance $|j-i|$) to propagate the occurrence of a term at position i to its neighbor position j. Then, they define a language model for each position of a document. The final score of a document to a query is determined according to the position-dependent language models.

At the same time, (Zhao & Yun 2009) proposed a new proximity language model (PLM), which performs empirically better than previous intuitive combination of proximity. In PLM model, the basic ranking function is based on KL-divergence of language models of query and document.

$$Rank(Q,D) = -KL(\hat{\theta}_Q || \hat{\theta}_D^B) \tag{5-1}$$

In their document model, they integrated the proximity information in the following way:

$$\hat{\theta}_{D,i}^B = \frac{c(w_i;D) + \lambda Prox_B(w_i) + \mu P(w_i|C)}{|D| + \sum_{j=1}^{|V|} \lambda Prox_B(w_j) + \mu} \tag{5-2}$$

where $c(w_i;D)$ is the count of word w_i in D, $Prox_B\ (w_i)$ is proximity centrality of term w_i, and $P(w_i|C)$ is collection language model for smoothing.

For non-query terms, $Prox_B(w_i)$ is assumed to be zero. For a query term, it is computed according to term minimum distance (*P_SumProx*), average distance (*P_AveDist*), or term proximity summed over pair proximity (*P_SumProx*). Their experiments show that *P_SumProx* performs the best. As a baseline, we will use *P_SumProx,* which is defined as follows*:*

$$P\_SumProx(q_i) = \sum_{q_j \in Q, q_j \neq q_i} f\left(Dis(q_i, q_j; D)\right)$$

where $f(x) = para^{-x}$, $Dis(q_i, q_j; D)$ is minimum distance between q_i and q_j in document D, and $para$ is a free parameter to be tuned.

We notice that all the methods described in this section assign a fixed parameter for each model component. In PLM, the parameter λ is fixed (albeit the fact that the value of proximity $Prox_B(w_i)$ changes). Therefore, all dependencies of a given type proximity are assumed to have equal importance in the whole retrieval process. Although this makes the model easier to implement, the assumption is not reasonable.

5.2.2 **Markov Random Field Models**

Metzler and Croft (Metzler & Croft 2005) proposed a Markov random field model for IR. In the model, they defined three types of potential functions on clique of single terms, ordered term clique, and unordered term clique. Each potential function is a language modeling estimation smoothed by the collection, and the parameters λ_T, λ_O, λ_U are weights associated to the models as followings.

$$P(D|Q) \overset{\text{rank}}{=} \sum_{c \in C(G)} \lambda_c f(c) = \sum_{c \in T} \lambda_T f_T(c) + \sum_{c \in O} \lambda_O f_O(c) + \sum_{c \in U} \lambda_U f_U(c) \tag{5-3}$$

Two specific dependency models are proposed (see Figure 2-2) MRF-SD and MRF-FD. In the former, it only considers dependence between adjacent query terms; while in the latter, a term in the clique depends on all the others. In practice, MRF-FD is difficult to implement because of its complexity, especially when the query becomes long.

Bendersky et al. (Bendersky, Metzler & Croft 2010) notice that the fixed parameters λ_T , λ_O, λ_U do not allow one to consider the variable impact of term dependencies. They extend the MRF-SD model so that the parameters become dependent on the individual term or term pair:

$$\lambda(q_i) = \sum\nolimits_{j=1}^{k_{uni}} w_j^{uni} g_j^{uni}(q_i)$$

$$\lambda(q_i, q_{i+1}) = \sum\nolimits_{j=1}^{k_{bi}} w_j^{bi} g_j^{bi}(q_i, q_{i+1}) \tag{5-4}$$

in which the functions $g_j^{uni}(q_i)$ and $g_j^{bi}(q_i, q_{i+1})$ correspond to the importance of a unigram q_i and a bigram/biterm q_i, q_{i+1} determined using a set of features. The features include the traditional tf, idf features, as well as those extracted from Google n-grams corpus and Microsoft 2006 RFP query logs. Documents are ranked according to the following equation:

$$\begin{aligned} P(D|Q) \overset{rank}{=} & \sum_{j=1}^{k_{uni}} w_j^{uni} \sum_{q_i \in Q} g_j^{uni}(q_i) f_T(q, D) \\ & + \sum_{j=1}^{k_{bi}} w_j^{bi} \sum_{q_i q_{i+1} \in Q} g_j^{bi}(q_i, q_{i+1}) [f_O(q_i q_{i+1}, D) + f_U(q_i q_{i+1}, D)] \end{aligned} \tag{5-5}$$

The above model is called Weighted MRF-SD (WSD). This extension goes in the same direction as the method we propose in this chapter. However, the above method is still limited in the two following aspects:

Term dependency is limited to two adjacent terms. More distant terms are still assumed to be independent.

The ordered term bigram and unordered term biterm are assigned the same importance, which may not be reasonable. This will be confirmed in our experiments.

In our model, we remove the above two limitations.

5.2.3 **Discriminative Models**

Another family of approaches to consider term dependencies uses discriminative models. In the recent past, discriminative models have been empirically successful in many applications of machine learning. Discriminative models model the dependence of an unobserved variable y on an observed variable x directly. For example of classification, the model learns a direct map from input x to the class labels. In contrast, the generative models estimate the conditional probability $P(x|y)$ and the prior probability $P(y)$, and calculate the posterior by $P(y|x) \propto P(x|y)P(y)$. One of the advantages for using discriminative models is as (Vapnik 1998) pointed, "one should solve the problem directly and never solve a more general problem as an intermediate step". Another advantage is the flexibility: the discriminative function can be a posterior probability $P(y|x)$ or simply a confidence score $g(y|x)$.

In IR, discriminative models have also been used widely, such as (Nallapati 2004) (Gao et al. 2005). A typical discriminative model is formulated as follows:

$$P(Rel|D,Q) = \frac{1}{Z}\exp\left(\sum_{i}^{n} \lambda_i f_i(Q,D)\right) \tag{5-6}$$

where $f_i(Q,D)$ is a feature function with weights λ_i and Z is a normalization constant.

In (Gao et al. 2005), a linear discriminative model (LDM) is defined, leading to a score function in the same form as Equation (5-6) . The features used are related to unigrams, bigrams, phrases in a query, and headwords of document. Compared to MRF models, discriminative models are more flexible to incorporate more features. In particular, when we allow dependencies to span over distant terms, it is not necessary to also consider the dependencies with all the terms between them. In contrast, the MRF models can only capture the dependence of non-adjacent terms in their full dependent models ($MRF\text{-}FD$). For example to consider the dependency between a and c of query "$a\ b\ c\ d$", the $MRF\text{-}FD$ must model "$a\ b\ c$" together. We can limit to pair-wise dependencies in discriminative models, which often correspond to the strongest dependencies that we want to capture. This flexibility allows us to consider more term dependencies, without having to increase the complexity of the model to account for more complex and less useful dependencies.

However, we observe once again that each type of feature used in previous discriminative models is assigned a fixed value (i.e. λ_i), which is not reasonable.

5.3 Our Approach: Variable Dependency Discriminative Model

As (Nallapati 2004) observed, discriminative models have been preferred over generative models in many machine learning problems. The discriminative function can be a posterior probability or simply a confidence score. We use a discriminative model as our framework due to its efficiency and flexibility. In addition to unigrams, we consider the term dependencies between the following types term pair:

- Ordered bigrams;
- Unordered co-occurrence dependency within some distances.

In our model, the second type of dependency is considered to integrate the proximity feature according to the distance between the terms in documents. Lv and Zhai (Lv & Zhai 2009) proposed several functions to model the impact of a term on another according to its distance or proximity. One might use such functions in the discriminative model. However, for simplicity, we will use a simpler approach: we will consider co-occurrences within several different window sizes in documents, each window size corresponds to one dependency model. Let us use C_w to denote term co-occurrences within the window size *w*. In particular, C_2 considers unordered adjacent terms, or biterms (Srikanth & Srihari 2002). Let us assume a set of window sizes W (in our implementation, we use 4 window sizes 2, 4, 8, and 16 for English; 3 window sizes 2, 4, and 8 for Chinese) when we construct document models. The ranking function is extended from Equation (5-6) to the following one:

$$\begin{aligned} P(Rel|D,Q) = &\sum_{q_i \in Q} \lambda_U(q_i|Q) f_U(q_i, D) + \sum_{q_i q_{i+1} \in Q} \lambda_B(q_i, q_{i+1}|Q) f_B(q_i q_{i+1}, D) \\ &+ \sum_{w \in W} \sum_{q_i, q_j \in Q, i \neq j} \lambda_{C_w}(q_i, q_j|Q) f_{C_w}(q_i, q_j, D) \end{aligned} \tag{5-7}$$

We name above model variable dependency discriminative model (DDM), which contains three classes of discriminative features: unigram features $f_U(q_i, D)$, bigram features $f_B(q_i q_{i+1}, D)$ and co-occurrence features $f_{C_w}(q_i, q_j, D)$ where w is the window size. Each feature is associated with a function $\lambda(\cdot)$ denoting the importance of the feature for the query Q. This function allows us to take into account the dependencies between bigrams and co-occurring terms according to their strength and utility. This is fundamentally different from most previous models (except (Bendersky, Metzler & Croft 2010)) in which a fixed weight is assigned to the whole component model rather than to individual features.

Another advantage of our model is the flexibility. The co-occurrences of query terms are not limited to adjacent terms (as $MRF\text{-}SD$) nor all the query terms (as $MRF\text{-}FD$). If the distance of two terms in a query is too far, they do not tend to be dependent. In our model, we only consider the co-concurrency of query terms which are in a certain query window ($Qwin$). The terms are considered independent if they are not in $Qwin$. Therefore, for the query length is *n,* the complexity of our model is $O(n)$, whereas the $MRF\text{-}FD$ is $O(2^n)$. In our implementation, we set $Qwin$ to 6 for English and 4 for Chinese.

The discriminative feature functions we use are as follows:

$$\begin{aligned} f_U(q_i, D) &= P_U(q_i|Q) \log P_U(q_i|D) \\ f_B(q_i q_{i+1}, D) &= P_B(q_i q_{i+1}|Q) \log P_B(q_i q_{i+1}|D) \\ f_{C_w}(q_i, q_j, D) &= P_{C_w}(\{q_i . q_j\}|Q) \log P_{C_w}(\{q_i . q_j\}_w|D) \end{aligned} \tag{5-8}$$

where $\{q_i . q_j\}_w$ denote a pair of co-occurring terms q_i and q_j in document within a window of size w, and $P(\cdot\,|Q)$ and $P(\cdot\,|D)$ are language models for query and document respectively. The features are defined in this way in order to better correspond to the traditional approaches in language modeling. This will make it easier to compare our model with other approaches using language modeling. However, one could well define different features.

We can notice that if all the λ functions are defined as a constant, then the above model degenerates to the previous ones, which are indeed linear combinations of the component models. However, as we discussed in Section 5.1, the λ functions should depend on the term or term pair. We will use a set of features to learn the importance of terms, bigrams and co-occurring terms in a query. This will be described in more detail in Section 5.4. Putting all together, we have the following final ranking model:

$$\begin{aligned} P(Rel|D,Q) \overset{rank}{=} & \sum_{q_i \in Q} \lambda_U(q_i|Q) P_U^{ml}(q_i|Q) \log P_U(q_i|D) \\ & + \sum_{q_i q_{i+1} \in Q} \lambda_B(q_i, q_{i+1}|Q) P_B^{ml}(q_i q_{i+1}|Q) \log P_B(q_i q_{i+1}|D) \\ + \sum_{w \in W} \sum_{\substack{q_i, q_j \in Q \\ 0<|i-j|<Qwin}} & \lambda_{C_w}(q_i, q_j|Q) P_C^{ml}(\{q_i . q_j\}|Q) \log P_{C_w}(\{q_i . q_j\}_w|D) \end{aligned} \quad (5\text{-}9)$$

For the query models, we will simply use Maximum Likelihood (ML) estimation as follows, where t_R is any item (a unigram, a bigram or a pair of co-occurring terms) and $c(t_R; Q)$ its count in the query:

$$P_R^{ml}(t_R|Q) = \frac{c(t_R; Q)}{|Q|_R}, \quad R \in \{U, B, C_2, C_4, \dots\}$$

For the document models $P(\cdot\,|D)$ on different types of items, we use Dirichlet smoothing as follows:

$$P_R(t_R|D) = \frac{c(t_R; D) + \mu_R \cdot P_R(t_R|C)}{|D|_R + \mu_R}$$

where $c(t_R; D)$ is the number of times the item t_R occurs in document D (within a window for C_w); $P_R(t_R|C) = \frac{\sum_{D\in C} c(t_R;D)}{\sum_{D\in C} |D|_R}$ is the collection language model; μ_R is a Dirichlet prior for the corresponding type of model; and $|D|_R$ is the document length in the expression of R, i.e. the total number of unigrams, bigrams or co-occurring terms within the corresponding window size. For instance, $|D|_U$ is the number of terms (unigrams) in the document, $|D|_{C_8} = 7(n-8) + \binom{8}{2} = 7(n-4)$ is the number of possible co-occurring terms in D if we use windows of size 8.

To give an example to illustrate the model, let us imagine a query of four words $a\ b\ c\ d$, and let $Qwin$ be 3. The first component of the model considers the unigrams a, b, c and d. The second component concerns the bigrams ab, bc and cd. The third component considers the co-occurring term pairs $\{a, b\}$, $\{a, c\}$, $\{b, c\}$, $\{b, d\}$ and $\{c, d\}$. The pair $\{a, d\}$ is not considered as a co-occurrence because the distance is large than $Qwin$. If we use several window sizes for co-occurrence document models, then the co-occurring pairs will be considered evaluated in all these co-occurrence models.

Notice that if we use a single document window size (e.g. 8) to construct co-occurrence document model, restrict co-occurring query terms to adjacent terms (i.e. $Qwin = 2$), and assume term-independent λ functions, then our model degenerates to MRF-SD.

5.4 Parameters Estimations

We have the following free parameters to be estimated: (1) Dirichlet priors μ for each component language model (2) Dependence strength $\lambda(\cdot)$ for each unigram, bigram and pair of co-occurring terms.

5.4.1 **Determining the Dirichlet Priors**

When we use 4 windows sizes (2, 4, 8 and 16), we have the following priors: $\mu_U, \mu_B, \mu_{C_2}, \mu_{C_4}, \mu_{C_8}, \mu_{C_{16}}$. For unigram language model, we empirically set $\mu = 1000$ (it results in the best average performance). It is intuitive to see that a longer document expression (e.g. with a larger window size) leads to a higher sparsity. This situation will require a large μ. This has been confirmed in general language modeling. To confirm this intuition in the IR context, we run a simple experiment on TREC Disk1 combining unigrams and co-occurring terms within window size 2 or 16. We want to see if $\mu_{C_{16}}$ should be set at a larger value than for μ_{C_2}. We use here a simple linear combination: the unigram model is assigned the weight of 1, while the co-occurrence models the weight of 0.1 or 0.2. Figure 5-1 shows the results we obtain. We can see from the figure that when C_2 is used, a relatively small μ_{C_2} is preferred, especially when its importance is 0.1. On the other hand, $\mu_{C_{16}}$ should be assigned a much larger value.

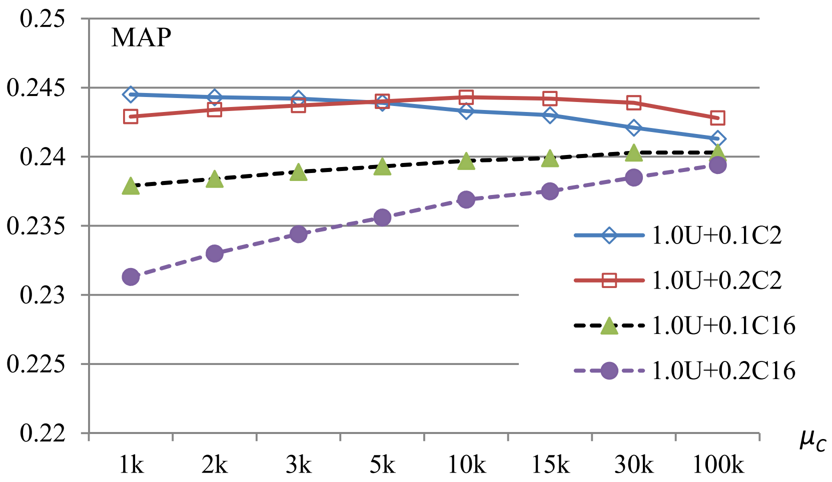

Figure 5-1. Impact of μ_C on $U + C_2$ and $U + C_{16}$ on collection Disk1: μ_{C_2} prefers a smaller value and $\mu_{C_{16}}$ prefers a lager value.

The above simple experiment confirms our intuition. Therefore, our Dirichlet priors are set according the document length in the bigram and co-occurrence expressions. If the length of a document is n in unigram expression, the lengths of document in bigram, co-occurrence in window 2, 4, 8, and 16 are respectively n-1, n-1, 3(n-2), 7(n-4), and 15(n-8). Consequentially, we set μ_B, μ_{C_2}, μ_{C_4}, μ_{C_8} and $\mu_{C_{16}}$ proportionally to 1000, 1000, 3000, 7000 and 15000 respectively. These values are not necessarily the best ones, but they turn out to perform well.

5.4.2 **Learning the Importance of a Dependency**

For a query Q_i consisting of n_i query terms, we have the following parameters to be estimated: $\boldsymbol{\lambda_{Ui}} = \lambda_{Ui,1}, \dots, \lambda_{Ui,n_i}$, $\boldsymbol{\lambda}_{Bi} = \lambda_{Bi,1}, \dots, \lambda_{Bi,n_i-1}$, $\boldsymbol{\lambda}_{Ci} = \lambda_{Ci,1}, \dots \lambda_{Ci,X}$, where $C = C_2, C_4, C_8, C_{16}$ if we set $W = \{2,4,8,16\}$, and $X = \begin{cases} n_i(n_i-1)/2 & if\ n_i < Qwin \\ (n_i - Qwin/2)(Qwin-1) & if\ n_i \geq Qwin \end{cases}$. All the parameters for Q_i is $\boldsymbol{\Lambda}_i = \{\boldsymbol{\lambda}_{Ui}, \boldsymbol{\lambda}_{Bi}, \boldsymbol{\lambda}_{C_2i}, \boldsymbol{\lambda}_{C_4i}, \boldsymbol{\lambda}_{C_8i}, \boldsymbol{\lambda}_{C_{16}i}\}$. As these parameters denote the importance of a specific term and term pair for IR, they should be tightly related to the expected retrieval effectiveness. In order to do this, we propose to learn these parameters using a set of training data including relevance judgments. In our experiments (see Section 5.5), we will use 10-fold cross validation, i.e., 1/10 of the data will be used in turn as the test data while the remaining 9/10 will be used as the training data.

Assume that we have l training queries $Q_1, \dots, Q_l$. First, we try to find the best parameters $\boldsymbol{\Lambda}_i^*$ for each Q_i according to the following equation:

$$\boldsymbol{\Lambda}_i^* = armax_{\boldsymbol{\Lambda}_i} E\left(R_{\boldsymbol{\Lambda}_i}; T_i\right)$$

where $R_{\boldsymbol{\Lambda}_i}$ is the document ranking under parameters $\boldsymbol{\Lambda}_i$; T_i is the training data (relevance judgments for Q_i); and $E(\cdot)$ is an evaluation function. In our case, we use mean average precision (MAP).

To find $\boldsymbol{\Lambda}_i^*$, we use the coordinate-level ascent algorithm introduced in (Metzler & Croft 2007). Coordinate ascent is a commonly used optimization technique for unconstrained optimization problems. The algorithm iteratively optimizes a multivariate objective function by solving a series of one dimensional searches. It repeatedly cycles through each parameter, holding all other parameters, and optimizes over the free parameter until some convergence criteria is met. This algorithm is a local search technique and only guaranteed to find a global maxima if the evaluation function E is concave. But it is efficient and effective for the parameters learning especially when there are a limited number of parameters. As we deal with short queries, the number of possible pairs of terms to consider is limited.

Once $\boldsymbol{\Lambda}_i^*$ is found, the training data can be transformed into a set of pairs $\{(x_i, z_i)\}$, where x_i is a unigram, bigram or a pair of co-occurring terms, and z_i is the optimal $\lambda_R^*(x_i)$ ($R \in \{U, B, C_2, C_4, \dots\}$) found by the coordinate-level ascent algorithm.

In the second step, we train the functions $\lambda_R(\cdot)$ such that they best fit $\lambda_R^*(\cdot)$ for the training data.

We define the features based on the current document collection and a general corpus. In this work, we will use the combination of all the test collections to simulate the general corpus. For simplicity, we assume in this study that $\lambda_B(\cdot)$ and $\lambda_C(\cdot)$ only depends on the features of the given bigram or co-occurring terms, but does not depend on other pairs in the query. This assumption is not always true, but it will simplify our definition of features. More complex features could be investigated in the future.

In our experiments, we use the following features for unigram u_i (where the $b_i\ and\ c_i$ are the query bigram and co-occurring term which includes u_i and has the largest PMI):

- $idf(u_i)$ in current collection and in general collection
- Binary test value $idf(u_i) > Threshold$?
- The frequency of u_i in current collection and in general collection
- PMI of b_i in the current collection and in general collection
- The ratio of $idf(b_i)$ and $idf(u_i)$
- PMI of c_i in the current collection and in general collection
- The ratio of $idf(c_i)$ and $idf(u_i)$

– Length of query
– The different of $idf(u_i)$ and $\max_{\mathrm{u_j \in Q}} idf(u_j)$

The following features are used for both bigram $b_i = q_i q_{i+1}$ and co-occurring terms $c_k = q_i, q_j$ (we note them x, and j means $i+1$ for bigram).

– Point-wise mutual information in the general collection: $PMI\_all(x)$
– A binary value according to the test: $PMI\_all(x) > Threshold$? (Threshold is set to 0 in our case)
– PMI in the current collection: $PMI\_coll(x)$
– Binary test value $PMI\_coll(x) > \ Threshold$?
– $idf(x) - idf(q_i) - idf(q_j)$
– $(idf(x) - idf(q_i) - idf(q_j))/(idf(q_i) + idf(q_j))$
– $count(x, coll)/min(count(q_i, coll), count(q_j, coll)$
– $count(x, coll)/max(count(q_i, coll), count(q_j, coll)$
– Whether x is in a large phrase dictionary (Termium)?
– Whether x is appears in the title of a Wikipedia article?
– The distance between the terms in the query $| j - i|$ (only for c_k).

In addition, we also define the following feature for a bigram b_i, which corresponds to the case of a bigram in which one of the constituent words only appears in this bigram:

$$\frac{\left|\{D \middle| c(b_i; D) > 1 \,\&\, c(b_i; D) = min\big(c(q_i; D), c(q_{i+1}; D)\big)\}\right|}{|\{D|c(b_i; D) > 1\}|}$$

The features defined above for an item x_i (unigram, bigram or co-occurring terms) form a vector $\mathbf{x}_i$. Now our training data are $\{(\mathbf{x}_1, z_1), (\mathbf{x}_2, z_2), \dots, (\mathbf{x}_m, z_m)\}$, $\mathbf{x}_i \in \mathbb{R}^n$, $z_i \in \mathbb{R}^1$, n is the number of features and m is the number of training data. We then use the epsilon Support Vector Machine Regression (ϵ-*SVR*) (Vapnik 1998) method to train $\lambda_R(\cdot)$ by determining the function

$$\lambda_R(x) = y(\mathbf{x}) = \mathbf{w}^T\phi(\mathbf{x}) + b$$

to minimize:

$$C\sum_{i=1}^{N}\left(\xi_i + \hat{\xi}_i\right) + \frac{1}{2}\|\mathbf{w}\|^2$$

subject to:

$$y(\mathbf{x}_i) - \epsilon - \hat{\xi}_i \leq z_i \leq y(\mathbf{x}_i) + \epsilon + \xi_i$$

$$\epsilon \geq 0, \xi_i \geq 0, \hat{\xi}_i \geq 0, i = 1, \ldots, m$$

By applying a kernel substitution, we can obtain a non-linear SVM. In our experiments, we use the LIBSVM[1] toolkit, and choose the radial basis kernel function. There are 3 parameters to be further tuned for ϵ-SVR: cost C, γ of the kernel function and the tolerance of termination criterion ε. In our experiments we use a grid search to determine them on the training data. In our experiments, C, γ, and ε are tuned in the range of 2~8, 1/31~1/256, 1/16~1/32 respectively, according to the training data.

With the above learning process, the λ parameters for different terms or term pairs are determined so as to maximize the expected MAP. Therefore, these parameters can be considered as denoting the possible impact of a term or pair of term on MAP.

[1] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

5.5 Experiments

5.5.1 Experimental setting, pre-processing, and indexing

Our experiments are on both English and Chinese collections. The English collections are carried out on TREC collections, and the Chinese collections are TREC and NTCIR. We use the title filed of topics as query. This choice is made to better correspond to real queries on search engines. The Characteristics of collections and queries are listed in Table 5-1.

Table 5-1. Characteristics of document collections and queries (the unit of query length is word for English and Chinese character for Chinese)

Coll.		Description	#Doc	Size (GB)	Query Set	Query IDs	#Qry	Avg. Q.Len
English	Disk1	AP89, WSJ87-89, FR89, DOE1/2, ZF1	510,637	1.26	Disk1	1-200	199	3.7
	Disk2	AP88, WSJ00-92, FR88, ZF2	231,219	0.90	Disk2	1-200	199	3.7
	Disk4	CR93, FR94, FT91-94	293,710	1.28	Disk4	251-350	100	2.9
	Disk5	FBIS96, LA89-90	262,367	0.95	Disk5	301-450	150	2.4
	WT10g	GOV test collection	1,692,096	11.03	WT10g	451-550	97	2.4
Chinese	TR56	People's Daily & Xinhua news agency	164,788	0.17	TREC5	CH1-28	28	12.3
					TREC6	CH29-54	26	12.0
	TR9	HK commercial data & daily news, Takongpao	127,938	0.09	TREC9	CH55-79	25	6.2
	NT34	CIRB011 & CIRB020	381,681	0.54	NTCIR4	001-060	59	8.8
	NT56	CIRB040r	901,446	1.11	NTCIR5	001-050	50	9.4
					NTCIR6	003-110	50	8.1

We performed the following pre-processing on all English documents:

- Some unimportant fields and tags are removed, such as DATALINE, SO, IN fields in WSJ collection and BYLINE, PUB, PAGE fields in FT91-94 collection;

- Stopwords are removed using a 625-stopword list from Lemur[1] toolkit;
- Stemming by Porter stemmer.

For Chinese collections, as they are in different coding schemas, we converted all the characters into GB codes. To compare to the word-based method, we use a word segmentation tool ICTCLAS[2] to segment Chinese texts to words, and use another segmentation program from LDC[3] to further segment long words into short words. For example, the long word 世界贸易组织 (World Trade Organization) will be further segmented in the second step into its constituent short words: 世界 (World), 贸易 (Trade), 组织 (Organization). The previous experiments showed that short words perform better than long words (Kwok 1997).

To train the parameters, we used two external data for English: a phrase dictionary built for French/English machine translation – Termium, which contains 853K phrase translations and the titles of Wikipedia articles (the archived file enwiki-20071018-pages-articles.xml[4] including 4,248K entries). For Chinese, we use the external data of Chinese Wikipedia articles[5], which includes 338,164 titles.

We use Indri[6] to build the basic indexes. For English, the basic index unit is word (we denote by U). For Chinese, the basic index units are Chinese characters (U). To compare to the baseline models in Chinese IR, we also build the indexes for other index units: words (W), bigrams (B), words and bigrams combined with unigrams (WU, BU).

To implement our model, we use a retrieval strategy similar to re-ranking: we first retrieve top 2000 documents for each query by the basic unigram language model, and then our method is applied to these documents. Different from the previous re-ranking

[1] http://www.lemurproject.org/
[2] http://ictclas.org/
[3] http://www.ldc.upenn.edu/Projects/Chinese/seg.zip
[4] http://download.wikimedia.org/enwiki/, on 2007-12-12
[5] http://download.wikimedia.org/chwiki/, on 2007-12-12
[6] http://www.lemurproject.org/indri/

approaches, in our re-ranking, we do not combine the initial score, because our final ranking score (Equation (5-9)) already contains a unigram language model component. The new ranking method is implemented outside the Lemur toolkit. The count the frequencies used in our model (unigrams, bigrams and term co-occurrences within windows in documents) can be gotten from basic indexes by Indri tools kit directly or by a set of our extended functions to this toolkit.

In Table 5-2, we summarize the models used in the experiments, where "diff. μ" means to use the following parameters: $\mu_U = \mu_B = \mu_{C_2} = 1000$, $\mu_{C_4} = 3000$, $\mu_{C_8} = 5000$, and $\mu_{C_{16}} = 15000$ (for English). "fixed $\lambda_R(\cdot)$" means to choose the best weights for each of the component model such that the linearly combined model achieves best result for the collection. "vary $\lambda_R(\cdot)$ " means that the weights are learnt for each pair of terms automatically through cross validation, as we described in Section 5.4.

In our experiments, we use the title (which usually only contains few keywords) as our query. Every word in the title is important for the query. Therefore, in our DDM model, we assume the importance of unigram, $\lambda_U(\cdot)$ to 1 first, such that we can focus on the impact of various bigrams and co-occurrence weights. In the sub-section 0, we will compare using fixed unigram weight to using various unigram weights.

Table 5-2. The description of our models used in the experiments

Model	**Adjacent query terms only**	**Discriminative functions used**	$\boldsymbol{\mu}$	$\boldsymbol{\lambda_R}(\cdot)$
MRF-SD	Yes	U, B, C_8	1000	Fixed
DDM-T1	No	U, B, C_8	1000	Fixed
DDM-T2	No	$U, B, C_2, C_4, C_8, (C_{16})$	diff. μ	Fixed
DDM	No	$U, B, C_2, C_4, C_8, (C_{16})$	diff. μ	Vary except λ_U=*1*
DDM$^+$	No	$U, B, C_2, C_4, C_8, (C_{16})$	diff. μ	Vary all λ_R

The DDM is a full implementation of our proposed model, $DDM\text{-}T1$ and $DDM\text{-}T2$ are models proposed for testing the usefulness of non-adjacent term and different distances with fixed weights. The DDM^{+} is the model varies all term weights including λ_U.

5.5.2 Experimental results on English collections

One independent model and three state-of-the-art dependent models are chosen as our baselines: Unigram mode (U), $MRF\text{-}SD$, weighted $MRF\text{-}SD$ (WSD), and PLM. The results with the baseline methods are shown in Table 5-3.

Table 5-3. The results of the baseline models (‡ t.test<0.01, † t.test<0.05)

Query &Coll.	Unigram	MRF-SD (best $\lambda_T, \lambda_O, \lambda_U$)		WSD (best param trained on the queries)		WSD-X (10-fold cross validation)		PLM (best $para, \lambda$)	
	MAP	MAP	%U	MAP	%SD	MAP	%WSD	MAP	%SD
Disk1	0.2382	0.2453	+3.0%‡	0.2478	+1.0%	0.2464	-0.6%	0.2425	-1.2%
Disk2	0.2340	0.2480	+6.0% ‡	0.2504	+1.0%	0.2487	-0.7%	0.2447	-1.3%
Disk4	0.1845	0.1956	+6.0%	0.1974	+0.9%	0.1871	-5.2%	0.2011	+2.8%
Disk5	0.2365	0.2461	+4.1% ‡	0.2476	+0.6%	0.2383	-3.8%	0.2469	+0.3%
WT10g	0.2042	0.2169	+6.2% †	0.2212	+1.9%	0.2199	-0.6%	0.2181	+0.5%

For $MRF\text{-}SD$ and PLM, we use a grid search to find the best MAP for each collection. The step for searching $\lambda_T, \lambda_O, \lambda_U$ for $MRF\text{-}SD$ is 0.05. The search space of $para$ in the PLM model is from 1.5 to 1.9 and λ from 3 to 9.

To compare with the WSD model, we simulate the implementation in (Bendersky, Metzler & Croft 2010). The features we use to determine the weights of $g_j^{uni}(q_i)$ and $g_j^{bi}(q_i, q_{i+1})$ are according to those used in (Bendersky, Metzler & Croft 2010). But two following datasets are not used in our simulation: *Google n-grams corpus* and *Microsoft 2006 RFP query logs*. Instead, we add the features from a large phrase dictionary

(Termium). As a consequence, our result with the simulation is slightly different from that reported in (Bendersky, Metzler & Croft 2010), but the general comparison to the other models (namely to $MRF\text{-}SD$) is consistent with it.

As we can see, the $MRF\text{-}SD$ model consistently outperforms the traditional unigram model, and the WSD model outperforms $MRF\text{-}SD$ model on all the collections. However, PLM has a performance globally similar to $MRF\text{-}SD$ – the improvements are not consistent over the collections.

In the following subsections, we will examine several questions.

5.5.2.1 Is non-adjacent query pair useful?

We compare the result of using adjacent pairs query terms ($MRF\text{-}SD$) vs. using both adjacent and non-adjacent pairs in the query ($DDM\text{-}T1$) (see Table 5-4 below). In both cases, the document co-occurrence model is constructed by considering co-occurrences within windows of size 8 as in (Metzler & Croft 2005). The result (in Table 5-6) shows that when non-adjacent term pairs are considered, we obtain consistent improvements in retrieval effectiveness, although the improvements are not statistically significant. This result tends to confirm our hypothesis that enlarging the dependencies to non-adjacent terms is useful in IR.

5.5.2.2 Co-occurrence within different distances

In DDM-T2, we define several component models for documents for term pairs within different window sizes: 2, 4, 8 and 16. The model for each window size is assigned a different weight. From Table 5-4, we can see that this model performs consistently better than $DDM\text{-}T1$, which uses a single window size. This result suggests that term pairs at different distances have different dependence strengths and impacts on IR. They should be considered separately. This observation is consistent with the approach used in (Metzler & Croft 2005).

Table 5-4. Using non-adjacent pairs and considering co-occurrences within different distances

Query &Coll.	DDM-T1 (best $\lambda_U, \lambda_B, \lambda_{C_8}$)			DDM-T2 (best $\lambda_U, \lambda_B, \lambda_{C_2}, \ldots, \lambda_{C_{16}}$)			
	MAP	**% U**	**% SD**	**MAP**	**% U**	**% SD**	**%T1**
Disk1	0.2457	+3.15%‡	+0.16%	0.2458	+3.2%‡	+0.20%	+0.04%
Disk2	0.2484	+6.15%‡	+0.16%	0.2486	+6.2%‡	+0.24%	+0.08%
Disk4	0.2023	+9.65%†	+3.43%	0.2053	+11.3%†	+4.96%†	+1.48%
Disk5	0.2465	+4.23%‡	+0.16%	0.2474	+4.6%‡	+0.53%	+0.37%
WT10g	0.2205	+7.98%†	+1.66%	0.2223	+8.9% ‡	+2.55%	+0.82%

In $DDM\text{-}T2$, we have the following component models: unigram, bigram, co-occurrence models. It is interesting to examine the relative contribution of each of these component models. This can be reflected by the weights we assign to them in the optimal setting. Table 5-5 shows the best weights assigned to the models for each collection. It is not surprising to see that the unigram model is the most important one, taking 79.1% of the importance in the global model on average. Bigram model appears to be the second most important model. The models C_2 and C_4 have slightly lower importance, while the models for larger window sizes (i.e. C_8 and C_{16}) are marginally important. We notice a lower importance of C_2 than C_4. However, this does not mean that the adjacent term pairs in C_2 are less important than those in C_4. One has also to consider the fact that part of the dependencies between adjacent terms is captured by bigrams (B). So, the lower weight for C_2 does not contradict the observation that smaller windows capture stronger and more useful term dependencies.

We can also observe a quick decay of the importance along with the increase of window size. This is intuitive, and confirms the assumption used in (Metzler & Croft 2005).

From the Table 5-5, we can also notice the different importance between bigrams (B) and co-occurring terms (C_w). This confirms our intuition that these two types of term dependencies should be treated in different ways. This supports our extension from the WSD model.

Table 5-5. Test the average importance of unigram, bigram and biterm in different windows

Best weight	$\boldsymbol{U}$	$\boldsymbol{B}$	$\boldsymbol{C_2}$	$\boldsymbol{C_4}$	$\boldsymbol{C_8}$	$\boldsymbol{C_{16}}$
Disk1	1.00	0.07	0.05	0.03	0.00	0.00
Disk2	1.00	0.10	0.05	0.07	0.00	0.01
Disk4	1.00	0.07	0.05	0.07	0.02	0.04
Disk5	1.00	0.10	0.03	0.20	0.00	0.00
WT10g	1.00	0.15	0.10	0.05	0.01	0.05
Average	79.1%	7.8%	4.4%	6.6%	0.5%	1.6%

5.5.2.3 Experiment result of DDM: using learnt weights of term pairs

In DDM, we set the fixed weight (1.0) to unigram terms and assign various weights to individual term pairs. The weights of term pairs are learnt by cross validation: for each collection, 9/10 of the queries are used in turn as training data while the remaining 1/10 of the queries are used as test queries. In Table 5-6, we report the average effectiveness obtained in the cross validation. Compared to the other models, we can see that this model performs generally better. The only exception is on Disk4 data, compared to PLM and $DDM\text{-}T2$. In a number of cases, the differences with the other models are statistically significant.

This result shows that the model we propose in this chapter can indeed lead to additional gains in retrieval effectiveness. Together with the previous comparisons, this result suggests that the two extensions we brought to this model, i.e. the consideration of more distant term dependencies and the weighting of individual term pairs, are indeed important factors that should be incorporated into IR models.

Notice that the weights we obtain from cross validation are far from optimal, while for the other models we tune the parameters to their best. So, the above comparison gave advantages to the other models (the $WSD\text{-}X$ in Table 5-3 is the cross-folder validation result for WSD, it shows steadily worse than WSD). In order to see the potential of a model with the above two extension, we try to determine the best weights for each individual term pair by a coordinate-level ascent search as explained in Section 5.4.2. The ideal case is shown in the last column of Table 5-6. We can see that the optimal effectiveness is far beyond what we can obtain by cross validation. This leads to two observations:

Table 5-6. Comparing DDM to baselines in MAP

Query &Coll.	**DDM (λ_U=1, $\lambda_R(\cdot)$ trained by 10-fold cross validation)**					
	MAP	**%U**	**%SD**	**%WSD**	**%PLM**	**%T2**
Disk1	**0.2489**	+4.5% ‡	+1.4%†	+0.4%	+2.6%‡	+1.2%†
Disk2	**0.2519**	+7.6% ‡	+1.6%†	+0.6%	+2.9%‡	+1.3%†
Disk4	0.1979	+7.3%	+1.2%	+0.3%	-1.6%	-3.6%
Disk5	**0.2500**	+5.7% ‡	+1.6%†	+1.0%	+1.3%	+1.0%
WT10g	**0.2255**	+10.4%‡	+3.9%†	+1.9%	+3.3%	+1.4%

- Our parameter tuning is not done at its best. Better parameters can be learnt. To do this, new features may be required and new learning methods may be necessary.
- Our model has a large potential not yet fully exploited (see the ideal result in Table 5-8). With better features and a better learning method, the proposed model can lead to even better results.

These are the elements that we will further examine in our future studies.

5.5.2.4 *Using manually selected phrases vs arbitrary term pairs*

Here, we exam whether using manually selected phrases is better than using arbitrary term pairs in dependency model. A query usually does not consist of strict phrase. So, our manually selected phrases are more flexible, such as “anti smooking”, “oil spill”, “human life”. We marked total 312 different phrases (term pair) from all queries. For each query set, the selected phrases number in the query set and average phrase number for each query are listed in Table 5-7.

Table 5-7. Compare using manually select phrase (M.S.P) to using all pairs in DDM

Query &Coll.	# M.S.P	#M.S.P per query	DDM-T2 (best fixed λ_R)			DDM ($\lambda_U = 1$, learnt λ_R)		
			All pairs	Selected pairs		All pairs	Selected pairs	
			MAP	MAP	%All pair	MAP	MAP	%All pair
Disk1	160	0.80	0.2458	0.2465	-0.3%	0.2489	0.2476	-0.5%
Disk2	160	0.80	0.2486	0.2470	-0.6%	0.2519	0.2492	-1.1%
Disk4	74	0.74	0.2053	0.1899	-7.5%	0.1979	0.1919	-3.0%
Disk5	92	0.61	0.2469	0.2471	+0.1%	0.2500	0.2457	-1.7%
Wt10g	37	0.37	0.2223	0.2119	-4.7%	0.2254	0.2066	-8.3%

We compare using selected phrases and using arbitrary phrases (all phrases) in two models: DDM-$T2$ and DDM (λ_U=1, other λs are learnt). The results show using only manually selected phrases in dependency models performs worse steadily than using all term pairs. Therefore, consider only natural phrase is neither necessary nor sufficient.

5.5.2.5 *Results using fixed unigram weight vs. learned unigram weight*

In our model, we allow using variable unigram weights as well as bigrams and co-occurrences. In the previous experiments, we simply set the unigrams weights to 1, and only focus on learning the weight of term pair. To enable variable unigram weights, we use the same 10-fold cross validation method for all unigrams, bigrams, and co-occurrences. The results in Table 5-8 show variable unigrams can get some improvement only for some collections. However, for some other collections, the results become worse. Two possible reasons may lead to the result:

- We use title as our query. It is short and every query term is important, especially for the query which only includes keywords. So given each unigram a fixed higher weight ($\lambda_U = 1.0$) is good enough.
- The training data and features are not enough to represent the importance of each unigram. The inaccurate learnt weight or disproportion weights of unigram will harm the performance.

From the ideal results (assign best weight to individual term or term pair), we notice that the DDM models have large potential rooms, especially for DDM with variable unigram weights.

Table 5-8. Compare DDM (fixed λ_U) to DDM$^+$ (variable λ_U)

Query &Coll.	**DDM ($\lambda_U = 1$, various for other λs)**		**DDM$^+$ (various all λs)**					
	Learnt	**Ideal**	**Learnt**				**Ideal**	
	MAP	**MAP**	**MAP**	**% U**	**%WSD**	**%DDM**	**MAP**	**%DDM**
Disk1	0.2489	*0.2944*	**0.2533**	+6.3%	+2.2%	**+1.8%**	*0.3202*	+8.8%
Disk2	0.2519	*0.3044*	**0.2534**	+8.3%	+1.2%	**+0.6%**	*0.3326*	+9.3%
Disk4	0.1979	*0.2386*	**0.1988**	+7.7%	+0.7%	**+0.7%**	*0.2685*	+12.5%
Disk5	0.2500	*0.2903*	**0.2435**	+3.0%	-1.6%	**-2.6%**	*0.3114*	+7.3%
WT10g	0.2254	*0.2749*	**0.2215**	+8.5%	+0.1%	**-1.7%**	*0.3043*	+10.7%

For short queries, the query terms are usually carefully selected for IR. All the unigrams should be same impartment. The learnt weights of unigram may be inaccurate (assigned a lower weight) sometime due to less training data. To verify this assumption, we do the same experiment for long query. We use the "DESC" part of topic as long query, and test on 5 TREC/WT10G collections: disk1, disk2, disk4, disk5, wt10g.

Table 5-9. Description of long queries vs. short queries

Query &Coll.	Topics	Num of queries	Avg. length of long query (DESC)	Avg length of short query (TITLE)
Disk1/Disk2	1-200	199	9.2	3.7
Disk4	251-351	100	8.9	2.9
Disk5	301-450	150	7.9	2.4
Wt10g	451-550	97	5.9	2.5

Table 5-10 The results of the baseline models for long query. The symbols ‡ and † mean statistical significance with t-test at $p<0.01$ and $p<0.05$ level.

Query &Coll.	Uni-gram	MRF-SD (best $\lambda_T, \lambda_O, \lambda_U$)		WSD (best param. trained on the test queries)		WSD-X (10-fold cross validation)			PLM (best para, λ trained on the test queries)	
	MAP	MAP	% U	MAP	% SD	MAP	% SD	%WSD	MAP	% SD
Disk1	0.2203	0.2281	+3.5%‡	0.2385	+4.6%‡	0.2379	+4.3%‡	-0.2%	0.2202	-3.5%‡
Disk2	0.2152	0.2307	+7.2%‡	0.2428	+5.2%‡	0.2429	+5.3%‡	+0.0%	0.2214	-4.1%‡
Disk4	0.1643	0.1850	+12.6%†	0.1785	-3.5%	0.1696	-8.3%†	-5.0%†	0.1798	-2.8%
Disk5	0.2171	0.2250	+3.7%‡	0.2291	+1.8%	0.2260	+0.4%	-1.3%	0.2191	-2.7%†
WT10g	0.1911	0.2044	+6.9%‡	0.2137	+4.5%†	0.2097	+2.6%	-1.9%	0.2008	-1.8%

Table 5-11. Compare DDM (fixed λ_U), DDM$^+$ (variable λ_U) and base models. The symbols ‡ and † mean statistical significance with t-test at $p<0.01$ and $p<0.05$ level.

Query Set	Uni-gram	MRF-SD (best $\lambda_T, \lambda_O, \lambda_U$)	W-SD (best weights)	WSD-X (10-X-V)	PLM (best para, λ)	DDM (10-X-V, fixed λ_U = 1)		DDM$^+$ (10-X-V, vary all λ)	Ideal DDM
	%DDM$^+$	%DDM$^+$	%DDM$^+$	%DDM$^+$	%DDM$^+$	MAP	%DDM$^+$	MAP	MAP
Disk1	-10.3%‡	-7.2%‡	-2.9%	-3.1%	-10.4%‡	0.2371	-3.5%‡	**0.2457**	0.3757
Disk2	-16.3%‡	-10.3%‡	-5.6%‡	-5.5%‡	-13.9%‡	0.2403	-6.6%‡	**0.2572**	0.3853
Disk4	-12.4%†	-1.4%	-4.9%	-9.6%†	-4.2%	0.1826	-2.7%	**0.1926**	0.3531
Disk5	-9.7%‡	-6.4%‡	-4.7%†	-6.0%	-8.9%‡	0.2275	-5.4%‡	**0.2404**	0.4044
WT10g	-20.5%‡	-15.0%‡	-11.1%‡	-12.8%‡	-16.5%‡	0.2171	-9.7%‡	**0.2405**	0.3935

The result shows that vary unigram weights (DDM^{+}) are much helpful for IR for long query. The query terms in long query are not as same important as in short queries. So, assigning the equal weight to all the unigrams will lead to more noise. The following analysis shows how the vary unigram weights lead to the good result:

For query 003 “Document will announce a new joint venture involving a Japanese company”. The scores of Unigram, $MRF\text{-}SD$, $W\text{-}SD$, DDM (fixed λ_U), and DDM^{+} (vary λ_U) are 0.3382, 0.3802, 0.4006, 0.3955, and 0.4108. After stop-word, the unigram weights are learnt and assigned: announce/0.37, joint/0.76, venture/0.9, japanese/1.0, company/0.48. We can find that the core words *venture* and *japanese* are assigned the larger weights, so that the DDM^{+} achieves the best result.

Another query 011 “Document discusses the goals or plans of the space program or a space project of any country or organization.” The score are 0.1074, 0.1283, 0.1292, 0.1218, and 0.1576. The unigram weights are learnt as discuss/0.25, goal/0.5, plan/0.35, space/0.8, program/0.43, project/0.6, country/0.34, organization/0.4. Assigning larger weights to the core words (*space*, *program*, and *project*) than other terms will make the result better than giving them the same weights.

5.5.3 **Experimental Results on Chinese**

We first provide the retrieval results of the baseline methods in Table 5-12. The combination parameters in $W{+}U$ and $B{+}U$ are tuned to their best. For a Chinese query $q_1q_2 \dots q_n$, we assume the word segmentation result to be $w_1, w_2, \dots w_m$. The baseline models are listed below:

- U: We use unigrams of character, and the query is “$q_1q_2 \dots q_n$”.
- B : We use bigrams of characters. The corresponding Indri query is “$\#1(q_1\ q_2)\ \dots\ \#1(q_{n-1}q_n)$” .

- BU: We use both bigrams and unigrams mixed up in a single query. The Indri query is "$\#1(q_1\ \ q_2)\ \dots\ \#1(q_{n-1}q_n)\ q_1\ q_2 \dots q_n$".
- $B{+}U$: of the scores using B and U are interpolated according to Formula (3-4).
- W: We use segmented words. The query is "$w_1\ w_2 \dots w_m$".
- WU: The segmented words are mixed up with character unigrams. The Indri query is "$w_1\ w_2 \dots w_m\ q_1, q_2 \dots q_n$".
- W+U: The scores using W and U are interpolated according to Formula (3-4).

Table 5-12. The baselines (MAP) of traditional Chinese IR models.

Query&Coll.	U	B	BU	B+U	W	WU	W+U
Trec5	0.3013	0.2696	0.3184	**0.3269**	0.2802	0.3265	0.3173
Trec6	0.3601	0.3610	0.3875	0.3878	0.3881	0.3983	**0.3998**
Trec9	0.2381	0.2119	0.2469	**0.2543**	0.1905	0.2283	0.2381
Ntcir4	0.2371	0.1995	0.2243	**0.2489**	0.2237	0.2396	0.2469
Ntcir5	0.3587	0.3151	0.3563	0.3681	0.3840	0.3817	**0.3998**
Ntcir6	0.2695	0.2448	0.2931	**0.3064**	0.2739	0.2863	0.3012

To see the importance of different type of index, we plot the results of the methods $B{+}U$ and $W{+}U$ on Trec6 and Ntcir6 collections in Figure 5-2. We can see that a reasonable interpolation usually leads to a higher effectiveness than using only one type of index (the two extremities of the curves). This shows that different types of indexes are complementary and it is useful to combine them. However, the best weight for each type of index depends on the collection and on the types of indexes combined. Indeed, the usefulness of different words and bigrams varies largely. The weight we assign to a type of index corresponds to a compromise among all the words and bigrams. As we will see in the experiment with our proposed model, it is better to assign a different weight to a word or a bigram depending on its usefulness.

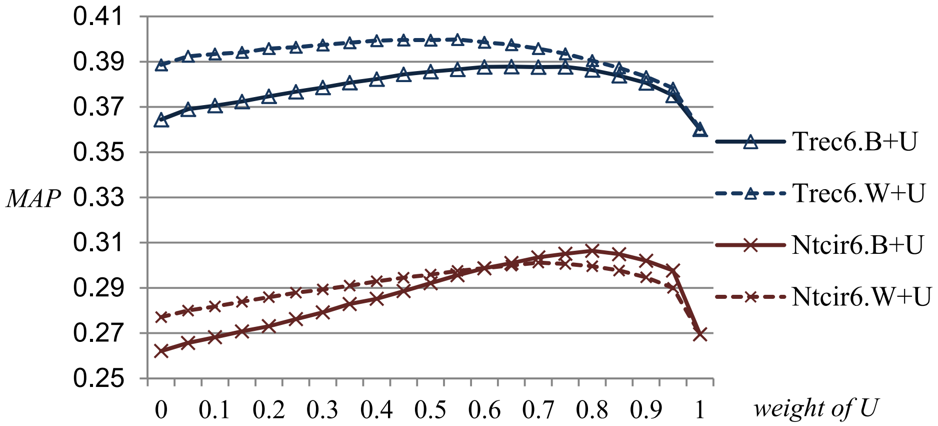

Figure 5-2. Compare the MAP of B, U, W, and their interpolations on Trec6 and Ntcir6 Collections.

Table 5-13. The baselines of dependency models: MRF-SD and WSD.

Query &Coll.	MRF-SD (best weight)				WSD (best weight)			WSD-X (10-folder cross validation)		
	MAP	% U	% B+U	%W+U	MAP	%U	%SD	MAP	%SD	%WSD
Trec5	0.3271	+8.6‡	+3.1	+3.1	0.3279	+8.8‡	+0.2	.3234	-1.1	-1.4
Trec6	0.3899	+8.3‡	+0.6	-2.5	0.3780	+5.0	-3.1	.3649	-3.1	-3.5
Trec9	0.2576	+8.2	+1.3	+8.2	0.2732	+14.8†	+6.0	.2556	+6.0	-6.4
Ntcir4	0.2490	+5.0†	+0.0	+0.8	0.2514	+6.0‡	+1.0	.2458	+1.0	-2.2
Ntcir5	0.3846	+7.2 ‡	+4.5	-3.8	0.3909	+9.0†	+1.6	.3713	+1.6	-5.0
Ntcir6	0.3066	+13.8‡	+0.0	+1.8	0.3088	+14.6‡	+0.7	.3092	+0.7	+0.1

In Table 5-13, we show the effectiveness with other baselines – MRF-SD and Weighted MRF-SD (WSD). For $MRF\text{-}SD$, we use a grid search to find the best parameters $\lambda_T, \lambda_O, \lambda_U$ so as to maximize MAP for each collection. Therefore, the effectiveness of this model is tuned to its best. The results with $MRF\text{-}SD$ are slightly better than $B+U$. Indeed, if we remove the unordered part, the $MRF\text{-}SD$ becomes identical to $B+U$. The difference between $MRF\text{-}SD$ and $B+U$ corresponds to the contribution of unordered unigram pairs. The WSD model is slight better than $MRF\text{-}SD$ except on Trec6. However, the differences between the two models are not statistically significant.

Table 5-14. The DDM results (fixed $\boldsymbol{\lambda_U}$ to 1).

Query &Coll.	DDM-T2 (best fixed λs)			DDM (λ_U=1, various other λs by 10-fold cross-validation)						
	MAP	%U	%SD	MAP	%U	%B+U	%W+U	%SD	%WSD	%T2
Trec5	0.3278	+8.8‡	+0.2	**0.3420**	+13.5‡	+4.6†	+7.8†	+4.6†	+4.3	+4.3†
Trec6	0.3916	+8.7‡	+0.4	**0.4171**	+15.8‡	+7.6‡	+4.3	+7.0‡	+10.4‡	+6.5†
Trec9	0.2627	+10.4	+2.0	**0.2793**	+17.3†	+9.8†	+17.3†	+8.4†	+2.2	+6.3
Ntcir4	0.2503	+5.5‡	+0.5	**0.2605**	+9.8‡	+4.7†	+5.5†	+4.6†	+3.6†	+4.1†
Ntcir5	0.3851	+7.4‡	+0.1	**0.3964**	+10.5‡	+7.7	-0.9	+3.1	+1.4	+2.9
Ntcir6	0.3070	+13.9‡	+0.1	**0.3176**	+17.9‡	+3.6†	+5.5‡	+3.6†	+2.9	+3.5†

The results in Table 5-14 show that the $DDM\text{-}T2$ model (DDM with fixed weights) is slightly better than $MRF\text{-}SD$. This is due to the fact that we added non-adjacent co-occurring characters. In Table 5-13 we show the effectiveness with other baselines – $MRF - SD$ and Weighted $MRF - SD$ (WSD). For $MRF - SD$, we use a grid search to find the best parameters $\lambda_T, \lambda_O, \lambda_U$ so as to maximize MAP for each collection. Therefore, the effectiveness of this model is tuned to its best. The results with $MRF - SD$ are slightly better than $B + U$. Indeed, if we remove the unordered part, the $MRF - SD$ becomes identical to $B + U$. The difference between $MRF - SD$ and $B + U$ corresponds to the contribution of unordered unigram pairs. The WSD model is slight better than $MRF - SD$ except on Trec6. However, the differences between the two models are not statistically significant.

In Table 5-13, we show the effectiveness with other baselines – $MRF\text{-}SD$ and Weighted $MRF\text{-}SD$ (WSD). For $MRF\text{-}SD$, we use a grid search to find the best parameters λ_T, λ_O, λ_U so as to maximize MAP for each collection. Therefore, the effectiveness of this model is tuned to its best. The results with $MRF\text{-}SD$ are slightly better than $B{+}U$. Indeed, if we remove the unordered part, the $MRF\text{-}SD$ becomes identical to $B{+}U$. The difference between $MRF\text{-}SD$ and $B{+}U$ corresponds to the contribution of unordered unigram pairs. The WSD model is slight better than $MRF\text{-}SD$ except on Trec6. However, the differences between the two models are not statistically significant.

The Results in Table 5-14 show that the $DDM\text{-}T2$ model (DDM with fixed weights) is slightly better than $MRF\text{-}SD$. This is due to the fact that we added non-adjacent co-occurring characters.

When we vary the weights of the bigram and the pair of co-occurring characters, the result becomes much better. In general, our model outperforms all the baseline methods except in one case. Many of the improvements are statistically significant. In comparison to $B{+}U$, $W{+}U$, $MRF\text{-}SD$ and $DDM\text{-}T2$, this result shows the benefit of assigning variable importance to pairs of characters. The result clearly validates the general approach we used in our model.

Notice again that in the above comparison, we gave considerable advantage to the baseline models, as their parameters are tuned to their best (see the difference of $WSD\text{-}X$ with cross folder validation in Table 5-13, it is steadily worse than WSD), which is not the case for our model.

In the previous Chinese experiences, we only variable the weights of bigrams and co-occurrences, unigrams weights are fixed to 1. Now we try to variable all weights in the DDM, and the results list in Table 5-15. Same to the English result, for short queries, the DDM^{+} (DDM with variable unigrams) get the marginable improvement than DDM with fixed unigram weight. It is confirmed that the DDM model with fixed unigram weighs ($\lambda_U = 1.0$) can perform as good as DDM^{+} with vary λ_U for short queries.

Table 5-15. Comparing DDM (fixed λ_U) to DDM$^+$ (variable λ_U)

Query &Coll.	DDM ($\lambda_U = 1$, various for other λs)		DDM$^+$ (various all λs)					
	Learnt	*Ideal*	Learnt				*Ideal*	
	MAP	MAP	MAP	% U	%WSD	%DDM	MAP	%DDM
Trec5	0.3420	*0.4633*	**0.3470**	+15.2‡	+5.8†	**+1.4**	0.5110	*+10.3*‡
Trec6	0.4171	*0.5515*	**0.4270**	+18.6‡	+9.5†	**+2.4**	0.5926	*+7.4*‡
Trec9	0.2793	*0.4060*	**0.2791**	+17.3	+2.2	**-0.1**	0.4634	*+14.1*‡
Ntcir4	0.2605	*0.3829*	**0.2570**	+8.4‡	+2.2	**-1.4**	0.3986	*+4.1*‡
Ntcir5	0.3964	*0.5478*	**0.3833**	+6.9‡	-1.9	**-3.3**	0.5719	*+4.4*‡
Ntcir6	0.3176	*0.4312*	**0.3318**	+23.1‡	+7.5‡	**+4.5**†	0.4570	*+6.0*‡

In order to have an idea of the potential of our model, we also show the effectiveness of DDM and DDM^{+} using the best parameters (best weights for each unigram, bigram and pair of characters). We can see that latter model with ideal parameters is better than the former one, and both of these two models can potentially largely outperform the existing models.

5.5.4 **Analysis and Discussion**

We have used the assumption that different pairs of terms should be weighted in different ways. Let us provide some concrete examples containing two terms to support it here. Let us examine $DDM\text{-}T1$, which include three component models – unigram, bigram and co-occurring terms. We fix the weight of the unigram model at 1 and vary the weights of the two other component models to see the impact on the following queries (with stopwords removed): “death cancer”, “black Monday”, “drug approval”. Figure 5-3 shows the variation in MAP along with the changes in the weights.

What we can see is that for “death cancer”, the best effectiveness can be obtained when both the bigram and co-occurrence models are weighted low (near 0). This shows that this query is best treated by the unigram model only. In other words, the dependency between the terms in this query is not important.

On the other hand, for “black Monday”, the weights of the bigram and co-occurrence models should be tuned high, showing that it is important to consider this query as a bigram and co-occurring terms. Indeed, this query corresponds to a specific expression, which should be considered as such. Any transformation (e.g. separation of the terms) would alter greatly the meaning. This explains the high weights that we should assign to the bigram and co-occurrence models.

For the third query “drug approval”, the bigram model should not be assigned a strong weight, while the co-occurrence model should. This query corresponds to an expression in which the two terms are strongly dependent; however, they do not form a specific expression. For example, a document containing “the approval of anti-depression drug” can also be relevant. Therefore, some flexibility should be allowed when matching the query with documents. Such flexibility is allowed in the co-occurrence model. This is why the co-occurrence model should be assigned a strong weight.

The above three cases are typical in IR. They correspond to the three cases we mentioned in Section 5.1. These examples clearly confirm our intuition that a component model does not have the same impact on different queries.

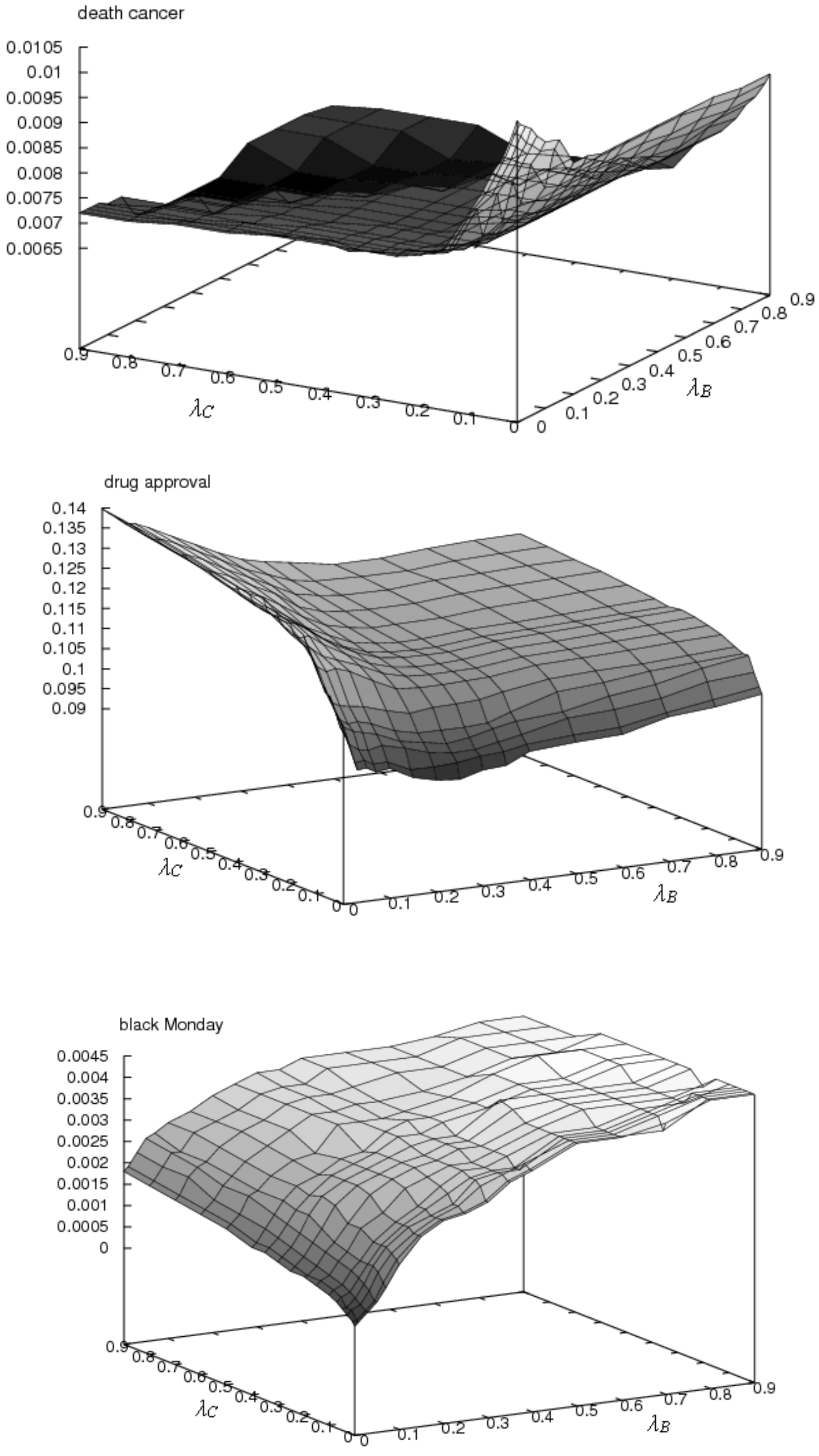

Figure 5-3. Three typical performance curves: independent (death cancer), dependent in ordered adjacent (black Monday), and dependent no matter the order and adjacency (drug approval). The importance weight of unigram is set to 1, bigram and co-occurrence weights vary.

In the following tables, we show why our model, by applying automatically learnt weights, performs better than the baseline model $MFR\text{-}SD$. Recall that for $MFR\text{-}SD$, we tuned the weights to their best. In the columns "DDM" and "Ideal", the component models with the weight 0 are not shown, and the weights are normalized.

In Table 5-16 and Table 5-17, the learnt weights are better than best fixed weights of $MRF\text{-}SD$ – the weights are closer to the ideal ones. However, the learnt weights in Table 5-18 are further away from the ideal weights, leading to a lower effectiveness. The last case shows that our learning method is not capable to determine the weights correctly in all the cases. This leaves much room for improvement in the future.

Table 5-16. Topic 121 "death cancer" for Collection Disk1

	MRF-SD	**DDM**	**Ideal**
Weights	$.90U\ .08B\ .02C_8$	$.98U\ \ .02B$	$1.0U$
MAP	0.0088	**0.0104**	0.0105

Table 5-17. Topic 105 "black Monday" for Collection Disk1

	MRF-SD	**DDM**	**Ideal**
Weights	$.9U\ .08B\ .02C_8$	$.71U\ \ .29B$	$.40U\ .32B\ .28C_2$
MAP	0.0016	**0.0034**	0.0059

Table 5-18. Topic 014 "drug approval" for Collection Disk1

	MRF-SD	**DDM**	**Ideal**
Weights	$.9U\ .08B\ .02C_8$	$.95U\ \ .02B$	$.28U\ .24C_4\ .24C_8\ .24C_{16}$
MAP	**0.1035**	0.0977	0.1505

Our model can also capture the dependencies between non-adjacent query terms. For some queries, such as "recycle automobile tire" (see Table 5-19), "recycle" and "tires" have a strong dependency. Ignoring it (as $MRF\text{-}SD$ and WSD do) will reduce the effectiveness.

Table 5-19. Topic 419 "recycle automobile tire" for Collection Disk5

	MRF-SD	**DDM**	**Ideal**
recycle automobile	$.88U$ $.09B$ $.03C_8$	$.92U$ $.08B$	$1.0U$
recycle tire	N/A	$.82U$ $.08C_2$ $.10C_{16}$	$.49U$ $.17B$ $.34C_2$
automobile tire	$.88U$ $.09B$ $.03C_8$	$.86U$ $.13B$ $.01C_2$	$.48U$ $.23B$ $.17C_4$ $.12C_{16}$
MAP	0.1873	**0.2065**	0.2589

For Chinese IR, setting proper weights to character pairs (high weights to useful pairs, low weights to noisy pairs), our model can benefit from the strengths of unigram model and dependency model, and avoid the disadvantages of them.

- Unigrams (characters) are useful for matching synonyms, near-synonyms or various forms of transliterations due to the characters they share. For example, the two variants of AIDS 爱滋病 and 艾滋病 can be partly matched because they share two characters 滋 (grow, multiply) and 病 (disease). In our experiments, for the query Ch73 in Trec9: "中国的艾滋病" (AIDS in China), the average precision (AP) using words is close to 0 because the documents use a different variant of AIDS - 爱滋病. On the other hand, using unigrams, we obtain an AP of 0.3344. Using DDM, we obtain an AP of 0.4070. In DDM, we observe that except for the bigrams 艾滋 and 滋病, the weights of other bigrams and co-occurring character pairs are close to 0. This means that our model heavily relies on unigrams for this query. However, as some of the bigrams (in particular, the bigram 滋病) have a non-zero weight, they help enhance the connections between these characters. This explains the improved effectiveness of DDM over unigram model.
- On the other hand, characters that are highly ambiguous should be combined and our model can successfully make use of dependencies in these cases. For example,

in the query Ch27 of Trec5 “中国 (China) 在(in) 机器人(robotics) 方面(area) 的 (of) 研制(research)”, if we use unigrams, both the terms 中国(China) and 机器人 (robot) are decomposed into very common characters 中(China, middle), 国 (country), 机 (machine, engine), 器 (machine, utensil), 人 (human, person). These latter lead to a low effectiveness of 0.1057. When words are used the average precision is increased to 0.4079. Although our DDM model is unable to decide to rely entirely on words in this case, it still assigns a quite strong relative importance to the words, leading to an average precision of 0.3030. The highly ambiguous characters are indeed put into dependencies as follows: 中国 (with a weight of 0.64), 器人 (0.59). These strong weights help solve the ambiguity problem of separate characters.

Our model can capture the dependencies between non-adjacent characters.

- For the query 003 of Ntcir4 “胚胎 (embryonic) 干细胞 (stem cells)”, we obtain an AP of 0.1891 using unigrams, 0.2174 using $MRF\text{-}SD$, and 0.2410 using WSD, while our DDM model results in an AP of 0.4096. The good performance of DDM is due to the fact that strong dependencies between non-adjacent characters are captured. In this case, we observe strong weights for the co-occurring characters 胎 and 干 (with a weight of 0.22), 胚 and 干 (0.54), 胎 and 胞 (0.27). These pairs do not correspond to legitimate words in this query, but their combinations tend to enhance the relationship between the words 胚胎 and 干细胞. We can see that co-occurring characters can also successfully capture relationships between different words.

The above examples illustrate why the two extensions to the previous dependency models we propose in this chapter can lead to gains in the retrieval effectiveness.

5.6 Conclusions

Terms in documents and queries are often dependent. A model that ignores term dependencies is prone to retrieve much noise, terms can have different meanings in different contexts. On the other hand, a model that treats all terms as equally dependent also runs the danger of connecting terms that are not strongly dependent and imposes such a false dependency as a requirement in the retrieval process. As a result, such a model may miss documents where the false dependency does not appear. If one treats all dependencies of the same kind in a unique way (i.e. by assigning a unique weight) as being done in most previous models, one will end up in assigning a moderate unique weight to all dependencies because of the above danger. The real problem is that term dependencies vary largely: a pair of terms such as “black Monday” is strongly dependent, and the consideration of the dependency in the retrieval process is highly beneficial; while other pairs of terms (e.g. “death cancer”) have weaker dependencies and can be treated separately. Therefore, each pair of terms should be treated in its own way according to the strength of the dependency and the usefulness of considering the pair of terms together. The approach proposed in this chapter goes in this direction.

Our model extends the existing dependency models on two following aspects:

- We assign weights to individual pairs of terms rather than to a type of dependency;
- We consider dependencies between terms of further distance, and different distances are also treated separately.

We tested our model and compared it to existing ones on several TREC and NTCIR collections for English and Chinese IR. Our experimental results showed that our model can consistently outperform existing approaches. In a number of cases, the improvements are statistically significant. While we cannot conclude our implementation fully exploited the potential of the model (because of the limitation in the learning process), it is clear that the model could potentially be significantly better than state-of-the-art methods. The results differences between our implementation and the ideal case can lead to future study in the following aspects: (1) extend the set of features and try other learning methods. (2) test on a larger amount of training data including query logs, user profiles, and click through data.

CHAPTER 6.
DISCUSSIONS AND CONCLUSION

Queries terms in IR are often dependent. A model that ignores the term dependency or simply considers all dependent terms in equal weight is prone to retrieving much noise. As a result, such a model may assign a low value to a relevant document without calculating the dependency of terms, or assigning a high value to an irrelevant document if over emphasized dependency in the document. Each pair of terms should be treated in its own way according to its strength of the dependency and the usefulness.

In this thesis, we tested several methods to capture term dependency for monolingual IR and cross-lingual IR as well. In CLIR, the dependency of source language terms needs to reflect in target language terms. We proposed three approaches to integrate the dependency: combination model of using language models, Dempster-Shafer theory based model, and discriminative language model.

Firstly, we tested combination approach on Chinese collection under language modeling framework. We tried the following index units: unigram character (U), bigram character (B), segmented word (W), mix of bigram and unigram (BU), mix word of unigram (WU), and the combinations of $W+U$, $B+U$, $W+B+U$. Results show that the combination approach lead to better retrieval effectiveness than using any single index unit, consistent with previous studies. We also found that Chinese unigrams are even more effective than either words or bigrams. For CLIR, we have chosen to use bigrams and unigrams as alternative translation units. Our experiments showed that these translation units are as effective as words. We observed only slightly higher retrieval effectiveness when combining unigrams and words/bigrams translations and using translations from English word pairs.

Our conclusions are (1) Chinese characters are very meaningful units, which can be used as both indexing and translation units (2) The linear combination with different index units is more effective than using single one, but the increase for CLIR is marginal. Further improvement can be done in the following aspects: consider strength of link between English words; try another way other than linear combination of different index units.

Secondly, we followed the principle of Dempster-Shafer theory and assigned probabilities to sets of terms instead of to their components separately. This consideration can well capture the phenomenon that terms strongly overlap in its individual form and in combined form. The same occurrence of a string can be simultaneously considered as that of a long term and that of shorter terms included in it. The approach allows us to avoid duplicating the occurrence artificially, resulting in a more principled way to estimate probability. We extended Dempster-Shafer's belief and plausibility functions to a general transfer function $t(A|B)$ by considering the possible relationships between term sets under specific characteristics about the language. This resulted in a more suitable matching function for IR.

We tested our model with several simple functions on Chinese IR. Results strongly suggest that the method we proposed is more suited for Chinese IR than state-of-the-art approaches. In particular, it can better take into account the overlapping nature of Chinese compound terms and simple terms, and cope their relationships during probability assignment. Although, more complex functions can be defined by choosing better criteria, especially when deal with term ambiguity. In addition, the transfer function can rely on richer linguistic knowledge rather than just character overlapping or term distributions.

This model can be also apply on European languages, which can be a worth area explorers in future studies.

The limitation of this model is that only term relations within the term-set are considered, while dependency between term sets is not well considered. It is also the problem we will solve next.

Thirdly, we proposed a discriminative language model for handling pairwise term dependencies according to their dependency stretch and usefulness in IR. Our model extends the existing dependency models in the two following aspects: (1) assigning weights to individual pairs of terms rather than to a type of dependency (2) considering dependencies between terms in further distance, and different distances are also treated separately. We tested our model on several TREC and NTCIR collections for English and Chinese IR. Experimental results showed that our model can consistently outperform existing approaches. The ideal case shows that the model has a great potential to be significantly better than state-of-the-art methods.

In conclusion, capturing term dependencies and taking into account of the dependency strength and usefulness are more helpful to IR. The discriminative language model we proposed can effectively integrate term dependency factors leading to good IR results. The difference between result in our implementation and in the ideal case suggests that the approach can be improved further, including:

- The set of features used to determine the weights on pairs of terms could be extended;
- Other learning methods to train importance of dependencies need to try;
- Finally, we may need a larger amount of training data. Query logs with user interactions (click-throughs) could be a valuable resource.

BIBLIOGRAPHY

R. Baeza-Yates and B. Ribeiro-Neto (1999). *Modern Information Retrieval*, Adison Wesley.

J. Bai, J-Y. Nie and G. Cao (2006). Context-dependent term relations for information retrieval. *EMNLP*, pp.551-559.

J. Bai, D. Song, P. Bruza, J-Y. Nie and G. Cao (2005). Query Expansion Using Term Relationships in Language Models for Information Retrieval. *CIKM*, pp.688-695.

L. Ballesteros and W. B. Croft (1998). Resolving ambiguity for cross-language retrieval. *SIGIR*, pp.64-71.

M. Bendersky and W. B. Croft (2012). Modeling higher-order term dependencies in information retrieval using query hypergraphs. *SIGIR '12*, pp.941-95.

M. Bendersky, D. Metzler and W. B. Croft (2010). Learning Concept Importance Using a Weighted Dependence Model. *Proceedings of the third ACM international conference on Web search and data mining*, pp.31-40.

C. M. Bishop (2006). *Pattern Recognition and Machine Learning*, Springer.

P. F. Brown, S. AD. Pietra, V. JD. Pietra and R. L. Mercer (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics*, vol 19, no.2, pp.263-311.

C. Buckley and E. M. Voorhees (2000). Evaluating evaluation measure stability. *ACM SIGIR*, pp.33-40.

C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton and G. Hullender (2005). Learning to rank using gradient descent. *ICML '05*, pp.89-96.

C-C. Chang and C-J. Lin 2001, LIBSVM: a Library for Support Vector Machines, http://www.csie.ntu.edu.tw/~cjlin/libsvm/.

S. F. Chen and J. Goodman (1999). An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, vol 13, pp.35-394.

K. Chen and S. Kiu (1992). Word identification for Mandarin Chinese sentences. *The 5th International Conference on Computational Linguistics*, pp.101-107.

J. Chen and J-Y. Nie (2000). Automatic construction of parallel English-Chinese corpus for cross-language information retrieval. *ANLP* Seattle, Washington, pp.21-28.

T-H. Chiang, J-S. Chang, M-Y. Lin and K-Y. Su (1992). Statistical models for word segmentation and unknown resolution. *ROCLING-92*, pp.123-146.

L. F. Chien (1995). Fast and quasi-natural language search for gigabytes of Chinese texts. *Proceedings of the 18h annual international ACM SIGIR conference on Research and development in information retrieval*, pp.112-120.

F. Crestani and C. J. van Rijsbergen (1995). Information retrieval by logical imaging. *Journal of Documentation*, vol 51, no.1, pp.3-17.

W. B. Croft (2003). Language Models for Information Retrieval. *Proceeding of the 19th International Conference on Data Engineering*, pp.3-7.

W. B. Croft, H. R. Turtle and D. D. Lewis (1991). The use of phrases and structured queries in information retrieval. *Proceedings of the 14th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.32-45.

R. Cummins and C. O'Riordan (2009). Learning in a pairwise term-term proximity framework for information retrieval. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp.251-258.

M. W. Davis and W. C. Ogden (1997). QUILT: implementing a large-scale cross-language text retrieval system. *SIGIR-1997*, pp.92-98.

L. M. de Campos, J. M. Fernández-Luna and J. F. Huete (2000). Building Bayesian Network-Based Information Retrieval Systems. *Proceedings of the 11th International Workshop on Database and Expert Systems Applications*, pp.543-550.

L. M. de Campos, J. M. Fernández-Luna and J. F. Huete (2003). The BNR model: foundations and performance of a Bayesian network-based retrieval model. *International Journal of Approximate Reasoning*, vol 34, no.2-3, pp.265-285.

A. P. Dempster (1968). A Generalization of Bayesian Inference. *Journal of the Royal Statistical Society, Series B*, vol 30, pp.205-247.

T. Dunning (1993). Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, vol 19, no.1, pp.61-74.

D. A. Evans and C. Zhai (1996). Noun-Phrase Analysis in Unrestricted Text for Information Retrieval. *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pp.17-24.

J. L. Fagan (1987). Automatic phrase indexing for document retrieval. An examination of syntactic and non-syntactic methods. *ACM SIGIR-1987*, pp.91-101.

J. Gao (2005). Linear Discriminant Model for Information Retrieva. *SIGIR-200*, pp.290-297.

J. Gao, X. He and J-Y. Nie (2010). Clickthrough-based translation models for web search: from word models to phrase models. *CIKM*, pp.1139-1148.

J. Gao, M. Li, A. Wu and C-N. Huang (2005). Chinese word segmentation and named entity recognition: A pragmatic approach. *Computational Linguistics*, vol 31, no.4, pp.531-574.

J. Gao, J-Y. Nie, H. He and W. Chen (2002). Resolving query translation ambiguity using a decaying co-occurrence model and syntactic dependence relations. *SIGIR-2002*, pp.183-190, 2002.

J. Gao, J-Y. Nie, G. Wu and G. Cao (2004). Dependence language model for information retrieval. *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.170-177.

J. Gao, J-Y. Nie, E. Xun, J. Zhang, M. Zhou and C. Huang (2001). Improving Query Translation for Cross-Language Information Retrieval using Statistical Models. *SIGIR-2001*, pp.96-104.

J. Gao, H. Qi, X. Xia and J-Y. Nie (2005). Linear Discriminant Model for Information Retrieva. *SIGIR-2005*, pp.290-297.

J. Hajič and B. Hladká (1998). Tagging Inflective Languages: Prediction of Morphological Categories for a Rich, Structured Tagset. *COLING-ACL*, pp.483-490.

D. Hawking and P. Thistlewaite (1995). Proximity Operators - So Near And Yet So Far. *TREC*

Y. Hou, X. Zhao, D. Song and W. Li (2013). Mining pure high-order word associations via information geometry for information retrieval. *ACM Transactions on Information Systems*, vol 31, no.3, article 12.

X. Huang, F. Peng, D. Schuurmans, N. Cercone and S. E. Robertson (2003). Applying Machine Learning to Text Segmentation for Information Retrieval. *Information Retrieval*, vol 6, no.3, pp.333-362.

X. Huang, S. E. Robertson, N. Cercone and A. An (2000). Probability-Based Chinese Text Processing and Retrieval. *Computational Intelligence*, vol 16, no.4, pp.552-569.

K. Järvelin and J. Kekäläinen (2000). IR evaluation methods for retrieving highly relevant documents. *SIGIR*, pp.41-48.

K. Järvelin and J. Kekäläinen (2002). Cumulated gain-based evaluation of ir techniques. *ACM Transactions on*, vol 20, no.4, p.422–446.

F. V. Jensen (1996). *An Introduction to Bayesian Networks.*, Springer Verlag, New York.

G. Jianfeng, K. Toutanova and W-T. Yih (2011). Clickthrough-based latent semantic models for web search. *SIGIR*, pp.675-684.

R. Jin and JY. Chai (2005). Study of cross lingual information retrieval using on-line translation systems. *SIGIR-2005*, pp.619-620.

J. Kalervo and K. Jaana (2000). IR evaluation methods for retrieving highly relevant documents. *SIGIR*, pp.41-48.

J. Kanis and L. Müller (2005). Automatic lemmatizer construction with focus on OOV words lemmatization. *TSD'05 Proceedings of the 8th international conference on Text, Speech and Dialogue*, pp.132-139.

E. M. Keen (1992). Some aspects of proximity searching in text retrieval systems. *Journal of Information Science*, vol 18, no.2, pp.89-98.

W. Kraaij, JY. Nie and M. Simard (2003). Embedding Web-based statistical translation models in cross-language information retrieval. *Computational Linguistics*, vol 29, no.3, pp.381-419.

D. H. Kraft and D. A. Buell (1983). Fuzzy Sets and Generalized Boolean Retrieval Systems. *International Journal on Man-Machine Studies Vol.19* , pp.45-56.

R. Krovetz (1993). Viewing Morphology as an Inference Process. *ACM-SIGIR*, p.191–203.

K. L. Kwok (1997). Comparing representations in Chinese information retrieval. *Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.34-41.

J. Lafferty, D. Sleator and D. Temperle (1992). Grammatical Trigrams: A Probabilistic Model of Link Grammar. *AAAI Fall Symposium on Probabilistic Approaches to Natural Language*, pp.89-97.

FW. Lancaster and EG. Fayen (1973). *Information Retrieval On-Line*, Melville Publishing Co., Los Angeles, California.

F. Lepage (2001). Partial Probabilistic Interpretations and General Imaging. *Proceedings of the 12th International Workshop on Database and Expert Systems Applications*, pp.254-258.

T. Liang, S-Y. Lee and W-P. Yang (1996). Optimal weight assignment for a Chinese signature file. *Information Processing and Management: an International Journal*, vol 32, no.2, pp.227-237.

N. Y. Liang and Y-B. Zhen (1991). A Chinese word segmentation model and a Chinese word segmentation system PC-CWSS. *COLIPS*, vol 1, pp.51-55.

B. Li, S. Lien, C. Sun and M. Sun (1991). A maximal matching automatic Chinese word segmentation algorithm using corpus tagging for ambiguity resolution. *R.O.C. Computational Linguistics Conference (ROCLING-IV)*, pp.135-146.

RWP. Luk, K. F. Wong and K. L. Kwok (2002). A comparison of Chinese document indexing strategies and retrieval models. *ACM Transactions on Asian Language Information Processing*, vol 1, no.3, pp.225-268.

Y. Lv and C. Zhai (2009). Positional Language Models for Information Retrieval. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp.299-306.

C. Manning, P. Raghavan and H. Schute (2008). *Introduction to Information Retrieval*, Cambridge University Press.

C. D. Manning and H. Schütze (1999). *Foundations of Statistical natural language Processing*, MIT Press.

D. Metzler and W. B. Croft (2005). A Markov Random Field Model for Term Dependencies. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.472-479.

D. Metzler and W. B. Croft (2007). Linear feature-based models for information retrieval. *Information Retrieval*, vol 10, no.3, pp.257-274.

GA. Miller (1995). WordNet: A Lexical Database for English.. *Communications of the ACM* , vol 38, no.11, pp.39-41.

D. RH. Miller, T. Leek and R. M. Schwartz (1999). A hidden Markov model information retrieval system. *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pp.214-221.

C. N. Mooers (1950). Information retrieval viewed as temporal signaling. *Proceedings of the International Congress of Mathematics, Volume 1*, pp.572-573.

R. Nallapati (2004). Discriminative Models for Information retrieval. *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.64-71.

R. Nallapati and J. Allan (2002). Capturing term dependencies using a language model based on sentence trees. *Proceedings of the eleventh international conference on Information and knowledge management*, pp.383-390.

J-Y. Nie, M. Brisebois and X. Ren (1996). On Chinese text retrieval. *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.225-233.

J-Y. Nie, J. Gao, J. Zhang and M. Zhou (2000). On the use of words and n-grams for Chinese information retrieval. *Proceedings of the fifth International Workshop on Information Retrieval with Asian Languages*, pp.141-148.

J-Y. Nie, W. Jin and M-L. Hannan (1994). A hybrid approach to unknown word detection and segmentation of Chinese. , pages 326–. *The International Conference on Chinese Computing* Sigapore, pp.326-335.

J-Y. Nie, M. Simard, P. Isabelle and R. Durand (1999). Cross-language information retrieval based on parallel texts and automatic mining of parallel texts from the Web. *SIGIR-1999*, pp.74-81.

C. D. Paice (1984). Soft evaluation of Boolean search queries in information retrieval systems. *Information Technology Research Development Applications*, vol 3, no.1, pp.33-41.

J-H. Park, W. B. Croft and D. A. Smith (2011). A Quasi-Synchronous Dependence Model for Information Retrieval. *CIKM '11*, pp.17-26.

J. Pearl (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan and Kaufmann, San Mateo.

F. Peng, F. Feng and A. McCallum (2004). Chinese segmentation and new word detection using conditional random fields. *COLING*, pp.562-568.

F. Peng, X. Huang, D. Schuurmans and N. Cercone (2002). Investigating the Relationship between Word Segmentation Performance and Retrieval Performance in Chinese IR. *COLING'02*, pp.1-7.

F. Peng and D. Schuurmans (2001). Self-Supervised Chinese Word Segmentation. *Intelligent Data Analysis, Proceedings of the Fourth International Conference (IDA-01)*, pp.238-247.

V. Plachouras and I. Ounis (2005). Dempster-Shafer Theory for a Query-Biased Combination of Evidence on the Web. *Information Retrieval*, vol 8, no.2, pp.197-218.

J. M. Ponte and W. B. Croft (1998). A language modeling approach to information retrieval. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pp.275-281.

M. Porter (1980). An Algorithm for Suffix Stripping , vol 14, no.3, pp.130-137.

B. Pôssas, N. Ziviani, W. Meira and B. Ribeiro-Neto (2002). Set-based model: a new approach for information retrieval.. *SIGIR*, pp.230-237.

R. Rao, S. K. Card, H. D. Jellinek, J. D. Mackinlay and G. G. Bobertson (1992). The information Grid: A Framework for Building Information Retrieval and Retrieval-Centered Applications. *ACM Sysmposium on User Interface Software and technology* Monterey, CA, USA

Y. Rasolofo and J. Savoy (2003). Term proximity scoring for keyword-based retrieval system. *Proceedings of the 25th European Conference on IR Research*, pp.207-218.

B. AN. Ribeiro and R. Muntz (1996). A belief network model for IR. *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.253-260.

B. Rio (2009). Part of Speech Based Term Weighting for Information Retrieval. *ECIR*

S. E. Robertson and K. Sparck Jones (1976). Relevance weighting of search terms. *Journal of the American Society for Information Science*, vol 27, no.3, pp.129-146.

S. E. Robertson and S. Walker (1994). Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. *ACM SIGIR*, p.232–241.

S. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull and A. Lau (1992). Okapi at TREC. *The first Text Retrieval Conference*, pp.21-30.

I. Ruthven and M. Lalmas (2002). Using Dempster-Shafer's Theory of Evidence to Combine Aspects of Information Use. *Journal of Intelligent Information Systems*, vol 19, no.3, pp.267-302.

G. Salton, E. A. Fox and H. Wu (1983). Extended Boolean Information Retrieval. *Communications of the ACM*, vol 26, no.12, pp.1022-1036.

G. Salton and ME. Lesk (1965). The SMART Automatic Document Retrieval System — An Illustration. *Communications of the ACM*, vol 8, no.6, pp.391-398.

G. Salton and M. J. McGill (1983). *Introduction to Modern Information Retrieval*, McGraw-Hill, New York.

G. Salton, A. Wong and CS. Yang (1975). A Vector Space Model for Automatic Indexing. *Communications of the ACM*, vol 18, no.11, pp.613-620.

G. Salton, CS. Yang and CT. Yu (1975). A Theory of Term Importance in Automatic Text Analysis.. *Journal of the American Society for Information Science*, vol 26, no.1, pp.33-44.

G. Shafer (1976). *Mathematical Theory of Evidence*, Princeton University Press.

L. Shi and J-Y. Nie (2006). Filtering or adapting: two strategies to exploit noisy parallel corpora for cross-language information retrieval. *CIKM*, pp.814-815.

L. Shi and J-Y. Nie (2007). Using unigram and bigram language models for monolingual and cross-language IR. *Proceedings of the 6th NTCIR Workshop Meeting on Evaluation of Information Access Technologies: Information Retrieval, Question Answering and Cross-Lingual Information Access*, pp.20-25.

L. Shi and J-Y. Nie (2009). Integrating phrase inseparability in phrase-based model. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp.708-709.

L. Shi and J-Y. Nie (2010A). Using Various Term Dependencies According to Their Utilities. *CIKM*, pp.1493-1496.

L. Shi and J-Y. Nie (2010B). *Modeling Variable Dependencies between Characters in Chinese Information Retrieval*, AIRS, 539-551.

L. Shi, J-Y. Nie and J. Bai (2007). Comparing different units for query translation in Chinese cross-language information retrieval. *The 2nd international conference on Scalable information systems*, article no. 63.

L. Shi, J-Y. Nie and G. Cao (2008). Relating dependent indexes using Dempster-Shafer theory. *Proceeding of the 17th ACM conference on Information and knowledge management*, pp.429-438.

F. Song and W. B. Croft (1999). A general language model for information retrieval. *Proceedings of the eighth international conference on Information and knowledge management*, pp.316-321.

M. Speretta and S. Gauch (2005). Personalizing search based on user search histories. *CIKM*, pp.622-628.

R. Sproat and C. Shih (1990). A Statistical Method for Finding Word Boundaries in Chinese Text. *Computer Processing of Chinese and Oriental Languages*, vol 4, no.4, pp.336-351.

M. Srikanth and R. Srihari (2002). Biterm language models for document retrieval. *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.425-426.

T. Tao and C. Zhai (2007). An exploration of proximity measures in information retrieval. *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.295-302.

WJ. Teahan,. Wen, R. McNab and IH. Witten (2000). A Compression-based Algorithm for Chinese Word Segmentation. *Computational Linguistics*, vol 26, no.3, pp.375-393.

A. Thanopoulos, N. Fakotakis and G. Kokkinakis (2002). Comparative evaluation of collocation extraction metrics. *Proceedings of 3rd International Conference on Language Resources and Evaluation (LREC'02)*, pp.620-625.

M. Theophylactou and M. Lalmasy 1998, A Dempster-Shafer Model for Document Retrieval using Noun Phrases, Tech Report, Dept of Computer Sciencs, University of Glasgow.

H. Turtle and W. B. Croft (1990). Inference networks for document retrieval. *Proceedings of the 13th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.1-24.

H. Turtle and W. B. Croft (1991). Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems*, vol 9, no.3, pp.187-222.

J. Urban, J. M. Jose and C. J. Rijsbergen (2006). An adaptive technique for content-based image retrieval. *Multimedia Tools and Applications*, vol 31, no.1, pp.1-28.

V. N. Vapnik (1998). *Statistical Learning Theory*, Wiley.

S. Vogel, H. Ney and C. Tillmann (1996). HMM-based word alignment in statistical translation. *COLING*, pp.836-841.

S. KM. Wong, W. Ziarko and P. CN. Wong (1985). Generalized vector spaces model in information retrieval. *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*, pp.18-25.

A. Wu (2003). Customizable segmentation of morphologically derived words in Chinese. *Computational Linguistics and Chinese Language Processing*, vol 8, no.1, pp.1-27.

N. Xue (2003). Chinese word segmentation as character tagging. *Computational Linguistics and Chinese Language Processing*, vol 8, no.1, pp.29-48.

Y. Yang (1999). An evaluation of statistical approaches to text categorization. *Information Retrieval*, vol 1, no.1-2, pp.69-90.

Y. Yang, J. G. Carbonell, R. D. Brown and R. E. Frederking (1998). Translingual information retrieval: learning from bilingual corpora. *Artificial Intelligence*, vol 103, pp.323-345.

T. Yao, G. Zhang and Y. Wu (1990). A rule-based Chinese automatic segmentation system. *Journal of Chinese Information Processing*, vol 4, no.1, pp.37-43.

C. Zhai and J. Lafferty (2001). A study of smoothing methods for language models applied to ad hoc information retrieval. *SIGIR*, pp.334-342.

H-P. Zhang, Q. Liu, X-Q. Cheng, H. Zhang and H-K. Yu (2003). Chinese Lexical Analysis Using Hierarchical Hidden Markov Model. *The Second SIGHAN Workshop*, pp.63-70.

J. Zhao and J. X. Huang (2014). An Enhanced Context-sensitive Proximity Model for Probabilistic Information Retrieval. *SIGIR'14* Queensland, Australia, pp.1131-1134.

J. Zhao, J. X. Huang and B. He (2011). CRTER: Using Cross Terms to Enhance Probabilistic Information Retrieval. *SIGIR'11*, pp.155-164.

J. Zhao, J. X. Huang and Z. Ye (2014). Modeling Term Associations for Probabilistic Information Retrieval. *ACM Transactions on Information Systems*, vol 32, no.2, article 7.

J. Zhao and Y. Yun (2009). A Proximity Language Model for Information Retrieval. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pp.291-298.