

Chinese Question Answering  
with  
Full-Text Retrieval Re-Visited

by

Yutao Guo

A thesis  
presented to the University of Waterloo  
in fulfillment of the  
thesis requirement for the degree of  
Master of Mathematics  
in  
Computer Science

Waterloo, Ontario, Canada, 2004

©Yutao Guo, 2004

I hereby declare that I am the sole author of this thesis.

I authorize the University of Waterloo to lend this thesis to other institutions or individuals for the purpose of scholarly research.

Signature

I further authorize the University of Waterloo to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

Signature

## **Borrower's Page**

The University of Waterloo requires the signatures of all persons using or photocopying this thesis. Please sign below, and give address and date.

## **Abstract**

This thesis addresses the passage retrieval and specific natural language processing (NLP) problems involved in the development of a Chinese question answering (QA) system with a re-visit of the traditional Chinese full-text retrieval task. The goal of this thesis is to investigate the applicability of the MultiText system, a collection of techniques and tools for information retrieval (IR) and QA, to languages other than English, for example, Chinese, one of whose main differences from English is that texts are written as consecutive characters without explicit word boundaries.

QA is focused on extracting small fragments of text as answers to natural language questions. Modern QA systems usually employ a pipeline architecture consisting of three main components: question analysis, IR, and answer extraction. Techniques for building English QA systems have been widely developed, but QA in Chinese has until recently drawn little attention. This is probably because of the difficulties of dealing with Chinese language characteristics, such as word segmentation, sentence structure analysis, and the recognition of non-word named entities, such as Chinese person names and numbers.

The MultiText research group had no previous experience in migrating QA techniques from English to Chinese, which motivated this thesis. In the course of our investigation of building a Chinese QA system, we found it necessary to re-visit Chinese full-text retrieval. For one reason, IR plays an important role in QA. For

another reason, the MultiText's Chinese experiments at TREC-6 only focused on full-text retrieval with manually constructed queries. Experiments with automatic queries were not conducted at that time. In addition, the new passage retrieval algorithm and answer extraction heuristics developed specifically for QA could be adapted for full-text retrieval purposes, including the ranking of full documents and pseudo-relevance feedback. While describing the incorporation of traditional and new passage retrieval techniques into Chinese text retrieval and QA, this thesis addresses approaches to specific NLP problems in Chinese text processing in depth. The evaluation results of our systems performance indicate that, with suitable modifications, the MultiText techniques are effectively applicable to Chinese text retrieval and QA as well.

## **Acknowledgements**

I would like to express my gratitude to Dr. Gord Cormack for his invaluable advice and assistance to the completion of this thesis. My work could not have existed without his dedication and support. He helped me in choosing the thesis topic, setting up experimental environments, conducting the research work, and revising this thesis.

I would like to thank Dr. Charlie Clarke for his technical explanations about the MultiText system, his efforts to crawl the Chinese Web data for me to build a Chinese text corpus, and his useful feedback on my thesis.

I am also indebted to Dr. Frank Tompa for reading my thesis carefully and finding the confusing bits and bugs.

My thanks must also be given to Thomas Lynam and Egidio Terra. My Chinese question answering interface was implemented based on Thomas' code, and my Chinese Web corpus was built with Egidio's assistance. Besides, they helped me understanding the MultiText implementation and shooting technical troubles during the period when I was conducting experiments.

Finally, special thanks must go to my parents in China for their encouragement.

# Table of Contents

|                                                                 |           |
|-----------------------------------------------------------------|-----------|
| <b>Chapter 1 Introduction</b> .....                             | <b>1</b>  |
| <b>Chapter 2 Background</b> .....                               | <b>5</b>  |
| 2.1 Document-Based Retrieval vs. Passage-Based Retrieval .....  | 5         |
| 2.2 The Text REtrieval Conference (TREC).....                   | 8         |
| 2.3 Full-Text Retrieval at TREC.....                            | 10        |
| 2.3.1 The Test Collection .....                                 | 11        |
| 2.3.2 The Evaluation Measures .....                             | 11        |
| 2.3.3 The Research Areas.....                                   | 13        |
| 2.4 Automatic Question Answering at TREC.....                   | 17        |
| 2.4.1 Evolution of TREC QA.....                                 | 17        |
| 2.4.2 Foundation of QA Architectures .....                      | 20        |
| 2.5 Full-Text Retrieval and Question Answering in Chinese ..... | 22        |
| 2.5.1 Challenges with the Tasks in the Domain of Chinese.....   | 22        |
| 2.5.2 Related Work on Chinese Full-Text Retrieval .....         | 28        |
| 2.5.3 Related Work on Chinese Question Answering.....           | 29        |
| 2.5.4 Need for New Experiments with MultiText.....              | 31        |
| <b>Chapter 3 Concepts and Methods</b> .....                     | <b>34</b> |

|                                                               |           |
|---------------------------------------------------------------|-----------|
| 3.1 The MultiText Retrieval System.....                       | 35        |
| 3.2 Ranking Algorithms.....                                   | 35        |
| 3.3 Chinese Segmentation.....                                 | 44        |
| 3.4 QA in Chinese with MultiText.....                         | 46        |
| 3.4.1 The QA System Architecture .....                        | 46        |
| 3.4.2 Question Analysis.....                                  | 47        |
| 3.4.3 Passage Retrieval.....                                  | 50        |
| 3.4.4 Answer Extraction.....                                  | 50        |
| <b>Chapter 4 Experimental Setup .....</b>                     | <b>54</b> |
| 4.1 Document collections.....                                 | 55        |
| 4.2 Topics.....                                               | 57        |
| 4.3 Question Sets.....                                        | 59        |
| 4.4 Relevance Judgments.....                                  | 59        |
| 4.5 Evaluation Measures .....                                 | 60        |
| 4.5.1 Interpolated Recall-Precision Averages .....            | 60        |
| 4.5.2 Mean Average Precision (Non-Interpolated) .....         | 61        |
| 4.5.3 Average Precision at a Given Document Cutoff Value..... | 61        |
| 4.5.4 Average Cover at a Given Document Cutoff Value.....     | 61        |
| 4.5.5 Mean Reciprocal Rank (MRR).....                         | 62        |
| 4.5.6 Accuracy.....                                           | 62        |
| 4.6 System Setup.....                                         | 63        |
| 4.6.1 Re-Encoding and Indexing of the Chinese Texts .....     | 63        |

|                                                                      |            |
|----------------------------------------------------------------------|------------|
| 4.6.2 User Interfaces.....                                           | 63         |
| <b>Chapter 5 Experimental Results and Analysis.....</b>              | <b>72</b>  |
| 5.1 Experiments on Full-Text Retrieval.....                          | 72         |
| 5.1.1 Runs with Short Manual Queries .....                           | 73         |
| 5.1.2 Runs with Automatic Queries .....                              | 77         |
| 5.1.3 Runs with Pseudo-Relevance Feedback and Web Reinforcement..... | 81         |
| 5.1.4 Fusion of Best Runs.....                                       | 85         |
| 5.2 Experiments on Question Answering.....                           | 86         |
| 5.2.1 Impact of Different Segmentation Schemes.....                  | 87         |
| 5.2.2 Impact of Using Answer Patterns.....                           | 91         |
| 5.2.3 Comparison with the Marsha Chinese QA System .....             | 93         |
| 5.2.4 Evaluation with New Questions and the Web Corpus .....         | 94         |
| <b>Chapter 6 Conclusion and Future Work .....</b>                    | <b>97</b>  |
| <b>Bibliography.....</b>                                             | <b>101</b> |

# List of Figures

|                                                                                                          |    |
|----------------------------------------------------------------------------------------------------------|----|
| 2.1 A Sample of Recall-Precision Curve (MultiText TREC-7 Ad Hoc Result) .....                            | 12 |
| 2.2 Example of Various Chinese Translations for the Same Question in English ...                         | 26 |
| 3.1 Sample Unstructured Query Terms and Query Tiers (for CD Ranking) .....                               | 39 |
| 3.2 Sample Unstructured Query Terms and Query Tiers (for Tiered Ranking) .....                           | 41 |
| 3.3 The MultiText QA System Architecture .....                                                           | 47 |
| 4.1 A Document in the TREC Chinese Corpus .....                                                          | 55 |
| 4.2 Chinese Topic 28 from TREC-5 .....                                                                   | 58 |
| 4.3 TREC-6 Chinese Track User Interface .....                                                            | 65 |
| 4.4 Sample of a Chinese Text Retrieval Interface for QAP/CDR/Okapi BM25 .....                            | 67 |
| 4.5 Chinese Text Retrieval Interface for CD ranking .....                                                | 68 |
| 4.6 Chinese Text Retrieval Interface for Tiered ranking .....                                            | 69 |
| 4.7 Chinese Question Answering Interface .....                                                           | 71 |
| 5.1 Precision-Recall Curves for Short Manual Term Queries for TREC-5 and 6<br>Topics (Topics 1-54) ..... | 74 |

|                                                                             |    |
|-----------------------------------------------------------------------------|----|
| 5.2 Precision-Recall Curves for Short Bigram Term Queries for TREC-5 and 6  |    |
| Topics (Topics 1-54) .....                                                  | 75 |
| 5.3 Precision-Recall Curves for Automatic Queries (TD&LDC) for TREC-5 and 6 |    |
| Topics (Topics 1-54) .....                                                  | 78 |

# List of Tables

|                                                                                                                                              |    |
|----------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.1 The Evolution of New Techniques Used for the Ad Hoc Task .....                                                                           | 13 |
| 2.2 The Evolution of the TREC QA Track .....                                                                                                 | 19 |
| 2.3 Examples of Chinese Nouns Described with Numerals and Unit Words .....                                                                   | 27 |
| 3.1 General Rules for Determining Question Types with “What” Like Question<br>Words .....                                                    | 49 |
| 5.1 Results for Runs Based on Short Manual Term Queries for TREC-5 and 6 Topics<br>(Topics 1-54) .....                                       | 74 |
| 5.2 Results for Runs Based on Short Manual Bigram Queries for TREC-5 and 6<br>Topics (Topics 1-54) .....                                     | 75 |
| 5.3 Results for Runs Based on Automatic Query Set TD&LDC and 5 Ranking<br>Algorithms for TREC-5 and 6 Topics (Topics 1-54) .....             | 79 |
| 5.4 Results for Okapi BM25 Runs Based on 9 Automatic Query Sets for TREC-5 and<br>6 Topics (Topics 1-54) .....                               | 80 |
| 5.5 Best Short Manual and Automatic Runs with Local and Web Pseudo-Relevance<br>Feedback and Okapi BM25 (for TREC-5 and 6 Topics 1-54) ..... | 84 |

|                                                                               |    |
|-------------------------------------------------------------------------------|----|
| 5.6 Running Results of Basic QA System with 4 Segmentation Combinations ..... | 90 |
| 5.7 QA Results with Question Classification and Pattern Matching .....        | 91 |
| 5.8 Evaluation of Chinese QA with New Questions and Web Corpus .....          | 95 |



# Chapter 1

## Introduction

This thesis investigates passage retrieval and specific natural language processing (NLP) problems involved in the development of a Chinese question answering (QA) system with a re-visit of the traditional Chinese full-text retrieval task [46].

Automatic question answering is applied in the situations in which the users prefer to ask a question in the form of a natural language sentence rather than to formulate more complex queries, and would like the system to return the specific answer rather than to require the users to locate the answer by themselves from a list of documents [27]. QA systems are difficult to develop due to the complication of NLP. However, certain types of questions, for example, many factoid questions, are apt to be answered correctly with simple NLP on top of Information Retrieval (IR) techniques.

Typical existing QA systems rely on an information corpus and employ a pipeline architecture consisting of three main components: question analysis, information retrieval, and answer extraction. A question posed in natural language is processed by the question analysis component to formulate a query. The query is then resolved in the corpus by the information retrieval component to retrieve documents or snippets that are likely to contain the answers to the question. Finally the answer extraction component determines the answers from the retrieved information.

As IR is an important component of a QA system, modern IR technology has significantly encouraged the research on QA. As mentioned above, the goal of IR is to identify appropriate information<sup>1</sup> that meets the user's requirement. In more detail, the process can be modeled as searching literature in a given library. In an information retrieval system a "library" is represented as a set of searchable documents held in a text corpus. The system indexes the corpus and provides some searching mechanism to select documents that are likely to satisfy the user's requirement. Most modern IR systems have entire documents in the corpus accessible and provide full-text retrieval capabilities. A traditional full-text retrieval task, defined by the Text REtrieval Conference (TREC), is also called *ad hoc* retrieval, where the IR system is aware of the text collection containing articles with a large diversity of subjects, but the specific topics to be explored are arbitrarily defined and unknown to the system in advance [38]. An appropriate query has to be formulated to

---

<sup>1</sup> In context of this thesis our focus is on traditional natural language texts in machine-readable form, such as electronic text documents stored on disks or accessible on the Web.

represent each topic retrieved from the text collection, and a ranked list of documents in decreasing order of proximity to the query is to be returned as the result.

Existing techniques developed for QA and IR are mainly applied to English texts only. In particular, the MultiText research group at the University of Waterloo has integrated a set of passage-based techniques into their QA and IR systems to rank full documents or small passages within an English text collection. Passage-based retrieval techniques are featured in that full documents are split into passages and only documents or fragments containing relevant passages are retrieved. During the University of Waterloo's participation in various tracks at TREC in recent years, the use of those passage-based techniques in MultiText systems has been shown effective for QA and IR in English.

However, in the domain of languages other than English, for example, Chinese, the effectiveness of those techniques has been seldom explored. Chinese differs from English mainly in that texts are written as consecutive characters without explicit word boundaries. For Chinese text processing, specific techniques are required to deal with Chinese language characteristics, such as word segmentation, sentence structure analysis, and the recognition of named entities in the form of non-regular words, such as Chinese person names and numbers.

As there was no previous work that investigated migrating the MultiText techniques from English to Chinese QA, the motivation of this thesis was to address the problems in developing a QA system in Chinese. In the course of this

investigation we found it necessary to re-visit Chinese full-text retrieval, which was explored during the MultiText's participation in the TREC-6 Chinese track. For one reason, IR plays an important role in QA. System set up specifications used in previous Chinese experiments would be very helpful in building our QA system. For another reason, the MultiText's Chinese experiments at TREC-6 only focused on full-text retrieval with manually constructed queries. Experiments with system-formulated queries were not conducted. Besides, the new passage retrieval algorithm and answer extraction heuristics developed specifically for QA could be adapted for full-text retrieval purposes, including the ranking of full documents and pseudo-relevance feedback.

This thesis therefore addresses both Chinese full-text retrieval and the development of a Chinese QA system. The content is organized as follows: Chapter 2 gives background information on document and passage-based retrieval strategies, TREC and its relations to full-text retrieval and QA research, as well as the challenges in conducting those tasks in the Chinese environment. Chapter 3 details the MultiText's retrieval algorithms, the various Chinese segmentation schemes, and many special considerations in building a Chinese QA system. Chapter 4 describes the experimental set up specifications. Chapter 5 reports the experimental results to evaluate the effectiveness of MultiText's retrieval strategies on both full-text retrieval and QA in Chinese. The impact of segmentation and pseudo-relevance feedback on retrieval performance is also investigated. Chapter 6 concludes this thesis and suggests future work.

# Chapter 2

## Background

### 2.1 Document-Based Retrieval vs. Passage-Based Retrieval

Text retrieval strategies are typically designed to identify documents relevant to a user's query in the text collection. In many statistically based retrieval systems, techniques using whole-document similarity measures have been well developed to rank documents according to their estimated likelihood of relevance. These techniques rank full documents by measuring the degree of similarity of a document to the query according to a heuristic similarity function.

The choice of a similarity function is crucial for ranking effectiveness. There have been many functions proposed. A proven effective formulation is the *cosine measure* [41,54,71], which is defined as:

$$C(q, d) = \frac{\sum_{t \in q \wedge d} (w_{q,t} \cdot w_{d,t})}{W_d}$$

where  $q$  is a query,  $d$  is a document,

$$W_d = \sqrt{\sum_{t \in d} w_{d,t}^2},$$

$$w_{d,t} = \log(f_{d,t} + 1),$$

$$w_{q,t} = \log(f_{q,t} + 1) \cdot \log(N / f_t + 1),$$

where the value  $f_{d,t}$  is the frequency of term  $t$  in  $d$  (TF),  $f_{q,t}$  is the frequency of  $t$  in  $q$ ,  $N$  is the total number of documents in the collection,  $f_t$  is the number of distinct documents that contain term  $t$ , and  $\log(N / f_t + 1)$  is known as the Inverse Document Frequency (IDF), a measure that estimates the rarity of term  $t$  in the collection. A document ranked by the cosine measure attracts high scores if it contains many of the query terms, and if those terms are common within the document but relatively rare in the collection. As longer documents tend to cover more terms, document length normalization is used to avoid favoring retrieval of longer documents.

Traditional text retrieval systems were used to search relatively short documents, such as abstracts of papers. Modern systems are facing the challenge of providing searching capabilities for a variety of full documents, which can be as arbitrarily long as several megabytes. Since the heuristic whole-document similarity measures disregard the location or proximity of the query terms within a document, the terms

occurring together in the same document are not necessarily close to each other, especially for longer documents. This kind of documents, although ranked high by the similarity measures, is often non-relevant to the query. In contrast, it has been observed that a document with a relatively short fragment—called a *passage*—containing a high density of query terms is more likely to be relevant than a document matching the same number of query terms that are located far away from each other. For example, for a query “Computer Science University of Waterloo”, a document with a short passage matching all these terms is very likely to be relevant; whereas a document without such a passage but instead talking about “Computer Science” and “University of Waterloo” in distantly separated blocks is less likely to satisfy the query. Inspired by this observation, researchers have developed alternative retrieval approaches—called *passage-based strategies*—to divide each document into a set of passages and compute the similarity between each passage and the query [20,43,52,55,57,61]. In these approaches the results returned to the user can be the highest-weighted passages, or a ranked list of documents, where the score assigned to each document is determined either according to its best passage [58], or by combining the weights of best passages it contains [57].

In spite of the potential advantage that passage-based retrieval techniques may improve retrieval effectiveness, ranking of passages can considerably increase computational costs, as a larger number of candidate text units have to be ranked. Therefore a practical passage-based strategy should appropriately define the type of passages to be ranked. In the past decade experiments have been conducted over the

following types: marked up *sections* [13,42-44,47,55,61,70], roughly equal-length *pages* or *paragraphs* [55], semantic units delimited according to topic shifts [26,57], fixed-length of words [52,57,58], and *arbitrary passages* where a passage is allowed to start at any point in the document with any length [58]. The results obtained by Kaszkiel and Zobel [58] show that in their experiments, the most effective type of passages are fixed-length passages of 150 to 300 words.

In the context of question answering, passage-based strategies have notable advantages over whole-document retrieval techniques, as the answer to a query (or question) is very likely to be covered by a small portion of text that satisfies the query with high density of query terms. Recently Tellex *et al.* [69] conducted quantitative evaluation over a set of passage retrieval algorithms used for existing QA systems, and they concluded that density-based measures of query terms significantly affect the passage ranking and hence the overall performance of a QA system.

## 2.2 The Text REtrieval Conference (TREC)

To review the evolution of research on full-text retrieval and question answering, one cannot ignore the significance of the Text REtrieval Conference (TREC), the overall goal of which is to foster research in information retrieval using large-scale test collections, and to encourage interaction among research groups from industry, academia and government in an open forum. TREC is co-sponsored by the National

Institute of Standards and Technology (NIST) and the Defense Advanced Research Projects Agency (DARPA). An important way TREC attracts participants and research, is that every year TREC defines a set of research tasks—called *tracks*—associated with standard test collections. All participants who take part in one or more of the tracks are required to run experiments on their own information systems and to submit results in specific formats to NIST. This structure allows different techniques to be compared based on the same evaluation standard, and thus participants may have the opportunity to understand the challenges with respect to each track, and to exchange research ideas on how to choose and improve their techniques. During the past 12 years, many tracks have been investigated, including ad hoc, routing, Web, QA, and tracks with multiple languages or multi-media.

For all tracks, the quality of the test collections is crucial to the success of TREC. A typical TREC test collection is similar to most traditional retrieval collections in that it consists of three main parts: The documents, the topics, and the relevance judgments [18]. The documents corpus should be large and should reflect a diversity of document length, subjects, vocabulary, and writing styles. To simulate a real user's information requests, the topics should facilitate constructing queries with a variety of methods, either manual or automatic, and should give clear criteria that make a document relevant. For each given topic, ideally relevant judgments need to be made upon all documents in the collection. This is impractical as it would result in a tremendous amount of judgment work. A successful approximation known as *pooling* [56] used at TREC is to judge relevance on the sample of documents selected by the

various participating systems. Details of the pooling method in our particular experiments are given in Chapter 4. Participants' runs against the test collection are evaluated using several measures. Traditional test collections are made from the full text of documents, and evaluation measures are based on *recall* and *precision* [48]. Recall is defined as the proportion of documents relevant to a search query that are retrieved by a given search formulation, whereas precision is defined as the proportion of documents retrieved by a given search formulation that are relevant. As TREC has expanded into many different tasks, new types of document sets as well as new ways of relevance judgments and evaluations have been devised. The details of test collections and evaluation measures for *ad hoc* retrieval and QA are described in the following sections.

### 2.3 Full-Text Retrieval at TREC

The full-text retrieval, or the *ad hoc* retrieval in context of this thesis, was started as a main task from TREC-1 [17]. The purpose is to investigate the retrieval performance of systems that search a fixed document collection with new topics. Participants are given a set of new topics to formulate queries and to retrieve a ranked list of 1000 documents for each topic from a given document collection, assumed to be in decreasing order of likelihood of relevance to the topic. The relevance judgments for those topics are not unknown by the participants in advance.

### 2.3.1 The Test Collection

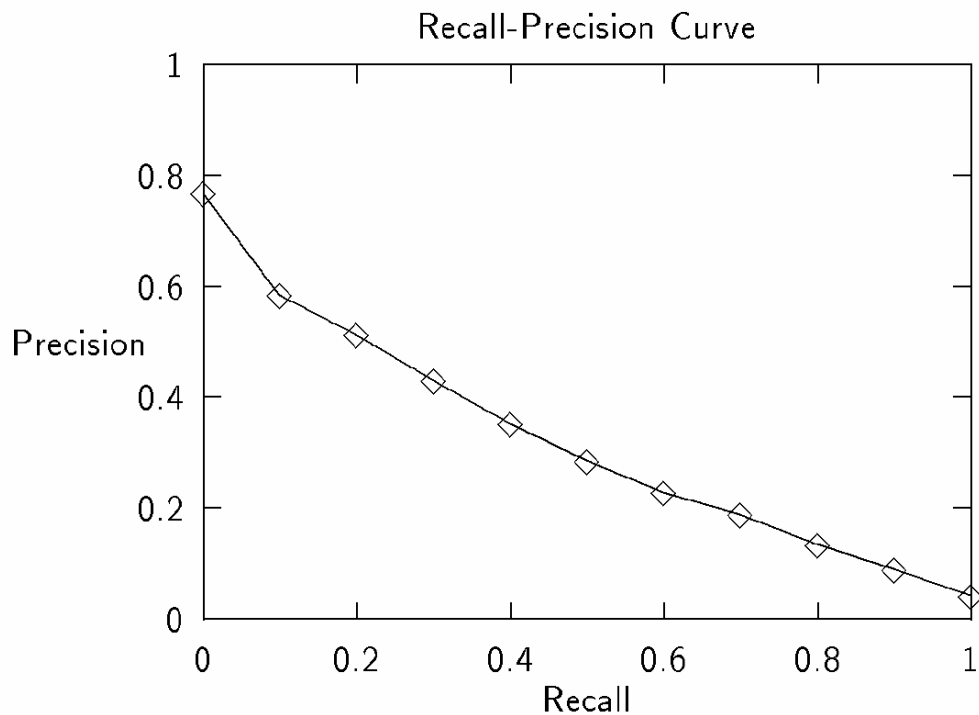
At TREC, the ad hoc test collection provided in English contains about 2 gigabytes of documents and 50 topics. The high level structure of each document is marked up with SGML tags to identify fields of texts such as the document boundaries, document identification numbers, headlines, and text bodies. Each topic is composed by a topic number, a “title” field with one to three keywords, a “description” field with one sentence description of the topic area, and a “narrative” field that gives the criteria of what makes a document relevant. Participants can construct queries in any matter they like, but have to distinguish manual queries from purely automatic queries. The former kind of methods allows manual intervention, either with or without machine assistance; whereas the latter extracts information automatically from the topics to formulate queries, and any query refinement must be automatic process as well. For each set of automatic queries, participants should also report which topic fields are used for query construction when submitting their runs.

### 2.3.2 The Evaluation Measures

The ad hoc runs are evaluated by a common scheme—the trec-eval package implemented by Chris Buckley [4]. This package contains several measures derived from recall and precision. The measures that have been mostly used to compare runs are the *recall-precision curve* and the *mean average precision* [37]. The recall-precision curve, as shown by the MultiText TREC-7 ad hoc result as an example in Figure 2.1, plots average precision over all test topics at each given recall level, and

reflects the retrieval behavior of a particular run over the entire spectrum of recall.

The roughly inverse relationship between recall and precision in the curve is because when more documents are retrieved, the absolute number of relevant documents usually increases, while the proportion of documents fetched that is relevant is likely to decrease. For each topic, the average precision score (non-interpolated) is the mean of the precision at the level that each relevant document is retrieved, and the mean of such scores over all topics is the mean average precision for a whole run. A more precise description of the measures used in our experiments and other evaluation schemes are given in Chapter 4.



**Figure 2.1: A Sample of Recall-Precision Curve (MultiText TREC-7 Ad Hoc Result)**

### 2.3.3 The Research Areas

For the ad hoc task, six major research areas have been involved: the ranking techniques, the use of passages, the use of top documents with or without other schemes for query expansion, the user-in-the-loop experiments for manual expansion, the combination of different runs using “data fusion”, and the query formulation methods based on particular topic fields. Table 2.1, quoted from Voorhees and Harman [35], shows the history of those now-widely-accepted techniques.

| TREC-2                                                                | TREC-3                                            | TREC-4                                                           | TREC-5                                                    | TREC-6                                                  |
|-----------------------------------------------------------------------|---------------------------------------------------|------------------------------------------------------------------|-----------------------------------------------------------|---------------------------------------------------------|
| baseline for most systems<br>beginning of Okapi weighting experiments | Okapi perfects BM25 algorithm                     | new SMART weighting algorithm<br>new INQUERY weighting algorithm | use of Okapi / SMART weighting algorithms by other groups | adaptations of Okapi / SMART algorithms in most systems |
| use of subdocuments by PIRCS system                                   | heavy use of passages / subdocuments              |                                                                  |                                                           | use of passages in relevance feedback                   |
|                                                                       | beginning of expansion using top X documents      | heavy use of expansion using top X documents                     | beginning of more complex expansion schemes               | more sophisticated expansion experiments by many groups |
|                                                                       | beginning of manual expansion using other sources | major experiments in manual editing / user-in-the-loop           | continued user-in-the-loop experiments                    | extensive user-in-the-loop experiments                  |
|                                                                       | initial use of “data fusion”                      | continued use of “data fusion”                                   | continued use of “data fusion”                            | more complex use of “data fusion”                       |
|                                                                       |                                                   |                                                                  | beginning of more concentration on initial topic          | continued focus on initial topic, including title       |

**Table 2.1: The Evolution of New Techniques Used for the Ad Hoc Task**

Table 2.1 only gives the history of the ad hoc track from TREC-2 to TREC-6. In fact, TREC-1 was the first time that IR research groups had ever produced their runs on the same data set and compared results using the same evaluation scheme. The

huge increase in the size of the text collection forced most participating groups to put their major efforts on scaling up their systems' retrieval capabilities [19]. Since TREC-2, there had been significant improvements on many systems' performance, as more and more new techniques were developed. By TREC-6, some techniques had been widely spread and become standard usage. Many systems' TREC-7 and TREC-8 runs were produced using the same basic processing as in TREC-6. Ad hoc was discontinued from TREC-9, as people believed that enough infrastructures already existed. It was not until TREC-12 that ad hoc was brought back, renamed *robust*, for which the focus was on investigating poorly performing topics and improve the consistency of retrieval technology [34].

Among the six research areas involved in the ad hoc task, the use of high-quality ranking techniques is of primary importance. The most widely used technique is a probabilistic term weighting algorithm called *Okapi*, originally developed by Robertson *et al.* from City University, London [65,66]. The main feature of Okapi is that it accounts for document term weight, query term weight, and document length. The initial versions of Okapi functions were refined and combined into BM 25 for TREC-3, and afterwards it was either implemented by other TREC participants, or combined with other weighting schemes and adapted into other systems, such as INQUERY from the University of Massachusetts [49] and SMART from Cornell [5]. The MultiText group at the University of Waterloo had produced runs for ad hoc task since their first participation in TREC-4, and had used Okapi in conjunction with various passage-based algorithms. The overall performance had proved very good.

The passage-based retrieval and Okapi BM25 formulae used in MultiText implementations are explained in the next chapter.

As for other research areas, the second row in Table 2.1 shows that passages were heavily used in TREC-2 and 3, but less widely used in TREC-4 and 5 as many participants were concentrating on improving their term weighting algorithms. There were more use of passages again in TREC-6, but mostly for query expansion with relevance feedback, which is shown in more detail in the third and fourth lines of the table, standing for automatic feedback and manual feedback respectively. The general scheme for feedback is to select words appearing in many relevant documents but in relatively few irrelevant documents, because they are likely to be related to the user's information request and helpful to retrieve more relevant documents. As retrieval runs are produced without relevance judgment information, it is assumed that the top ranked documents are relevant. Automatic feedback using the top retrieved documents is also known as pseudo-relevance feedback, which had been used in various ways by most participating groups by TREC-6.

Another widely used technique, data fusion, refers to combining runs produced from different techniques. An observation is that different techniques may be suitable in different situations. For example, some weighting schemes perform better at low recall levels whereas others work better at high recall levels. Merging results from multiple runs can be helpful to compensate for the shortfalls of different techniques and improve the overall retrieval performance.

The last row in Table 2.1 regards query formulation. Queries in early TRECs were considered as bags of equally weighted words extracted from topics. As the ad hoc task went on, the query formulation process became more complex. For example, Waterloo's MultiText group used GCL [12], a query language developed within the group to generate queries with Boolean expressions, ordering, tiering and other features [46]. Another issue with respect to query formulation is the investigation of different topic fields used for creating automatic queries, such as title only (Very Short version), description only (Short version), and all fields (Full version). In the case of English ad hoc, the Very Short version surprisingly works as well as Full version, while the Short version is the worst. However, given this result, it is difficult to tell the length of queries that is generally most suitable for retrieval purpose. For one reason, the appropriate length varies from topic to topic. Some "bad" topics inherently have very few relevant documents in the corpus, while the "good" topics have a considerable number of relevant documents. Adding more terms into queries simply makes good topics perform better, whereas bad topics even worse. A second reason is that different retrieval techniques may be suitable for different lengths of queries. Some groups therefore applied different schemes to different query versions [6,47], and accordingly the runs were not easily comparable with regards to query lengths only.

## **2.4 Automatic Question Answering at TREC**

### **2.4.1 Evolution of TREC QA**

The main goal of question answering (QA), as mentioned in the previous chapter, is to have the system return actual answers in response to a question, rather than a ranked list of documents as for traditional ad hoc tasks. The assumption is that users may prefer to have their questions answered in a short snippet of text (for example, an answer “Abraham Lincoln” to the question “Who is the 16th President of the United States?”) rather than look for the answer in full documents. Research on QA can date back to the 1960’s, with a long history in complex natural language processing (NLP) [40]. In recent years, the research focus is on extracting answers from large text collections with various strategies built on top of modern IR technology.

The QA track has run since TREC-8 in 1999, and during its five years’ evolution, both the scope and difficulty have been substantially expanded. Table 2.2 gives a comprehensive comparison among the QA tracks over the past five years [27,28,30-33], based on the criteria of defined tasks, the number of participating groups and submitted runs, test collection specifications, requirement of submissions, judgment decisions, evaluation measures, and best results for the main tasks.

|                                                              | <b>TREC-8</b>                                                                                                           | <b>TREC-9</b>                                                                                         | <b>TREC-10</b>                                                                                                                                 | <b>TREC-11</b>                                                                                                                                                                         | <b>TREC-12</b>                                                                                                                                                                                                                                                                                                                                             |
|--------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Tasks</b>                                                 | 50-Byte /<br>250-Byte                                                                                                   | Same as<br>TREC-8                                                                                     | Main/List<br>/Context                                                                                                                          | Main/List                                                                                                                                                                              | Main/<br>Passage                                                                                                                                                                                                                                                                                                                                           |
| <b>No. of<br/>Participants</b>                               | 20                                                                                                                      | 28                                                                                                    | 36                                                                                                                                             | 34                                                                                                                                                                                     | 33                                                                                                                                                                                                                                                                                                                                                         |
| <b>No. of Runs</b>                                           | <u>50-Byte</u> : 20<br><u>250-Byte</u> : 25                                                                             | <u>50-Byte</u> : 34<br><u>250-Byte</u> : 44                                                           | <u>Main</u> : 67<br><u>List</u> : 18<br><u>Context</u> : 7                                                                                     | <u>Main</u> : 67<br><u>List</u> : 9                                                                                                                                                    | <u>Main</u> : 54<br><u>Passage</u> : 21                                                                                                                                                                                                                                                                                                                    |
| <b>Document<br/>Collection<br/>(newspaper/<br/>newswire)</b> | TREC-8 Ad<br>Hoc<br>Collection<br>(TREC disks<br>4, 5; 528,000<br>documents;<br>2GB)                                    | TREC disks<br>1-5; 979,000<br>documents<br>3GB                                                        | Same as<br>TREC-9                                                                                                                              | AQUAINT<br>Corpus of<br>English<br>News;<br>1,033,000<br>documents<br>3GB                                                                                                              | Same as<br>TREC-11                                                                                                                                                                                                                                                                                                                                         |
| <b>No. of<br/>Questions</b>                                  | 198 factoids,<br>guaranteed<br>to have<br>answers in<br>corpus<br>(released:<br>200)                                    | 682 more like<br>“real”<br>questions<br>guaranteed to<br>have answers<br>in corpus<br>(released: 693) | <u>Main</u> : 500<br><u>List</u> : 25<br><u>Context</u> : 42,<br>(grouped in<br>10 series)<br>(questions not<br>guaranteed to<br>have answers) | <u>Main</u> : 500<br><u>List</u> : 25<br>(not<br>guaranteed<br>to have<br>answers)                                                                                                     | <u>Main</u> :<br><u>Factoid</u> : 413<br><u>List</u> : 37<br><u>Definition</u> :50<br><u>Passage</u> : 413<br>(same as main<br>factoids)<br>(not guaranteed<br>to have answers)                                                                                                                                                                            |
| <b>Question<br/>Source</b>                                   | FAQ Finder<br>Log,<br>Assessors,<br>Participants                                                                        | Encarta log,<br>Excite log,                                                                           | MSNS logs,<br>AskJeeve<br>logs                                                                                                                 | MSNS logs,<br>AskJeeve<br>logs                                                                                                                                                         | AOL and<br>MSN search<br>Logs                                                                                                                                                                                                                                                                                                                              |
| <b>Answer<br/>Formats</b>                                    | A ranked<br>list of 5<br>[document-id,<br>answer-<br>string]<br>pairs per<br>question,<br>limited<br>to 50/250<br>bytes | Same as<br>TREC-8                                                                                     | <u>Main/context</u> :<br>Same as<br>TREC-8<br><br><u>List</u> : an<br>Unordered<br>List of<br>[document-<br>id, answer-<br>string] pairs       | <u>Main</u> :<br>only one<br>[document-<br>id, answer]<br>pair, exact<br>answer or<br>“NIL”<br><br><u>List</u> : same as<br>TREC-10<br>list, but<br>required to<br>be exact<br>answers | <u>Main</u> :<br><u>Factoid</u> : same as<br>TREC-10 Main<br><u>List</u> : same as<br>TREC-10 list,<br>But no target<br>Number<br><u>Definition</u> : same<br>As list, but no<br>Limit to answer<br>String lengths<br><br><u>Passage</u> : one<br>response to<br>each question,<br>within-document<br>offset and span<br>length (250-byte<br>limit) marked |
| <b>Correctness<br/>Judgments</b>                             | “correct” if<br>string<br>contains right<br>answers;<br>unsupported<br>strings<br>are correct                           | Correct/<br>Incorrect/<br>Unsupported<br><br>Lenient<br>scores:<br>unsupported=                       | <u>Main/Context</u> :<br>Same as<br>TREC-9<br><br><u>List</u> :<br>Correctness/<br>Distinctness                                                | <u>Main</u> :<br>Incorrect/<br>Unsupported/<br>Inexact/<br>Correct<br><br><u>List</u> : same as                                                                                        | <u>Main</u> :<br><u>Factoid/list</u> :<br>Incorrect/<br>Unsupported/<br>Inexact/<br>Correct                                                                                                                                                                                                                                                                |

|                                       |                                                                |                                                                               |                                                                                  |                                                                                                                                                       |                                                                                                                                                                      |
|---------------------------------------|----------------------------------------------------------------|-------------------------------------------------------------------------------|----------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                       |                                                                | correct<br><br>Strict (official)<br>)<br>scores:<br>Unsupported=<br>incorrect |                                                                                  | TREC-10 list                                                                                                                                          | <u>Definition:</u><br>“Information<br>nuggets” <sup>2</sup><br>created<br>and marked by<br>assessors<br><br><u>Passage:</u><br>Incorrect/<br>Unsupported/<br>Correct |
| <b>Evaluation<br/>Measures</b>        | MRR <sup>3</sup>                                               | MRR                                                                           | <u>Main/Context:</u><br>MRR<br><br><u>List:</u> Average<br>Accuracy <sup>4</sup> | <u>Main:</u> No. of<br>Correct;<br>Confidence<br>Weighted<br>Score <sup>5</sup> ; NIL<br>Accuracy <sup>6</sup><br><br><u>List:</u> same<br>as TREC-10 | <u>Main:</u><br>FinalScore=<br>1/2*FactoidScore<br>+1/4*ListScore <sup>7</sup><br>+1/4*DefScore <sup>7</sup><br><br><u>Passage:</u><br>Accuracy                      |
| <b>Best Main<br/>Task<br/>Results</b> | <u>50-Byte:</u><br>MRR: 0.66<br><u>250-Byte:</u><br>MRR: 0.646 | <u>50-Byte:</u><br>MRR: 0.58<br><u>250-Byte:</u><br>MRR: 0.76                 | MRR: 0.68                                                                        | # correct: 415<br>Confidence<br>weighted<br>score: 0.856                                                                                              | Final: 0.559<br>(Factoid: 0.7,<br>list: 0.396,<br>Def: 0.442)                                                                                                        |

Table 1.2: The Evolution of the TREC QA Track

---

<sup>2</sup> “Information nugget”: A fact that an assessor can make a binary decision if a response contains the nugget [33]

<sup>3</sup> MRR: Mean Reciprocal Rank: the score of each question is the reciprocal of the rank where the first correct answer is found, if there is any; otherwise its score is 0. The mean of all questions reciprocal ranks is assigned as the score for a run. [36]

<sup>4</sup> Average Accuracy [31]

<sup>5</sup> Confidence Weighted Score:  $F = \frac{1}{Q} \sum_{i=1}^Q \frac{\text{number correct in first } i \text{ ranks}}{i}$  [32]

<sup>6</sup> NIL Accuracy [32]

<sup>7</sup> FactoidScore: Accuracy: percentage of correct answers. [33]

ListScore: Equally weighted Instance Recall (IR) and Instance Precision (IP), or  $F=2*IP*IR/(IP+IR)$  [33]

DefScore: A F score based on Nugget Recall and Nugget Precision [33]

### 2.4.2 Foundation of QA Architectures

In the early years of TREC QA, the answers required to be returned were five 50 or 250-byte document extracts per question. Although this allowed systems to locate answers with simple bag-of-words approaches, especially for the 250-byte tasks, participants were compelled to introduce more or less natural language techniques. The typical general architecture of a QA system was already set up, which consisted of three main components: question analysis, search, and answer extraction. The question analysis was mainly used to identify the category of the question and key terms so as to formulate a searchable query. The searching component then retrieved relevant documents or passages, which were finally parsed by the answer extraction component to determine the most possible answers as results.

Later on as the task requirements became more and more sophisticated, including returning a single exact short answer rather than five document extracts to a factoid question, and the special answer formats for list and definition questions, systems became increasingly complex. Although most systems still followed the general pipeline architecture, they had little in common at more detailed levels. In some systems there were successive feedback loops within or across the basic pipeline components [22,64]. Some other systems employed a parallel architecture in which multiple QA agents answered the same question independently and voted the final answer with knowledge-based justification [50].

For question classification, there was a wide range of ways to define the ontology of question types, from very broad or highly specified. The techniques used for understanding the incoming questions could be keyword finding, pattern matching or natural language parsing. To formulate appropriate queries, an online lexicon WordNet [7] was widely used to expand the initial query with related words as well as verify answer types in the later answer extraction component.

In the search component, the two major different approaches were full-document retrieval [2,25,60] and passage retrieval [15,53]. As mentioned before, a passage covering a high density of query words in a document has much greater probability to include the actual answers than other parts of the document; therefore to have the system directly return passages has more advantages over retrieving full documents. In fact, systems using full-document retrieval usually still require a second sentence selection phase to essentially simulate passage retrieval; accordingly, compared to full-document retrieval, passage retrieval is also more efficient as it can reduce the amount of information to be processed for answer extraction.

In the answer extraction component, many systems extracted named entities corresponding to a question's category as answer candidates [63], while others viewed all retrieved short snippets matching predefined simple patterns as candidates [15]. Systems that attempted to fully understand questions also tended to apply sophisticated natural language processing to relate answers to questions, such as recognizing syntactic alternations, resolving anaphora, and abductive proofs [22]. However, many other systems avoided understanding the structure and meaning of

language by using data-driven approaches. Data-driven methods are based on the observation that the massive amounts of data in very large corpora (e.g., the Web) are likely to produce repeated occurrences of the same answer across different documents. Such data redundancy could provide simple justification for proposed answers and facilitate a voting scheme to determine the best ones according to the frequency of candidate answers in the retrieved passages [8,15,24].

The MultiText group at the University of Waterloo has participated in the main or passage subtasks of QA track since TREC-8, and the system has achieved top-six or better performances [10,14,16,21,45]. The system features in arbitrary passage retrieval and answer accuracy validation with term redundancy. More details of the heuristics are given in the next chapter.

## **2.5 Full-Text Retrieval and Question Answering in Chinese**

### **2.5.1 Challenges with the Tasks in the Domain of Chinese**

In the Chinese language, texts are written as a linear sequence of consecutive ideographic characters. A character is neither like a word nor a letter in English. It represents a complete syllable and may have a set of basic meanings associated with it. However, the actual independent linguistic units in the Chinese texts are not characters, but Chinese words, most of which consist of more than one character in a specific order and the length of a word varies.

As the word boundaries are not given as explicitly as in English-like languages, automatic segmentation is one of the main challenges in processing Chinese texts. There has been much research in successful Chinese text segmentation [1,68,74].

One scheme proposed by Chen *et al.* [1] is based on simple statistics without the use of a Chinese dictionary. Let  $p(c_1, \dots, c_n)$  be the probability of a Chinese string  $c_1 \dots c_n$  occurring in the collection, which can be obtained by:

$$p(c_1, \dots, c_n) = \frac{f(c_1, \dots, c_n)}{N}$$

where  $f(c_1, \dots, c_n)$  is the number of occurrences of this string, and  $N$  is the total number of characters in the collection. The *mutual information*  $I(c_1, c_2)$  between character  $c_1$  and the next one  $c_2$ , defined by Sproat and Shih [68], is formulated as:

$$I(c_1, c_2) = \log \frac{p(c_1, c_2)}{p(c_1) \times p(c_2)} = \log \frac{f(c_1, c_2) \times N}{f(c_1) \times f(c_2)}$$

If  $I(c_1, c_2)$  is high, the bigram composed by  $c_1$  and  $c_2$  may be a word. To segment a whole text, Chen *et al.* suggested to first determine the character frequencies from the text and presume each bigram with a mutual information value above a threshold as a word, then parse the sentences repeatedly by delimiting the words.

Since the threshold value for mutual information is fixed, even if it is carefully chosen (Chen *et al.* used 7), this statistical method is likely to fail to find uncommon words while finding some non-real words. In contrast, another type of approach, segmentation with a Chinese dictionary [74], usually produces more accurate words.

These approaches can be classified into three groups: the longest match, the shortest match, and the overlap match [72]. The longest match is also called *greedy parsing*, in which the text is scanned sequentially and matched against the dictionary for the longest entry. In the shortest match, the text is sequentially scanned and the first strings found to match the dictionary are taken as words. Compared to the longest match, shortest match usually generates more words with less specific meaning. In both of these methods, when a word entry is found, the word boundary is marked and the match process starts from the next character. The overlap match differs from them in that words generated are allowed to overlap with each other in the text.

None of the above schemes can guarantee segmenting Chinese text fully correctly. Due to the ambiguous nature of many characters' meaning, accurate segmentation can be so difficult that even humans might disagree on segmenting the same piece of text. Fortunately, despite the difficulties in segmentation, there can be different requirements on segmentation for different applications, depending on the need to understand the meaning of the text. For example, natural language processing and machine translation may require more accurate segmentation, whereas indexing and query formulation in text retrieval can be either based on advanced segmentation of the text, or on single or fixed-size blocks of characters. In fact, text retrieval with well-segmented words does not necessarily result in better performance. In indexing, if the indexed words are produced with the longest match algorithm, a query containing a short word may be a partial match to the text. In query formulation, long

query words may be too specific and are likely to miss many relevant documents, while the conjunction or disjunction of short words may result in many irrelevant documents being retrieved.

In contrast, indexing with the character-based approaches, such as unigrams, bigrams or trigrams, can be more flexible, but can also have problems. Bigrams and trigrams are more likely to carry specific meaning than unigrams; however, an index with bigrams or trigrams may be too large to be manageable. For instance, a corpus containing only 1000 distinct Chinese characters can result in 1 million bigrams. As for query formulation, bag-of-unigrams are too ambiguous to represent the user's information need; bigrams and trigrams may have the same problem as using well-segmented words, however, overlapped bigrams (or trigrams) can either stand for distinct short words or reformulate longer words, and therefore are probably more suitable for the text retrieval purpose.

In the context of question answering in Chinese, since more natural language processing is involved, there may be a need for correct segmentation. Besides, there are other challenges with respect to the Chinese features. Special processing techniques are necessary to overcome the difficulties.

Firstly, most questions in English start with a “*wh-*” word or phrase, but Chinese questions have no such “standard” format. For example, a question in English “**When**

did Hong Kong return to China?” can be translated in various forms in Chinese, as shown in Figure 2.2:

|                                                                                                                                                                                                                                          |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>A question in English:<br/><b>When</b> did Hong Kong return to China?</p> <p>Chinese translations:<br/>香港<b>什么时候</b>回归的中国?<br/>香港回归中国是<b>什么时候</b>?<br/>香港<b>何时</b>回归的中国?<br/>香港在<b>哪一年</b>回归了中国?<br/><b>哪一年</b>香港回归了中国?<br/>.....</p> |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**Figure 2.2: Example of Various Chinese Translations for the Same Question in English**

Five of the possible translations are listed. In each Chinese sentence, the bold characters stand for the same meaning as the English question word “when”. It can be noticed from Figure 2.2 that a question word may be written in various ways and may be hidden in the start, or the middle, or the end of a Chinese question. This results in more complexity in question analysis, as there is no general template that can be used to classify the question type.

Secondly, Chinese characters are all case insensitive. In other words, there are no upper or lower-case specifications. Many named entities, such as person names, organizations and locations, are easy to identify in English texts because they are

written as words starting with capital letters. In Chinese, however, there are no explicit signals to indicate such entities. They are written as normal sequences of Chinese characters and hidden in the non-segmented text. Since they are relatively rare in a text collection, and many of them (e.g. person names) are not even predefined words, in many cases neither statistics nor dictionary-based segmentation schemes can recognize them correctly.

Similar to the named entities problem, numerals are also difficult to identify. Each Chinese number is a combination of one or more numeral characters, usually followed by a unit or measure word. A Chinese segmenter does not view a number as a regular word. Moreover, most Chinese nouns require specific unit words to describe them, even though the unit words are not used for measurement. Table 2.3 gives several examples:

| Chinese       | English                   |
|---------------|---------------------------|
| 一个女孩          | <b>a</b> girl             |
| 五百棵 <u>树</u>  | <b>five hundred</b> trees |
| 十五辆 <u>车</u>  | <b>fifteen</b> cars       |
| 二十二匹 <u>马</u> | <b>twenty-two</b> horses  |

**Table 2.3: Examples of Chinese nouns described with numerals and unit words**

In each Chinese phrase example, the underscored character is the unit word, and the text before it is the numeral corresponding to the bold text in the English translation. The rest of the text is the noun. One can notice that the use of those unit words is a

special characteristic in Chinese, and different unit words are designated to serve different nouns.

### **2.5.2 Related Work on Chinese Full-Text Retrieval**

Multilingual ad hoc tracks started at TREC-3, and Chinese was introduced in TREC-5 and 6 [3,62]. The Chinese tracks in both TREC-5 and 6 were supplied with the same text corpus—a collection with 164,811 Chinese newspaper/newswire articles with no segmentation information. The GB<sup>8</sup> encoded raw text is about 170 megabytes. The definition of the Chinese track was the same as the standard ad hoc retrieval: to search the given collection with new topics and submit a ranked list of 1000 documents for each topic. In TREC-5, participants were given 28 topics, and in TREC-6, 26 new topics were used. 10 groups took part in the TREC-5 Chinese track, and 12 groups were involved in TREC-6.

The MultiText group participated in the Chinese track only at TREC-6 [46]. Each Chinese character appearing in the text collection was re-encoded into a 6-byte “word” as described in Section 4.6.1 of Chapter 4, and was then indexed individually. A Chinese word composed by a sequence of adjacent characters was therefore comparable to an English phrase and could be searched with the phrase searching capability provided by the MultiText retrieval system.

---

<sup>8</sup> A common encoding standard in China and Singapore.

MultiText only submitted a run with manual queries. For each topic, the query was manually formulated with the GCL query language based on human understanding of the topic and then searched against the corpus with a passage-based retrieval strategy called “shortest substring ranking” (SSR) [13]. The query was then modified repeatedly in an interactive way by judging the relevance of top retrieved documents, and adding or removing terms from the original query. If the query did not retrieve enough documents, more relaxed tiers of the queries were formulated to search for more documents. The MultiText experiment gave the best manual run at TREC-6, which showed that the passage-based techniques developed by the MultiText group are also suitable for the retrieval applications in Chinese.

Other participating groups that submitted runs with automatic queries generally explored indexing and query formulation based on words or fixed-size blocks of characters. Both TREC-5 and TREC-6 results implied that simple bigram approaches, which avoid difficult issues in segmentation, can be comparable with many other more complicated techniques for the retrieval purpose.

### **2.5.3 Related Work on Chinese Question Answering**

Compared to text retrieval, research on Chinese Question Answering is much less reported in the literature. One possible reason is that multilingual QA has not been investigated as a TREC task. QA differs from text retrieval in that the task requires much more human understanding of the language. Most TREC participants and assessors are not experts in a specific non-English language. Accordingly there are

more difficulties in test collection construction, task participation and assessment.

Recently, some other workshops, such as CLEF<sup>9</sup> and NTCIR<sup>10</sup> have introduced QA in European and Japanese languages, but Chinese has not been included yet.

Nonetheless, there are some preliminary experiments conducted for Chinese QA. Li and Croft [73] implemented a system named Marsha, whose main components were similar to most existing QA systems in English: question analysis, information retrieval with the Hanquery search engine, and answer extraction.

To solve the problem that in Chinese there is no standard format in asking questions, Li and Croft defined 170 question templates, grouped into 9 question categories: PERSON, LOCATION, ORGANIZATION, DATE, TIME, MONEY, PERCENTAGE, NUMBER, and OTHER. Each incoming question was matched against the templates to determine the question type and remove the question words. The rest of the question was parsed by BBN Identifinder<sup>11</sup> to mark up named entities. The unmarked part was segmented into Chinese words with stop words removed.

The named entities and segmented words were then formulated as the query submitted to Hanquery, a Chinese version of the Inquiry [49] retrieval system developed at the University of Massachusetts, to retrieve the top 10 ranked documents. Named entities appearing in the documents were again marked up by Identifinder. Passages were defined as overlapping sentence pairs and ranked by 5

---

<sup>9</sup> Cross-Language Evaluation Forum. <http://clef.iei.pi.cnr.it:2002/>

<sup>10</sup>NII-NACSIS Test Collection for IR Systems. <http://research.nii.ac.jp/ntcir/index-en.html>

heuristics. Those heuristics favored selection of passages with more query words matched in smaller window size and appearing in the same context as the original question.

The answer to the question was extracted only from the top ranked passage. The named entities matching the question type were voted based on their distance to the matching window. The candidate closest to the matching window was chosen as the final answer.

To evaluate the Marsha system, Li and Croft used the TREC-5 and 6 Chinese Track document corpus, and collected 51 questions from Chinese students at the University of Massachusetts. Marsha answered 24 questions correctly. Since the answer to each question was extracted only from the top passage, the lower bound of mean reciprocal rank (MRR) score was equal to accuracy: 0.47.

#### **2.5.4 Need for New Experiments with MultiText**

For Chinese full-text retrieval, although the MultiText group produced the best manual run in TREC-6, query construction required users equipped with trained skills in manually creating long structured queries involving human relevance feedback. In real applications, especially in interactive settings, queries are usually short and unstructured [11]. Moreover, as analyzed in Section 2.5.1, human selected real words may not be the best queries for retrieval.

---

<sup>11</sup> A language independent software tool that can scan texts and locate named entities, including variations in

To [67] described short manual query construction using the MultiText system. The query consisted of one, two, or three of what he considered to be the most important terms. A term is either a word or a linguistic unit in other forms, e.g. Arabic numerals. The length of each term ranges from 3 to 8 characters. For each new query, he also generated two versions, the version “term as is”, and the version where each term was further split into overlapping bigrams. His results showed that both short terms and bigrams were effective for Chinese retrieval, where bigrams were slightly better. Short unstructured queries did not perform as well as long manual queries, but they required much less human intervention. Nevertheless, in To’s experiments, the short queries were still formulated manually, and bigrams were only selected from pre-segmented terms instead of long terms or sentences. Besides, feedback was done by the human-in-the-loop approach; pseudo-relevance feedback was not explored.

As for ranking algorithms, previous Chinese retrieval experiments already covered the shortest substring ranking (SSR), cover density ranking, and Okapi BM11. Several new ranking strategies introduced after TREC-6 were not evaluated against Chinese texts yet, which are explained in the next chapter.

For Chinese Question Answering, we observed that the Marsha system developed by Li and Croft used document retrieval followed by sentence-based passage selections in the search component, which, according to previous analysis in this

chapter, is less effective than passage-based retrieval. Moreover, the answer extraction heavily relied on finding and matching named entities marked up by *IdentiFinder*. The limit of *IdentiFinder* resulted in the system unable to suggest answers for the NUMBER and OTHER types. Even for named entities that matched the expected question type, because term weights were not estimated, the final answer selected was less likely to be correct. All these problems might be overcome by *MultiText* passage retrieval and statistical answer selection strategies. However, no prior work has investigated migrating the *MultiText* techniques from English to Chinese QA.

As mentioned in the previous chapter, this thesis was motivated by the need to build a Chinese QA system with *MultiText*. To address the problems involved in this task, it was necessary to re-visit Chinese full-text retrieval. In the remainder of the thesis, we investigate the unexplored issues described in this section by explaining how we employed both old and new passage-based retrieval techniques for document ranking and pseudo-relevance feedback, what topic fields and Chinese segmentation schemes were used to automatically construct queries, and how to develop and evaluate a Chinese QA system with *MultiText*.

## **Chapter 3**

# **Concepts and Methods**

All of our experiments described in this thesis were conducted with the MultiText System [13], which has been developed since 1993 and now includes a wide collection of techniques and tools for distributed information retrieval and question answering. In this chapter, we give a brief overview of the MultiText system, together with various ranking algorithms and Chinese segmentation schemes that were applied at different aspects in our experiments. In addition, we address the solutions to the specific text processing problems that arise in building a Chinese question answering system.

### 3.1 The MultiText Retrieval System

The MultiText retrieval system consists of the index engines maintaining the inverted index file structures and providing search capabilities, the text servers providing retrieval capabilities, and the marshaller/dispatcher as a client interface. MultiText also provides a special powerful query language named GCL [12] which is able to retrieve passages with arbitrary length as solution extents. GCL supports phrase queries, Boolean operators, an ordering operator that can link the start and end positions of text intervals, and a set of containment operators that can specify query structural relationships. Each retrieved solution extent is denoted as  $(p, q)$ , where  $p$  and  $q$  are the assigned integer positions of the start and ending words of the extent. As an extent, it must follow the *shortest substring rule*: it satisfies a query and does not contain any shorter substrings that also satisfy the same query.

### 3.2 Ranking Algorithms

The MultiText group has incorporated a variety of ranking algorithms for text retrieval and QA purposes, most of which are based on the passage-based schemes.

The following algorithms were used in our Chinese experiments:

1. *Shortest Substring Ranking (SSR)* [13], a passage-based document retrieval algorithm supporting a single tier of a structured query in GCL syntax;

2. *Coordination level-based Cover Density Ranking (CD)* [46], a retrieval algorithm designed on top of SSR, which is mainly used to handle short unstructured queries;

3. *Tiered Ranking (Tiered)* [45], a retrieval algorithm designed on top of SSR, similar to CD but more sophisticated;

4. *QAP* [15], a passage retrieval technique originally developed specifically for passage selection in question answering;

5. *CDR*<sup>12</sup>, a passage-based document retrieval technique developed on top of QAP for document retrieval purpose;

6. *Okapi BM25* [66], a probabilistic document ranking algorithm widely used in modern retrieval systems.

### **Shortest Substring Ranking (SSR)**

SSR is based on two assumptions: the smaller the solution extent, the more likely that the corresponding document is relevant; and, the more solution extents a document contains, the more likely that the document is relevant.

Given a query  $Q$  and a document  $D$  that contains solution extents  $(p_1, q_1), \dots, (p_n, q_n)$ , the score of  $D$  is given by:

---

<sup>12</sup> proposed by Clarke on a MultiText group seminar

$$S(D, Q) = \sum_{i=1}^n I(p_i, q_i) \quad (3.1)$$

$$\text{where, } I(p, q) = \begin{cases} \frac{k}{|q-p+1|}, (\text{if } |q-p+1| \geq k) \\ 1, (\text{otherwise}) \end{cases} \quad (3.2)$$

Here  $K$  is a *cutoff* parameter whose value ranges between 1 and 16. In the Chinese retrieval experiments  $K=16$  were used. Extents with length less than  $K$  are assigned a score of 1. In general, the score given to each solution extent is inversely proportional to the length of the extent.

### **Coordination Level-based Cover Density Ranking (CD)**

Shortest Substring Ranking itself is only capable of supporting a single-tiered structured query in GCL syntax. To support unstructured queries, it has been adopted into *Cover Density Ranking*, a family of techniques for automatically deriving high-performance queries in multiple tiers. It is shown to achieve high-precision retrieval especially from a small number of query terms. In the experiments in this thesis, two cover density methods were used: a *coordination level-based Cover Density Ranking* (CD) that measures the query terms' within-document frequency and the proximity of their co-occurrence, and the *Tiered Ranking* (Tiered) that estimates the commonality of a set of terms based on the probability that they would co-occur in a random passage with fixed length.

CD assumes that *coordination level*, the number of distinct query terms contained in the document is an important consideration for the user to determine the relevancy of a document. Specifically, for a query containing  $N$  terms, the following query tiers are generated:

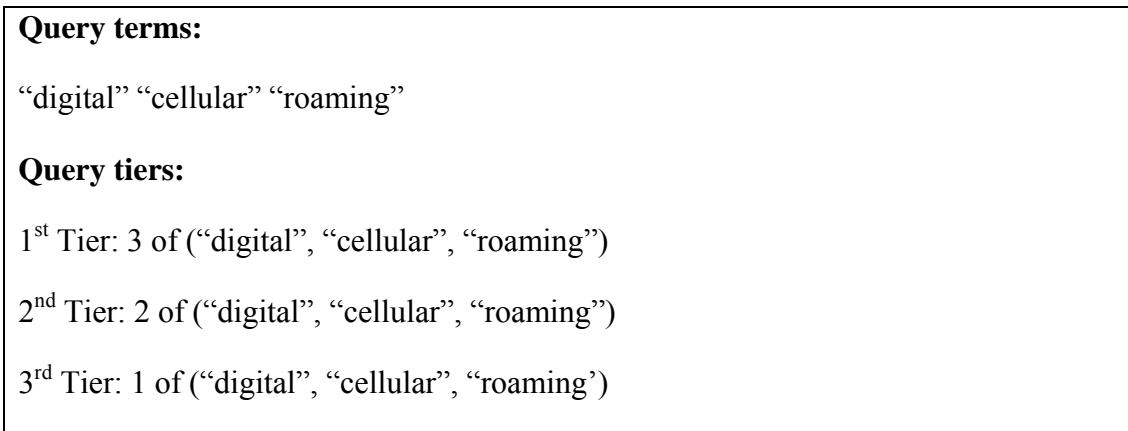
- Top tier: all of the terms;
- The second tier:  $N-1$  of the query terms;
- The  $k^{\text{th}}$  tier:  $N-(k-1)$  terms.

Figure 3.1 gives a sample of unstructured query terms as well as the query tiers generated for them. The “ $n$  of” operator, supported in GCL syntax, takes the conjunction of any  $n$  of the terms specified in the term list to retrieve documents containing at least all of the  $n$  terms. Therefore, a tier in Figure 3.1:

2 of (“digital”, “cellular”, “roaming”)

is equivalent to a Boolean query:

(“digital” and “cellular”) or (“digital” and “roaming”) or (“cellular” and “roaming”)



**Figure 3.1: Sample Unstructured Query Terms and Query Tiers (for CD Ranking)**

Each tiered query is a coordination level to which there is a length restriction that all selected terms are contained in 128 words, and the Shortest Substring Ranking is applied. Documents retrieved by a higher tier are ranked ahead of those retrieved by the next lower tier. Documents retrieved by multiple tiers are only assigned to and ranked in the earliest tier.

### **Tiered Ranking (Tiered)**

A shortcoming of CD is that the relative quality of the search terms is not considered. In the second tier of the sample queries in figure 3.1, it is possible that text fragments containing “digital” and “roaming” have a better chance of being relevant than fragments containing “cellular” and “digital”. *Tiered Ranking* (Tiered) provides an approach to this shortcoming. It firstly finds all non-empty subsets of the search terms, where in each set the terms are joined by conjunction as a candidate query. Then the precision score of each candidate is estimated assuming that a query

containing an uncommon set of terms is more likely to be relevant than a query containing a common set. Candidate queries with similar scores are joined together as a disjunction and formed as a query tier.

To estimate the commonality of a term set, Tiered Ranking computes the probability  $P$  of co-occurrence of the terms in a random fragment with fixed length of  $n$  words. Given a term set  $T = \{t_1, t_2, \dots, t_k\} \subseteq Q$ , where  $Q$  is the set containing all search terms,  $P$  is formulated as the following, based on the assumption that all term occurrences are uniformly and independently distributed:

$$\begin{aligned}
 P &= \prod_{i=1}^k (1 - (1 - p_{t_i})^n) \\
 &\approx \prod_{i=1}^k n \cdot p_{t_i} \\
 &= n^k \prod_{i=1}^k p_{t_i}
 \end{aligned} \tag{3.3}$$

where,

$$p_{t_i} = \frac{f_{t_i}}{N}, \text{ (the probability that any precision term is } t_i) \tag{3.4}$$

$f_{t_i}$  = the number of occurrences of term  $t_i$  in the corpus

$N$  = the corpus size

$n$  = the fixed length the fragment

The score assigned to each candidate query is the self-information of  $P$  (i.e.,  $-\log(P)$ ) and the query tiers are defined as following, with  $n=128$  in our experiments:

- Top tier: all of the terms;
- The second tier: query candidates with score approximately half as good as tier 1;
- The  $k^{th}$  tier: query candidates with score approximately half as good as tier  $k-1$ .

For the query terms in figure 3.1, the new type of tiers is generated as shown in Figure 3.2.

|                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>Query terms:</b></p> <p>“digital” “cellular” “roaming”</p> <p><b>Query Tiers:</b></p> <p>1<sup>st</sup> Tier: “digital” and “cellular” and “roaming”</p> <p>2<sup>nd</sup> Tier: “digital” and “roaming”</p> <p>3<sup>rd</sup> Tier: (“cellular” and “digital”) or (“roaming” and “cellular”)</p> <p>4<sup>th</sup> Tier: “digital” or “roaming”</p> <p>5<sup>th</sup> Tier: “cellular”</p> |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**Figure 3.2: Sample of Query Terms and Query Tiers (for Tiered Ranking)**

## QAP

The QAP ranking algorithm is a newer technique developed specifically for question answering. It is a probabilistic approach that views the whole document corpus as a long single string to locate the best solution extent by trading off the passage length, the number of query terms contained in the passage, and the IDF-like term weights. Such an extent is therefore a “hotspot” where query terms are clustered in close

proximity. Similar to SSR, the passage found by QAP can also be an arbitrary substring that is constrained by document boundaries but does not necessarily conform to semantic units, such as sentences or paragraphs.

Given a query term set:  $T = \{t_1, t_2, \dots, t_k\} \subseteq Q$ , where  $Q$  is the set containing all search terms, the score for an extent  $H$  with length  $l(H)$  containing the terms in  $T$  computed by QAP is essentially the same as Tiered Ranking, except that  $l(H)$  is not fixed. By replacing  $n$  in formula 3.3 by  $l(H)$ , and taking the self-information of  $P$ , the score for an extent  $H$  is therefore given by:

$$\sum_{i=1}^k \log(N / f_{t_i}) - k \log(l(H)) \quad (3.5)$$

Unlike SSR that combines the weights of best passages a document contains and returns a ranked list of documents, QAP simply returns a ranked list of passages with the highest scores, where no two passages are taken from the same document. The passages it retrieves are usually less than 50 bytes.

## **CDR**

As the QAP algorithm was originally designed to fetch “hotspots” that are most likely to contain the answer to a question directly from the corpus. It skips the document retrieval procedure in question answering. However, if QAP is applied to document retrieval, a document has to be scored the same as the best hotspot it contains. In other words, QAP may be treated as a document-retrieval approach by expanding the

window around the hotspot to the document boundaries at both ends so that to include an entire document. A recently proposed modification to QAP is called CDR, an algorithm that ranks a document by summing up the scores of all non-overlapping hotspots it contains calculated by the QAP approach.

### Okapi BM25

Okapi BM25, a well-known and effective probabilistic algorithm for document retrieval, is supported by the MultiText System so that the retrieval effectiveness of passage-based techniques can be compared with other existing probabilistic measures. The current MultiText implementation of Okapi BM25 is based on the description by Roberston *et al.* [66] with typical parameter values ( $b=0.75$ ,  $k_1=1.2$ ,  $k_2=0$ ,  $k_3=\infty$ ). Given a term set  $Q$ , the score of a document  $d$  is computed by:

$$\sum_{t \in Q} w^{(1)} q_t \frac{(k_1 + 1)d_t}{K + d_t} \quad (3.6)$$

where

$$w^{(1)} = \log\left(\frac{D - D_t + 0.5}{D_t + 0.5}\right) \quad (3.7)$$

$D$  = number of documents in the corpus

$D_t$  = number of documents containing  $t$

$q_t$  = frequency that  $t$  occurs in the topic

$d_t$  = frequency that  $t$  occurs in  $d$

$$K = k_1((1 - b) + b \cdot l_d / l_{avg})$$

$l_d$  = length of  $d$

$l_{avg}$  = average document length

This measure accounts for the document term frequency, query term frequency and document length to look for the similarity between a document and a given query.

The MultiText implementation extended Okapi BM25 with the support for queries containing arbitrary phrases. As Chinese words are comparable to English phrases with MultiText re-encoding and indexing, such “phrases” can be treated as individual terms in our experiments.

### 3.3 Chinese Segmentation

Since the MultiText system supports phrase search, in our experiments the index was based on individual characters, following what the group did at TREC-6. This approach allowed us to experiment with various segmentation methods. The segmentation methods we compared, from the simplest to the most advanced, were overlapping bigrams (Bigrams), basic variable-length n-grams (BVN) based on mutual information, and a dictionary-based segmentation by the UPenn LDC Segmenter (LDC).

**Overlapping Bigrams (Bigrams)**

As mentioned before, this method refers to splitting a sequence of consecutive characters in the Chinese texts into overlapping character-pairs. For example, suppose “ABCD” is a 4-character sequence. The segmented terms are: “AB”, “BC”, and “CD”. This method is based on the observation that more than half of Chinese words are composed of only two characters. It does not require any statistics or dictionary and thus is very easy to implement. However, such bigrams suffer from over-generation as there is no heuristics used to decide whether a bigram is meaningful or not, and furthermore, those Chinese words that are truly single characters cannot be generated.

**Basic Variable-length N-grams (BVN)**

This method is similar to the statistic segmentation approach described by Chen *et al.* [1]. As their method can only extract bigram Chinese words, we have adapted it into the following procedure:

A given Chinese string is scanned from the beginning. At each character currently reached, the mutual information  $I(c_1, c_2)$  is computed between the current character  $c_1$  and the next one  $c_2$ . If  $I(c_1, c_2)$  is above a threshold, then keep  $c_1$  and  $c_2$  together and continue to examine the next character pair in the same way, otherwise  $c_1$  and  $c_2$  are split apart.

In our experiments, we used the same threshold value 7 that Chen *et al.* selected.

With this segmentation method, more meaningful words (not restricted to 2-character words) can be generated than using overlapping bigrams. However, lack of a lexicon makes it fail to acquire rare words and uncommon named entities.

### **UPenn LDC Segmenter (LDC)<sup>13</sup>**

This Chinese LDC segmenter developed at the University of Pennsylvania is a more advanced tool for Chinese segmentation with high-accuracy. It uses a lexicon containing a list of Chinese words and their relative frequency information. Only terms contained in the list can be generated, but users can easily modify the lexicon according to their needs to improve the segmentation performance. In our experiments of applying LDC segmenter to the topic and description fields in the Chinese track topics, 95% of the terms were correctly segmented.

## **3.4 QA in Chinese with MultiText**

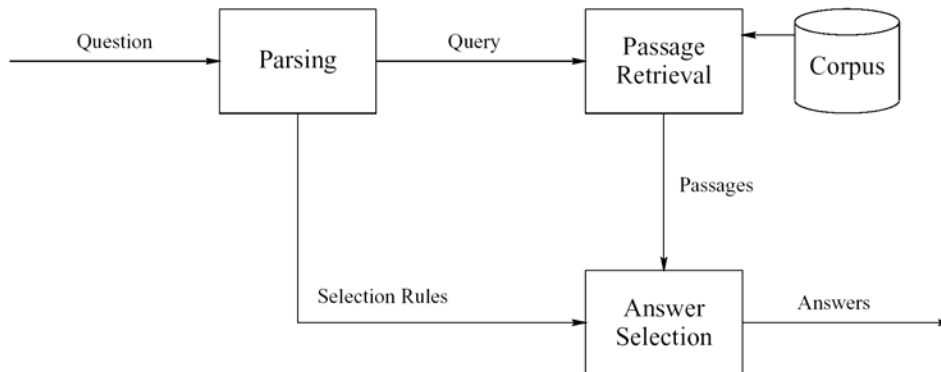
### **3.4.1 The QA System Architecture**

The basic version of the MultiText QA system, developed at TREC-8, only used term and document statistics to drive both passage retrieval and answer selection. In later years, this approach was augmented with natural language processing heuristics, such

---

<sup>13</sup> [www ldc.upenn.edu/ctb](http://www ldc.upenn.edu/ctb)

as question parsing, categorization, and pattern matching. Figure 3.3 gives a simplified overview of the system.



**Figure 3.3: The MultiText QA system architecture**

The Chinese QA system has a similar architecture. It employs the same passage retrieval strategy and a simplified version of the answer selection heuristic. For question analysis, it also classifies the questions to determine answer categories, but to avoid the complexity in recognition of Chinese part-of-speech, it does not employ a parser.

### 3.4.2 Question Analysis

In this component, we defined the same 7 question categories: PERSON, LOCATION, ORGANIZATION, DATE, TIME, NUMBER, and OTHER. This is slightly different from the Marsha system developed by Li and Croft in that we considered “percentage”, “currency” and “number” as all belonging to the NUMBER category.

Unlike Marsha, which matched a question against 170 predefined templates, we determined a question's type based on a set of simple heuristics. For example:

A question containing the question word “谁” (who) expects the answer to be a PERSON;

A question containing the question word “多少” (how much/ how many) expects the answer to be a NUMBER;

A question containing question word “哪里” (where) expects the answer to be a LOCATION;

If a question contains question words like “什么”, “哪些”, “哪个”, “哪”, or “何”, which all have the meaning “what” or “which”, even though they may appear at any position in the question, usually the word indicating the question type is located close to the question word following some rules as shown in Table 3.1. For simplicity, let “what” represent the questions words, “**XX**” be the question type word, “*is*” stand for the Chinese word “是”, and “.....” be anything else in the rest part of the question.

|   | Question Format                         | Chinese question example                                        | Question type |
|---|-----------------------------------------|-----------------------------------------------------------------|---------------|
| 1 | <u>What</u> <b>XX</b> <i>is</i> ..... ? | 什么花是荷兰的国花?<br>(What is the national flower of the Netherlands?) | OTHER         |
| 2 | ..... <u>what</u> <b>XX</b> ..... ?     | 香港何时回归的中国?<br>(When did Hong Kong return to China?)             | DATE          |
| 3 | ..... <i>is</i> <u>what</u> <b>XX</b> ? | 人类第一次登上月球是什么时候?<br>(When was the first man on the moon?)        | DATE          |
| 4 | ..... <b>XX</b> <i>is</i> <u>what</u> ? | 中国的首都是什么?<br>(What is the capital city of China?)               | LOCATION      |

**Table 3.1: General Rules for Determining Question Types with “What” Like Question Words**

It is easy to see that, no matter whether the question word appears in the beginning, in the middle, or at the end of the question, the word indicating the question type is usually located directly after the question word, or followed by “是” (is) and together in front of the question word. Of course due to the complexity of the Chinese language, there are many other ways of asking questions. Nonetheless, the above heuristics can reduce the amount of manual work required to create question templates.

When the question category is decided, the question words are eliminated. In order to evaluate the effectiveness of statistical answer selection with simple pattern matching strategies, we did not use a named entity markup tool, such as BBN *Identifinder* that is used by Marsha, but based the answer selection on simple heuristics and answer patterns created by ourselves.

### 3.4.3 Passage Retrieval

In the search component, the segmented words in the question are submitted together to search for satisfying passages. Some question types may suggest specific answer patterns. For instance, for the type PERSON, if after removing the question words, the remaining part of the question is **R**, then the following patterns may be expected in the corpus:

*..R..PersonName.*

*..R..是(is) PersonName*

*PersonName 是(is) ..R..*

If there are defined patterns for a question type, the regular expressions of the patterns are calculated to search against the corpus. If no passages are found or the question does not have a defined pattern, the question is segmented into bag of query terms to retrieve up to 10 passages with the QAP algorithm described in the previous section. The window size of each matched passage (hotspot) is expanded by 10 characters at both ends.

### 3.4.4 Answer Extraction

The basic heuristic developed by MultiText to determine the score for a candidate answer term  $t$  is formulated as:

$$w_t = pf_t \cdot \log\left(\frac{N}{f_t \cdot (\text{loc}(H_i, t) + 1)}\right) \quad (3.8)$$

where

$pf_t$  = the frequency that  $t$  occurs in the retrieved passages

$N$  = the corpus size

$f_t$  = the frequency that  $t$  occurs in the corpus

$H_i$  = the hotspot in the  $i^{\text{th}}$  ranked passage, where  $1 \leq i \leq 10$

$$loc(H_i, t) = \begin{cases} 0, & \text{if } t \text{ appears in the hotspot} \\ \text{the distance from } t \text{ to the hotspot } H_i, & \text{otherwise} \end{cases}$$

*distance* = number of term positions

All candidate terms are extracted from the retrieved passages. If the passages are well-segmented, the answer selection heuristic does not even necessarily require a question classification because all questions can be treated as in the OTHER type and all terms appearing in the passages are viewed as potential answers to be scored.

In order to be comparable with the Marsha system, the goal of our QA system is to return exact term answers instead of short passages limited to 50 or 250 bytes. A score assigned to a 50 or 250-byte passage is determined by accumulating the scores of all terms it contains, while a score given to an exact answer is determined by the term score only, assuming that only single-term answers are expected.

To return exact answers the system needs accurate segmentation of the retrieved passages or effective recognition of named entities, especially with question classification and pattern matching. It is difficult because some named entities, such as person names and numerals, are neither regular Chinese words nor written with

explicit signals that can help determining the word boundaries. Even advanced segmenters can not detect them. Therefore, special processing is needed. For example, a name recognizer is implemented with the following tiering method:

- Tier 1: special names, e.g. predefined foreign name translations that can be found from most modern Chinese dictionaries, for example, “阿姆斯特朗” (Armstrong);
- Tier 2: regular Chinese names:
  - A surname: defined from a fixed surname set containing around 300 Chinese surnames;
  - A first name: one or two Chinese characters (could be any characters);
  - A full Chinese name: A surname followed by a first name;
- Tier 3: anything else.

Similarly, a Chinese cardinal number is usually formulated as a sequence of numeral characters sometimes followed by a unit word. A number recognizer can sequentially scan from the first occurrence of numerals and stop at the first non-numeral character. This character is matched against a pre-stored unit set. If it is a unit word, then it is included as part of the number; otherwise only the numeral part is returned as the number term. A Chinese ordinal is even simpler to recognize because it simply adds a “第” in front of a cardinal. The only difficulty is constructing the unit set. Different nouns require different unit words to “decorate” them. Unlike Chinese

surnames, there is no existing database for Chinese unit words. More human effort is required to populate the database. We have collected nearly 60 unit words and they seem sufficient to serve most nouns.

With question classification, the weight of a term generated corresponding to the designated category (e.g., the terms in tier 1 and tier 2 for the PERSON category) is multiplied by a number greater than one, while the score of a term that is not an instance of the given category is multiplied by a number less than one. By doing this, candidates matching a category are likely to be ranked ahead of others, and in the case that the category is OTHER, or if the terms of a category is not found because the instance list is incomplete (for example, the expected answer is a city name that has not yet been included in the LOCATION type), the candidates are ranked as if by using the heuristic formula 3.8 alone.

# **Chapter 4**

## **Experimental Setup**

Our experiments included both the evaluation of our Chinese QA system and the techniques for Chinese full-text retrieval. To set up the experiments, first of all the test collections were defined, including the document collections, topics for text retrieval, question sets for question answering, relevance judgments, and performance evaluation measures. Then the system specifications were given. Both full-text retrieval and question answering systems required the provision of search capabilities with the Chinese texts re-encoded and indexed. At the user end, online interfaces were created to facilitate both user interaction and automatic evaluations.

## 4.1 Document collections

Two document collections were used. The main corpus was the TREC Chinese track collection as mentioned before (henceforth called TREC Chinese corpus). It contained 164,811 Chinese articles selected from the People's Daily newspaper and the Xinhua newswire from 1991 to 1995. The GB encoded raw text was marked up with SGML tags and was about 170 megabytes in size, and no segmentation information was provided. Figure 4.1 gives an example of a document in the corpus.

A secondary corpus was built from Web data crawled by Clarke in the MultiText group from the classified sites originating from the following commercial directory links:

<http://dir.sina.com.cn/>

<http://dir.sohu.com/>

<http://search.163.com/>

<http://cn.yahoo.com/>

Starting with the above links as a seed set, pages were collected in breadth-first order [59]. At a given depth from the seed set, pages were gathered in a random order. Duplicate pages were removed with only one left. After eliminating the noise that is neither Chinese nor English text, each crawled page was transformed into the same markup format as TREC documents. The corpus to be indexed was about 17 gigabytes in size (henceforth called Web Chinese corpus).

---

```
<DOC>
<DOCNO> pd9202-2984 </DOCNO>

<HL>大型航天试验通信网初步建成精度达百万分之一秒，误差不超过十万分之一</HL>
<TEXT>

新华社北京 2 月 1 4 日电（记者许志敏、通讯员黄刚）经过广大国防科技工作者和解放军官兵 3 0 多年的努力，一个具有国际先进水平的大型航天试验通信网已在我国初步建成，并将担负下月“澳星”发射的通信保障任务。

国防科工委通信部门负责人告诉记者，目前，这个通信网遍及全国 2 3 个省、自治区、市，拥有近 1 0 万套先进的通信设备和数百个专用台站，以及远洋航天测量船队等，构成陆海空多层次、大容量的立体通信网络，可满足我国各种航天试验活动的需要。

据介绍，航天试验通信网近年来还积极应用国内外高新技术成果，朝以数字程控交换处理中心为核心、以卫星通信为骨干信道方向发展，集多种传输手段于一体，具有话音、数据、图像等多种传输功能及移动通信能力，在上万公里范围内时间精度可达到百万分之一秒，其传输误差不超过十万分之一。专家们认为，它的通信保障能力已达到国际先进水平。

这个通信网采用了先进的图像传输和数字处理技术，可及时通过电视屏幕，观察和指挥千里之外发射现场的设备运行及人员工作情况。为适应对外发射服务需要，通信网还建有外事通信系统和海事卫星测控通信系统。
</TEXT>
</DOC>
```

---

**Figure 4.1: A Document in the TREC Chinese Corpus**

## 4.2 Topics

For full-text retrieval, the TREC-5 and 6 Chinese topics were used. Specifically, there were 28 topics for TREC-5 and 26 for TREC-6. The format for each topic was similar to an English ad hoc topic, which also contained a “title”, a “description” and a “narrative” field. However, the “title” field in a Chinese topic was a long phrase or a sentence, while the “description” field contained several keywords. This organization is the opposite of that used in the English topics described in Chapter 2. Besides, both Chinese version and English translation were given for each topic field. Figure 4.2 shows topic 28 from TREC-5 as an example.

---

```
<DOC>

<DOCNO> Number: CH28</DOCNO>
<E-title> The Spread of Cellular Phones in China
<C-title> 移动电话在中国的成长

<E-desc> Description:
digital, cellular, cellular phone, net, automatic roaming

<E-narr> Narrative:
A relevant document contains the following kinds of information:
the number of cellular phone users, area coverage, or how PSDN
is implemented for national cellular communication. A non-relevant
document includes reports on commercial manufacturers or brand name
cellular phones.

<C-desc> Description:
    数字, 蜂窝式, 移动电话, 网络, 自动漫游

<C-narr> Narrative:

    相关文件应包括下列信息: 中国移动电话用户数,
    覆盖地区, 中国如何以数据分组交换网覆盖
    全国移动电话的通讯. 不相关文件则包括 有关
    制造移动电话厂商的报道, 以及移动电话的
    厂牌.

</DOC>
```

---

**Figure 4.2: Chinese Topic 28 from TREC-5**

### 4.3 Question Sets

In order to compare the performance of our Chinese QA system with that of Marsha developed by Li and Croft, we requested the 51 Chinese questions used for the evaluation of Marsha from Li (henceforth called UMass questions). According to their description [73], 26 of the questions were selected from 240 questions collected from Chinese students in the Department of Computer Science at the University of Massachusetts, because only those were known to have answers in the TREC Chinese corpus. The remaining 25 questions were created by reformulating some of the 26 questions.

In addition to the Marsha questions, we also created a new question set containing 149 questions. Some were selected and modified from the Chinese Millionaire game question corpus<sup>14</sup>, and others were collected from Chinese students in the School of Computer Science at the University of Waterloo. These questions were not known to have answers in the TREC Chinese corpus.

### 4.4 Relevance Judgments

For full-text retrieval, the relevance judgments for the 54 TREC Chinese topics were those obtained by NIST using the pooling method. The top 100 documents for each

---

<sup>14</sup> <http://www.hkatv.com/infoprogram/millionaire/question/0815.html>

topic retrieved by each submitted run were collected into a pool, and human assessors manually judged the relevance of each document. The relevance information of documents in the pool was then collected in a file, which was used as a standard to compare against each submission. Unjudged documents in the corpus were considered not relevant.

For question answering, we first manually judged the correctness of each answer to a given question. To facilitate automatic judgments, the set of answers that were judged correct was created as a set of perl string-matching patterns [39]. An answer string matching any pattern of its question was viewed correct, and was judged incorrect otherwise.

## 4.5 Evaluation Measures

### 4.5.1 Interpolated Recall-Precision Averages

To compare the retrieval performance of ad hoc runs, a recall-precision curve is used based on interpolated recall-precision averages as described in Chapter 2 [36]. The interpolated precision at recall level  $R_i$  is defined as the maximum precision value at all recall levels between  $R_{i-1}$  and  $R_i$ . As in common practice, 11 standard recall levels are defined: 0.0, 0.1, ..., 0.9, 1.0, and the interpolated average precision over all test topics at each given recall level is plotted. This measure roughly reflects the retrieval behavior of a particular run over the entire spectrum of recall.

### 4.5.2 Mean Average Precision (Non-Interpolated)

The non-interpolated average precision [36] is one of the primary measures used to compare the overall performance of distinct ad hoc runs. The value is calculated by aggregating the precision values obtained when each relevant document is retrieved, and then dividing the sum by the total number of relevant documents.

### 4.5.3 Average Precision at a Given Document Cutoff Value

For an ad hoc run, the average precision at document cutoff value  $k$ , denoted by  $p@k$ , is given by averaging the precision value after  $k$  documents are retrieved over all topics.

### 4.5.4 Average Cover at a Given Document Cutoff Value

The *Cover Measure*, recently proposed by Cormack in the MultiText group, differs from the traditional precision measure in that, among a ranked list of  $k$  documents or passages retrieved, if any one of the  $k$  units contains a piece of information satisfying the user's request (which refers to a document that is relevant to the given topic in context of document retrieval, or a passage that contains a correct answer to a given question in context of question answering), the cover at document (or passage) level  $k$ , is 1, otherwise it is 0. The average cover at a given document cutoff value  $k$ , denoted by  $c@k$ , is obtained by averaging the cover value after  $k$  documents are retrieved over all topics.

### 4.5.5 Mean Reciprocal Rank (MRR)

The *mean reciprocal rank* (MRR) is an important traditional measure used to evaluate the performance of a QA system [29]. In early years of QA tracks, each participant was required to submit a ranked list of 5 [document-id, answer-string] pairs for each question. Every pair was judged as *correct* if the answer string contained the correct answer and the associated document also supported it, *incorrect* if the answer string did not contain the answer, and *unsupported* if the answer string contained the correct answer but the document did not support it. In the *strict* evaluation measure, unsupported answers were viewed incorrect; while in the *lenient* measure, unsupported answers were judged correct.

To calculate the MRR for a run, the score given to each question is equal to the reciprocal of the rank where the first correct answer is found, if there is any; otherwise its score is 0. Therefore a question can only receive a score of 1, 0.5, 0.33, 0.25, 0.2, or 0. The mean of all questions reciprocal ranks is assigned as the MRR score for a run.

### 4.5.6 Accuracy

In recent QA tracks, only one response is allowed to return to each factoid question. The main evaluation score for a run in such tasks is *accuracy* [33], which is defined as the fraction of questions that are judged correct.

## 4.6 System Setup

### 4.6.1 Re-Encoding and Indexing of the Chinese Texts

In the raw text of both TREC and Web Chinese corpora, to encode each Chinese character with the GB coding scheme, two bytes are used, where each byte is represented by an 8-bit char with a value greater than 128. As the original implementation of the MultiText system was only able to index printable ASCII character strings, in our experiments we used To's [67] solution that each 2-byte Chinese character was re-encoded into a 6-byte ASCII string, which consisted of the character's hexadecimal value with prefix "xx". Each re-encoded string was followed by a space. Therefore, every Chinese character could be treated as an "English word", and each multi-character Chinese word could be viewed as an "English phrase". The non-Chinese part, such as English texts, Arabic numerals, and SGML tags, was left unchanged. For example, the text "<title> 中国, China </title>" was converted into "<title> xxD6D0 xxB9FA , China </title>". The index of the text collection was based on individual characters and English words to allow for the flexibility in segmentation. The searching for a Chinese word was by using the phrase searching capability provided by MultiText, where a phrase was defined as a sequence of adjacent English words or Chinese characters.

### 4.6.2 User Interfaces

#### User Interfaces for Full-Text Retrieval

An online interface for full-text retrieval was originally developed for the TREC-5 ad hoc experiments [9]. It was then slightly modified to support displaying Chinese characters for TREC-6 Chinese track, as shown in Figure 4.3. For each query submitted through the interface, a ranked list of documents (document ID and an expanded hotspot passage) was returned to the user. This allowed the user to quickly look through the hotspots of each retrieved document and interactively formulate the manual queries.

## “北京奥运”

---

---

[pd9110-2168](#)

[...]

各地收件人的手中，EMS 已被国际奥委会指定为一九九二年巴塞罗那奥运会的速递信使。

北京奥运会申办委和远东及南太平洋地区残疾人运动会组委会着手工作以来，大量国际、国内邮件需 [...]

[pd9111-2555](#)

[...]

挚、友好的交谈。

国际奥委会副主席、中国奥委会主席何振梁参加了会见。

海格曼女士是应北京奥运会申办委员会的邀请，前往观摩正在广东举行的第一届世界女足锦标赛途中顺访北京的。

</TEXT>

[...]

[pd9202-3308](#)

[...]

上，运动员和首都大学生代表将“万众一心申办奥运奔向2000年”的签名横幅递交给了北京奥运会申办委员会万副全秘书长。全国54支男女大学生排球队将从2月25日起分别在北 京、 [...]

[pd9205-4027](#)

[...]

张征东) 由北京市常务副市长、北京2000年奥运会申办委员会常务副主席张百发率领的北京奥运会申办代表团一行6人，今天在这里出席了国际奥委会与2000年奥运会申办城市联席会 [...]

[pd9211-3513](#)

[...]

Figure 4.3: TREC-6 Chinese Track User Interface

This interface could only accept one query at each time. To facilitate our experiments with a large set of topics, we adapted the interface to accept a query file as the input, which could contain multiple queries, either manually or automatically constructed. The retrieval result is a ranked list of document IDs for each query. When clicking on any document ID, the whole original document text can be displayed in a new window.

For ranking algorithms such as QAP, CDR and Okapi BM25, there are not explicit multiple tiers for each query. The interfaces have similar appearances, as illustrated in Figure 4.4.

For ranking algorithms CD and Tiered, multiple query tiers are generated for each query. Interfaces are created to facilitate users viewing each query tier together with the document list retrieved at each tier. The interfaces for CD and Tiered are shown in Figure 4.5 and 4.6, respectively.

---

load queries from file: "sq.terms.gcl"

---

**"最惠国待遇" "人权" "分离"**

[pd9304-2418](#)  
[CB011028-BFJ-483-1011](#)  
[CB016027-BFJ-372-946](#)  
[pd9303-1753](#)  
[pd9303-143](#)  
[pd9207-4844](#)  
[CB044021-BFJ-364-227](#)  
[pd9211-4838](#)  
[pd9202-4463](#)  
[pd9105-172](#)

*Topic 1 done.*

*Number of retrieved documents (excluding topics): 10*

---

**"中国" "台湾" "一国两制"**

[pd9103-2275](#)  
[pd9207-4207](#)  
[pd9309-404](#)  
[pd9309-1901](#)  
[pd9303-1231](#)  
[CB011022-BFW-1083-169](#)  
[pd9205-4992](#)  
[pd9309-1271](#)  
[pd9106-364](#)  
[pd9103-2030](#)

*Topic 2 done.*

*Number of retrieved documents (excluding topics): 10*

---

**"核电站" "大亚湾" "安全"**

[CB055006-BFW-134-412](#)  
[pd9306-1196](#)  
[pd9103-2532](#)  
[CB053006-BFW-693-161](#)  
[CB060018-BFW-950-536](#)  
[pd9205-3936](#)

**Figure 4.4: Sample of a Chinese Text Retrieval Interface for QAP/CDR/Okapi BM25**

Load queries from file: "auto\_desc\_LDC.gcl"

---

---

“最惠国待遇” “中国” “人权” “经济制裁”  
 “分离” “脱钩”

*weakened query: 5 of (“最惠国待遇”, “中国”, “人权”, “经济制裁”, “分离”, “脱钩”) <[128]*

*weakened query: 4 of (“最惠国待遇”, “中国”, “人权”, “经济制裁”, “分离”, “脱钩”) <[128]*

[CB021029-BFW-1506-567](#)  
[CB011028-BFJ-483-1011](#)  
[CB027013-BBJ-3024-213](#)  
[CB031014-BFJ-385-370](#)  
[CB025002-BFW-608-155](#)  
[CB002001-BFJ-1654-17](#)  
[CB021027-BFW-1332-468](#)  
[CB015027-BBW-708-961](#)  
[CB016027-BFJ-372-946](#)  
[CB029027-BBW-701-1051](#)  
[CB044029-BFW-861-588](#)  
[CB045005-BBJ-1969-342](#)  
[CB007005-BCJ-575-175](#)  
[CB012025-BFW-2925-360](#)  
[CB030024-BFW-803-303](#)  
[CB030024-BFW-783-350](#)  
[CB021009-BBJ-1265-4](#)  
[CB036011-BFJ-426-328](#)

*weakened query: 3 of (“最惠国待遇”, “中国”, “人权”, “经济制裁”, “分离”, “脱钩”) <[128]*

[pd9207-4844](#)  
[pd9107-1848](#)  
[pd9307-282](#)  
[pd9305-2949](#)  
[pd9302-171](#)  
[pd9304-2418](#)

**Figure 4.5: Chinese Text Retrieval Interface for CD Ranking**

Load queries from file: "auto\_desc\_LDC.gcl"

New Query from file

in Article
Search
Rank
Query Builder
???
Clear

---

"最惠国待遇" "中国" "人权" "经济制裁"  
"分离" "脱钩"

---

*weakened query: (("脱钩""分离""经济制裁""人权""最惠国待遇"))<[128]*

---

*weakened query: (("脱钩""分离""经济制裁""中国""最惠国待遇"))<[128]*

---

*weakened query: (((("脱钩""经济制裁""人权""中国""最惠国待遇")+("脱钩""分*

---

*weakened query: (((("脱钩""分离""人权""中国""最惠国待遇")+("分离""经济*

---

*weakened query: (("脱钩""分离""经济制裁""最惠国待遇"))<[128]*

---

*weakened query: (((("脱钩""经济制裁""人权""最惠国待遇")+("脱钩""分离""*

---

*weakened query: (((("脱钩""分离""人权""最惠国待遇")+("分离""经济制裁""*

---

*weakened query: (((("脱钩""经济制裁""中国""最惠国待遇")+("脱钩""分离""*

---

*weakened query: (((("脱钩""分离""中国""最惠国待遇")+("分离""经济制裁""*

---

*weakened query: (((("脱钩""人权""中国""最惠国待遇")+("经济制裁""人权""*

---

[CB021029-BFW-1506-567](#)  
[CB011028-BEJ-483-1011](#)  
[CB027013-BBJ-3024-213](#)  
[CB031014-BEJ-385-370](#)  
[CB025002-BFW-608-155](#)  
[CB002001-BEJ-1654-17](#)  
[CB021027-BFW-1332-468](#)  
[CB015027-BBW-708-961](#)  
[CB016027-BEJ-372-946](#)  
[CB029027-BBW-701-1051](#)  
[CB044029-BFW-861-588](#)  
[CB045005-BBJ-1969-342](#)  
[CB007005-BCJ-575-175](#)  
[CB012025-BFW-2925-360](#)  
[CB030024-BFW-803-303](#)  
[CB030024-BFW-783-350](#)  
[CB021009-BBJ-1265-4](#)  
[CB036011-BEJ-426-328](#)

---

*weakened query: (("分离""人权""中国""最惠国待遇"))<[128]*

Figure 4.6: Chinese Text Retrieval Interface for Tiered Ranking

### **User Interface for Question Answering**

The user interface for question answering was originally designed by Lynam in the MultiText group to facilitate answering user's input questions and manual judgments. We slightly modified the underlying code to provide more useful information on the screen and to support QA in Chinese. After typing a question and clicking on the "Answer" button with the number of results selected, a list of expected number of ranked answers is given to the user, which includes the following information as shown in Figure 4.7: the answer score, the answer term, the supporting document and hotspot passage, and the judgment options if "eval" is chosen for answer evaluation. A user can read the answer term, the passage, and even the full document by clicking on the document ID to determine the correctness of each answer. The judgment result is automatically saved for future evaluations.

Input Question

史记的作者是谁?

Answer 10 results New? eval

| Rank | Score    | Answer                                                                         | Judgment                                                                                        |
|------|----------|--------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
| 1    | 8.52e+04 | 司马迁<br><a href="#">pd9109-2192</a> 一句, 司马是指《史记》的作者司马迁。班是写《                     | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 2    | 6.52e+01 | 历史<br><a href="#">CB018020-BFJ-983-56</a> 有关厄尔尼诺现象的历史记载可以回溯到一个                 | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 3    | 2.31e+01 | 记载<br><a href="#">CB018020-BFJ-983-56</a> 有关厄尔尼诺现象的历史记载可以回溯到一个                 | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 4    | 5.00e+00 | 通史<br><a href="#">pd9108-1306</a> 一部纪传体通史《史记》一书的作者。 草圣: 张芝, 汉                  | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 5    | 4.89e+00 | 瑰丽<br><a href="#">pd9309-2728</a> 它集雕刻绘画、文史记载、国风民俗于一身, 博大精深、瑰丽恢宏。作者运用辩证唯物主     | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 6    | 4.56e+00 | 精深<br><a href="#">pd9309-2728</a> 它集雕刻绘画、文史记载、国风民俗于一身, 博大精深、瑰丽恢宏。作者运用辩证唯物主     | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 7    | 4.49e+00 | 辩证唯物主义<br><a href="#">pd9309-2728</a> 它集雕刻绘画、文史记载、国风民俗于一身, 博大精深、瑰丽恢宏。作者运用辩证唯物主 | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 8    | 4.32e+00 | 媲美<br><a href="#">pd9111-3057</a> 认为它可与司马迁的《史记》相媲美, 是中国                        | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 9    | 4.14e+00 | 国风<br><a href="#">pd9309-2728</a> 它集雕刻绘画、文史记载、国风民俗于一身, 博大精深、瑰丽恢宏。作者运用辩证唯物主     | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |
| 10   | 4.05e+00 | 翔实<br><a href="#">pd9112-1121</a> 忆录应该做到翔实的历史记载和生动的文学笔                         | correct <input type="radio"/> incorrect <input type="radio"/> unsupported <input type="radio"/> |

**Figure 4.7: Chinese Question Answering Interface**

# **Chapter 5**

## **Experimental Results and**

## **Analysis**

### **5.1 Experiments on Full-Text Retrieval**

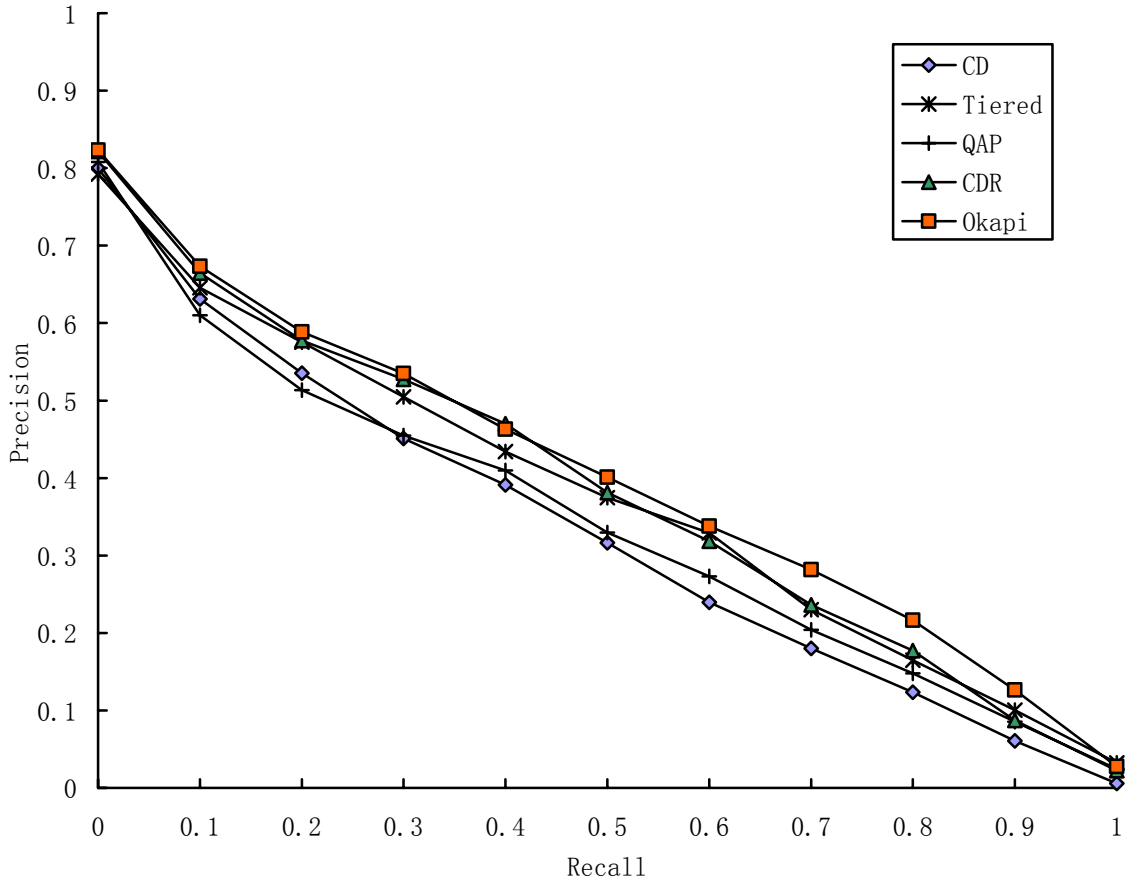
To compare the effectiveness of existing retrieval techniques supported by the MultiText system, our preliminary experiments used only simple short queries manually created by To [67]. We then moved on to the automatic generation of queries combining different topic fields with different segmentation schemes. The manual and automatic queries that produced the best runs were then used for pseudo-relevance feedback with and without Web reinforcement. The method was adapted

from the question answering passage retrieval (QAP) and answer extraction heuristics. Finally data fusion of several runs was applied to produce our best result that was comparable to most of the TREC-6 Chinese track submissions.

### 5.1.1 Runs with Short Manual Queries

As mentioned previously, To [67] used two versions of short queries for TREC-5 and 6 Chinese Topics 1-54: as-is terms and bigrams. In this section the effectiveness of each query set is evaluated. Moreover, a direct comparison among five relevance ranking algorithms CD, Tiered, QAP, CDR and Okapi BM25 using the same query set was made.

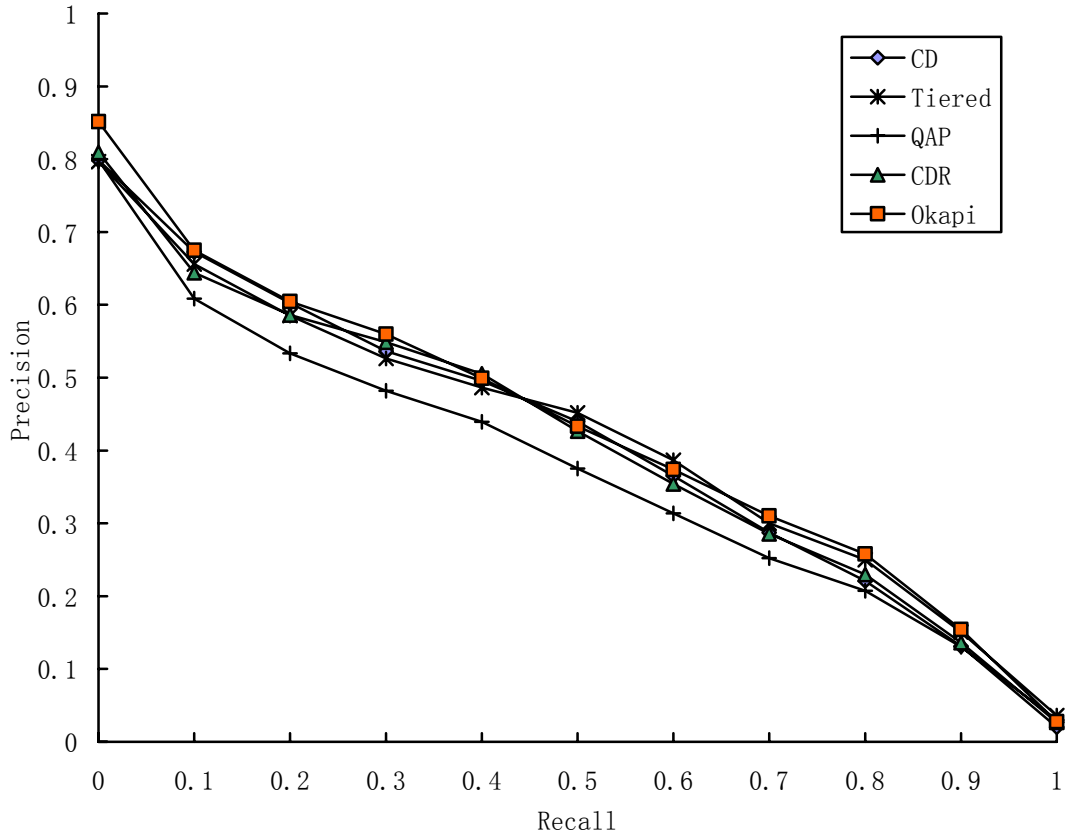
We denote the manual runs on the short query sets as *QAPManualTerms*, ..., *OkapiManualTerms*, *QAPManualBigrams*, ..., *OkapiManualBigrams*. Figure 5.1 and 5.2 show the recall-precision curves for as-is term queries and bigram queries, respectively, where precisions at 11 different recall levels (0.0, 0.1, 0.2, ..., 1.0) are plotted. Table 5.1 and 5.2 give the average precision at document cutoff levels 1, 5, and 20 (denoted by  $p@1$ ,  $p@5$ ,  $p@20$  respectively), the average cover at document cutoff levels 1, 5, 20 ( $c@1$ ,  $c@5$ ,  $c@20$ ), the non-interpolated mean average precision, and the  $p$ -value of the Wilcoxon matched-pairs signed-rank test calculated over the mean average precisions of adjacent retrieval runs. Columns are shown in increasing order of mean average precision values.



**Figure 5.1: Precision-Recall Curves for Short Manual Term Queries for TREC-5 and 6 Topics (Topics 1-54)**

| Measures               | CD          | QAP         | Tiered      | CDR         | Okapi       |
|------------------------|-------------|-------------|-------------|-------------|-------------|
|                        | MaunalTerms | ManualTerms | ManualTerms | ManualTerms | ManualTerms |
| $p@1$                  | 0.6481      | 0.6481      | 0.6481      | 0.6667      | 0.6852      |
| $p@5$                  | 0.6037      | 0.5370      | 0.6074      | 0.6333      | 0.6444      |
| $p@20$                 | 0.5481      | 0.5176      | 0.5528      | 0.5722      | 0.5824      |
| $c@1$                  | 0.6481      | 0.6481      | 0.6481      | 0.6667      | 0.6852      |
| $c@5$                  | 0.9074      | 0.8519      | 0.8889      | 0.9074      | 0.9074      |
| $c@20$                 | 0.9815      | 1.0000      | 0.9815      | 0.9815      | 0.9630      |
| Mean Average Precision | 0.3205      | 0.3231      | 0.3611      | 0.3734      | 0.3945      |
| Intercolumn $p$ -value |             | 0.87        | $<10^{-5}$  | 0.04576     | 0.0396      |

**Table 5.1: Results for Runs Based on Short Manual Term Queries for TREC-5 and 6 Topics (Topics 1)**



**Figure 5.2: Precision-Recall Curves for Short Manual Bigram Queries for TREC-5 and 6 Topics (Topics 1-54)**

| Recall                 | QAP<br>ManualBigrams | CDR<br>ManualBigrams | CD<br>ManualBigrams | Tiered<br>ManualBigrams | Okapi<br>ManualBigrams |
|------------------------|----------------------|----------------------|---------------------|-------------------------|------------------------|
| $p@1$                  | 0.6296               | 0.6296               | 0.6481              | 0.6481                  | 0.7037                 |
| $p@5$                  | 0.5593               | 0.6481               | 0.6370              | 0.6296                  | 0.6630                 |
| $p@20$                 | 0.5333               | 0.5759               | 0.5023              | 0.5778                  | 0.5926                 |
| $c@1$                  | 0.6296               | 0.6296               | 0.6296              | 0.6481                  | 0.7037                 |
| $c@5$                  | 0.8519               | 0.9074               | 0.9074              | 0.9074                  | 0.9630                 |
| $c@20$                 | 1.0000               | 1.0000               | 1.0000              | 1.0000                  | 1.0000                 |
| Mean Average Precision | 0.3555               | 0.4004               | 0.4023              | 0.4058                  | 0.4202                 |
| Intercolumn $p$ -value |                      | $<10^{-4}$           | 0.6115              | 0.2627                  | 0.05934                |

**Table 5.2: Results for Runs Based on Short Manual Bigram Queries for TREC-5 and 6 Topics (Topics 1-54)**

By comparing bigram queries with term queries, we first noticed that, for all ranking algorithms, using overlapping bigrams as queries was overall slightly better than using original terms, especially with CD ranking (up to 18.8% in mean average precision). Similar results were observed by To as well. A possible reason was that, by dividing the terms into bigrams, additional relevant documents containing variations of the original query terms were retrieved. While searching for a Chinese word “ABC”, this word was divided into bigrams “AB” and “BC”. A document in which these two bigrams co-occurred very close or overlapped with each other was likely to be retrieved. Such a document had a good chance of being relevant to the topic.

Second, to compare the five ranking algorithms using the Wilcoxon test over mean average precisions, we observed that for both term and bigram queries, most of the ranking algorithms provided similar performance. QAP were slightly less successful than most others with both term and bigram queries. This indicated that QAP might probably not be very suitable for short queries. Okapi BM25 produced the highest mean average precision for both types of queries, but according to the  $p$ -values, the differences were not sure to be significant at the 95% confidence level. In our experiments, Okapi runs were most comparable to Tiered and CDR for both types of queries.

By looking at the  $p@k$  and  $c@k$  values, it was interesting to notice that, first,  $p@1$  was always the same as  $c@1$ . This is obvious, as for each document, both precision and cover can only receive a score of either 0 or 1, depending on whether the

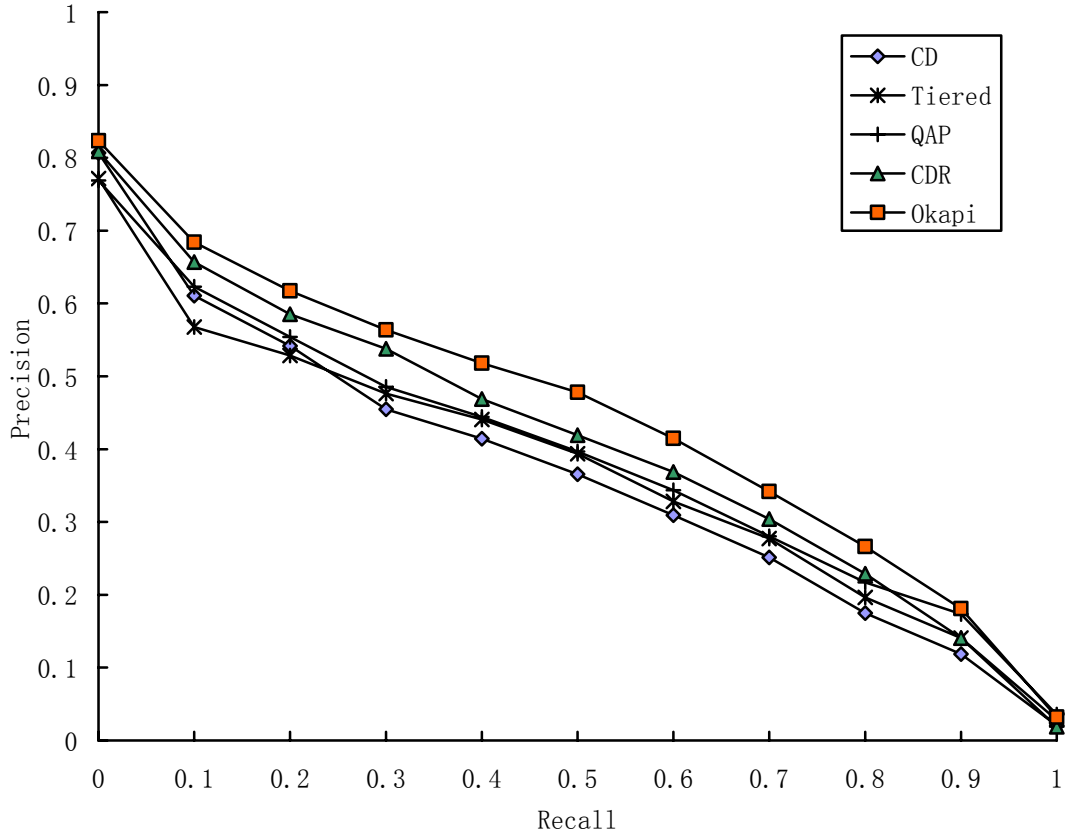
document is relevant. Second,  $c@20$  was always close to 1, which implied that every ranking scheme retrieved at least one relevant document for almost all topics at the document cutoff value 20. Accordingly, in subsequently describing our experiments, it is not necessary to show the results of  $c@1$  and  $c@20$ , as they are not very meaningful in comparing retrieval sets.

### 5.1.2 Runs with Automatic Queries

Another contribution in this report is our experiments with automatically formulated queries that had never been conducted by the MultiText group before. As mentioned in previous sections, the generation of automatic queries involved using different fields or field combinations in each of the 54 topics as resources: “title only” (denoted as T), “description only” (D), and “title + description” (TD). In contrast to English topics, in which the order of length of the fields was  $T < D < TD$ , in many Chinese topics “title” was not necessarily shorter than “description”. For each type of query resource, the three described segmentation methods were applied respectively: Bigrams, BVN, and LDC. Therefore we had ended up with 9 automatic queries for each topic. In addition, the same five ranking algorithms were tested against each query set, which produced 45 runs in total for our experiments in this section.

Okapi BM25 again produced the highest mean average precisions for all of the 9 query sets. Figure 5.3 and Table 5.3 show the results of runs with automatic queries built upon “title+description” and segmented by LDC segmenter (*TD&LDC*). The

Okapi BM25 with TD&LDC queries also suggested the highest mean average precision among the 45 automatic runs.



**Figure 5.3: Precision-Recall Curves for Automatic Queries (TD&LDC) for TREC-5 and 6 Topics (Topics 1-54)**

|                        |  | Automatic queries (TD & LDC) |        |        |         |         |
|------------------------|--|------------------------------|--------|--------|---------|---------|
| Measures               |  | CD                           | Tiered | QAP    | CDR     | Okapi   |
| $p@1$                  |  | 0.5741                       | 0.5926 | 0.5556 | 0.5926  | 0.6481  |
| $p@5$                  |  | 0.6074                       | 0.5407 | 0.5778 | 0.6037  | 0.6778  |
| $p@20$                 |  | 0.5333                       | 0.5111 | 0.5176 | 0.5648  | 0.5991  |
| $c@5$                  |  | 0.9074                       | 0.8333 | 0.9259 | 0.8889  | 0.9444  |
| Mean Average Precision |  | 0.3514                       | 0.3571 | 0.3751 | 0.3983  | 0.4389  |
| Intercolumn $p$ -value |  |                              | 0.4669 | 0.7763 | 0.00297 | 0.00043 |

**Table 5.3: Results for Runs Based on Automatic Query Set TD&LDC and 5 Ranking Algorithms for TREC-5 and 6 Topics (Topics 1-54)**

According to Figure 5.3 and Table 5.3, Okapi BM25 outperformed all other retrieval algorithms with TD&LDC queries. In fact, for all query sets extracted from “title+description”, the differences between Okapi BM25 and other approaches were significant ( $p < 0.001$ , Wilcoxon). This is probably because Okapi accounts for each search term’s query frequency. When combining the “title” and “description” fields, duplicate terms were not eliminated. Those repeated terms that were likely to be weighted higher by Okapi also had a good chance to be more important key words than non-duplicate words.

The other ranking algorithms still provided similar performance. In this particular case CDR is slightly better than CD, Tiered and QAP. But in fact, by analyzing the performance of all these four algorithms applied to other types of queries, we found no common trends regarding which algorithms were consistently better than others. This may be because the automatic queries were mostly much longer than the short manual queries described in the previous section. All these four algorithms are

essentially quite similar in their designs and promise to derive high-performance from short queries. When applied to longer queries, some of them might lose effectiveness, most presumably because the more query terms, the less likely that the terms would appear in the same context in the documents as in the original topic fields.

Table 5.4 shows the results of Okapi BM25 runs for each of the 9 query sets.

|                              |  | Okapi runs for automatic query sets |            |            |               |            |            |                |             |             |
|------------------------------|--|-------------------------------------|------------|------------|---------------|------------|------------|----------------|-------------|-------------|
|                              |  | T &<br>Bigram                       | T &<br>BVN | T &<br>LDC | D &<br>Bigram | D &<br>BVN | D &<br>LDC | TD &<br>Bigram | TD &<br>BVN | TD &<br>LDC |
| Mean<br>Average<br>Precision |  | 0.3369                              | 0.2888     | 0.3490     | 0.3877        | 0.3894     | 0.3846     | 0.4280         | 0.4238      | 0.4389      |
| $p@1$                        |  | 0.5370                              | 0.5185     | 0.6296     | 0.5926        | 0.5741     | 0.6296     | 0.6852         | 0.5926      | 0.6481      |
| $p@5$                        |  | 0.5370                              | 0.5259     | 0.5667     | 0.6111        | 0.5845     | 0.6222     | 0.6407         | 0.6037      | 0.6778      |
| $p@20$                       |  | 0.4861                              | 0.4417     | 0.5157     | 0.5870        | 0.5278     | 0.5324     | 0.5750         | 0.5657      | 0.5991      |
| $c@5$                        |  | 0.8704                              | 0.8519     | 0.8148     | 0.9074        | 0.8148     | 0.9259     | 0.9074         | 0.9074      | 0.9444      |

**Table 5.4: Results for Okapi BM25 Runs Based on 9 Automatic Query Sets for TREC-6 and 6 Topics (Topics 1-54)**

The results shown in Table 5.4 imply that, among the three different fields or field combinations used for query construction, “title+description” (TD) produced the best runs. All the three Okapi runs with TD were comparable with the best short manual runs, such as *OkapiManualBigrams* ( $p>0.2$ , Wilcoxon). “Description only” (D) led to less successful performance, but was better than “title only” (T), as opposed to the typical results in English ad hoc results, where “title only” performed almost as well as “title+description” and they were both better than “description only”. The difference between Chinese and English runs was due to the different topic presentations. In an English topic the title area contains no more than 4 terms, which

are further described by a sentence in the description area. On the contrary, the Chinese description field of each topic consists of a set of keywords, ranging from 2 to 10 terms, whereas the title was written as a sentence, which was harder to segment properly and contained more noisy information, e.g. non-key words and stop words. Most descriptions were longer than their corresponding titles, while some descriptions were shorter. Very short automatic Chinese queries could not be derived from either title or description fields. The retrieval effectiveness was therefore primarily relying on the recognition of key words rather than the query length.

Second, over the three segmentation methods, it was apparent that LDC worked better than other simpler segmentation schemes in the experiments. Those results appeared interestingly opposite to what we discovered in the case of short manual queries. A possible explanation was that, for short queries, the number of overlapping bigrams was small and the system was more able to decide correctly which bi-grams should co-occur or overlap. For long queries, however, bigrams were over-generated without relative position information. The retrieval system was more likely to “misunderstand” the meaning of the query, and therefore the retrieved documents were less likely to be relevant to the topic. Similarly, BVN also generated more noise to hurt the queries than LDC.

### **5.1.3 Runs with Pseudo-Relevance Feedback and Web Reinforcement**

As described before, in the MultiText QA system, the passage retrieval algorithm QAP is used to obtain hotspots with surrounding texts containing a high density of query terms, and the answer extraction heuristics can extract terms by scoring each term in the passage based on the term's passage frequency, inverse document frequency, and distance to the hotspot. In full-text retrieval, the QAP algorithm and the answer extraction heuristic can be adapted for pseudo-relevance feedback [21]. The procedure is as follows:

1. The initial query is submitted to the retrieval system and ranked by the QAP algorithm to retrieve the top  $m$  hotspots;
2. Each non-query term appearing in the hotspots or surrounding texts is viewed as a candidate feedback term and ranked with the weighting scheme similar to the answer extraction heuristics (formula 3.8) used in the QA system.

Specifically, the feedback score for term  $t$  is given by:

$$w_t = \sum_{1 \leq i \leq m} \log\left(\frac{N}{f_t \cdot L(H_i, t)}\right) \quad (5.1)$$

where

$N$  = the corpus size;

$f_t$  = frequency that  $t$  occurs in the corpus;

$H_i$  = the  $i^{\text{th}}$  hotspot,  $1 \leq i \leq m$ ;

$$L(H_i, t) =$$

$$\left\{ \begin{array}{l} l(H_i), \text{ if } t \text{ appears in the hotspot } H_i \\ \text{term positions of the shortest passage containing both } t \text{ and } H_i, \text{ if } t \text{ is outside } H_i \quad ; \\ N/f_t, \text{ if } t \text{ is outside a large window surrounding } H_i \text{ or not in the document containing } H_i \end{array} \right.$$

$l(H_i)$  = the length of  $H_i$  in term positions.

3. The top  $k$  feedback terms are extracted and added to the original query. In order to signal that the feedback terms are less important than the original query terms, the retrieval weight of each feedback term  $t$  is scaled with a scaling factor  $S_t$  as follows:

$$S_t = \frac{C \cdot w_t}{W}; \quad (5.2)$$

where

$C$  = a constant coefficient with a value smaller than 1;

$W$  = the score of the top-ranking feedback term.

4. The scaling factors are used to modify the Okapi BM25 formula 3.6 by adjusting the retrieval weights:

$$\sum_{t \in Q} S_t w^{(1)} q_t \frac{(k_1 + 1)d_t}{K + d_t}; \quad (5.3)$$

This implies that the scaling factors for all original query terms are equal to 1, but for all feedback terms are no more than  $C$ .

5. The expanded query is submitted to retrieve the top 1000 documents with the modified Okapi BM25 measure.

In our experiments,  $m=20$ ,  $k=35$ ,  $C=1/3$ . We selected the best short manual query set *ManualBigrams* as well as the best automatic query set *TD&LDC* as original queries. We used Bigrams instead of LDC words as feedback words, because in the large corpus the limited built-in dictionary used for LDC segmenter resulted in many ambiguous unigrams. Both the TREC Chinese corpus and the Web corpus were used for extracting feedback terms, called *local feedback* and *Web feedback*, respectively. A specific rule with Web feedback was that only the terms appearing in local feedback term lists were considered as feedback candidates. This rule was to ensure that all feedback terms could be found in the TREC corpus.

Table 5.5 shows the impact of pseudo-relevance feedback strategies applied to *ManualBigrams* manual queries and *TD&LDC* automatic queries.

|                        | Okapi for <i>ManualBigrams</i> |                |              | Okapi for <i>TD&amp;LDC</i> |                |              |
|------------------------|--------------------------------|----------------|--------------|-----------------------------|----------------|--------------|
|                        | No Feedback                    | Local Feedback | Web Feedback | No Feedback                 | Local Feedback | Web Feedback |
| $p@1$                  | 0.7037                         | 0.7407         | 0.7037       | 0.6481                      | 0.6481         | 0.6481       |
| $p@5$                  | 0.6630                         | 0.6741         | 0.6556       | 0.6778                      | 0.6704         | 0.6778       |
| $p@20$                 | 0.5926                         | 0.6167         | 0.6028       | 0.5991                      | 0.6037         | 0.6056       |
| $c@5$                  | 0.9630                         | 0.9444         | 0.9259       | 0.9444                      | 0.9444         | 0.9444       |
| Mean Average Precision | 0.4202                         | 0.4347         | 0.4318       | 0.4389                      | 0.4451         | 0.4463       |
| Intercolumn $p$ -value | 0.00017                        | 0.07451        |              | $<10^{-4}$                  | 0.8737         |              |

**Table 5.5: Best Short Manual and Automatic Runs with Local and Web Pseudo-Relevance Feedback and Okapi BM25 (for TREC-5 and 6 Topics 1-54)**

Two conclusions can be drawn from Table 5.5. First, pseudo-relevance feedback was helpful in improving retrieval performance for both manual and automatic queries. Although the improvement of mean average precision was only up to 3.5%, Wilcoxon tests showed that the difference between using feedback or not was significant. Second, there seemed to be not much difference between local and Web feedback. This was probably because the documents in the relatively small TREC corpus were written during 1991 to 1995 and focused on political topics, while the Web data were massive and more up-to-date. Few additional terms found from the Web were added into the candidate term list.

#### **5.1.4 Fusion of Best Runs**

It has been observed by Lee [51] and Fox et al. [23] that the weighting schemes in different types of retrieval algorithms may retrieve different set of documents, and the overall retrieval effectiveness can be improved by fusing the results of multiple runs produced by different retrieval strategies. We used the fusion method described by Lee [51] to blend four of our best runs—the Okapi BM25 local and Web feedback runs using *ManualBigrams* and *TD&LDC* queries, respectively—combining the documents for each topic retrieved in the four runs were combined by taking the intersection of the result sets. Each document score was assigned as the sum of its *normalized scores* in different runs, where, the normalized score of a document in a particular run is given by:

$$\text{Normalized score} = \frac{\text{Original score} - \text{Min. document score for the topic}}{\text{Max. document score for the topic} - \text{Min. document score for the topic}}$$

The fusion of the four runs retrieved 4731 relevant documents for TREC-5 and 6 topics 1-54, with non-interpolated average precision over all relevant documents 0.4838. This improved the previous best results by 8.4%.

In order to compare our results with TREC submissions, we also evaluated the non-interpolated average precision for TREC-6 topics (29-54) only. The result 0.5606 was better than 23 out of the 28 runs submitted for the TREC-6 Chinese track. The five retrieval sets that outperformed our fusion run include: Queens College automatic (mean average precision=0.6263), University of Waterloo manual (0.6203), ETH manual (0.5868) and automatic (0.5733), and CLARITECH manual (0.5797). Interestingly, the best Chinese run at TREC-6 was based on automatic queries (Queens College automatic) instead of manual queries.

## 5.2 Experiments on Question Answering

Our experiments started with the basic version of our Chinese QA system, which included only term and document statistics for both passage retrieval and answer extraction. In the question analysis component, we simply removed stop words and question words like “谁” (who), “何” (what/which), “什么” (what), “哪

个”(what/which), “哪些” (what/which), “几” (how much/many), “多少” (how much/many), etc., and then segmented the remaining part of the question into bag of query words to retrieve passages with the QAP algorithm. There were no question categories or answer patterns defined in the baseline. The answer extraction was simply based on segmenting the top passages and using the answer scoring heuristic (formula 3.8) that accounts for candidate term passage frequency (term redundancy), rarity, and distance to the hotspots. In Section 5.2.1 we examine the effectiveness of those statistics-based heuristics along with different segmentation schemes. The test collection only included the TREC Chinese corpus and the UMass question set. We then augmented our system with question categorization and answer patterns. In Section 5.2.2 we experimented on the impact of adding these natural language processing heuristics. After comparing our system’s performance with Marsha using the same test collection, we extended the evaluation of our system with the new question set and the Web corpus described in Chapter 4.

### **5.2.1 Impact of Different Segmentation Schemes**

Both passage retrieval and answer extraction components required segmentation of Chinese texts. The segmentation schemes to be applied were not necessarily the same for these two phases. For example, we could use bigrams for passage retrieval but LDC words for answer extraction. As we had three implemented segmentation methods at hand: Bigrams, BVN, and LDC, from the simplest to the much more advanced, the combination could end up with six versions of runs. For simplicity, we

only evaluated two ends of the spectrum—Bigrams and LDC—to produce four runs, for the general purpose of investigating whether a simple or a sophisticated segmentation scheme should be better for each component of question answering.

As shown in Section 5.1, bigrams and LDC words were both feasible, although may not be equivalent, for information retrieval. In the answer extraction component, however, since the user would not expect answers always in the form of bigrams, special processing was required when using bigrams for segmentation. We proposed a simple heuristic to address this problem.

Recall the answer extraction formula 3.8:

$$w_t = pf_t \cdot \log\left(\frac{N}{f_t \cdot (\text{loc}(H_i, t) + 1)}\right)$$

When overlapping bigrams were extracted from the retrieved passages and ranked with this formula, the IDF-like weighting component would likely favor non-words, because these bigrams could be rare. This could result in returning nonwords as answers to the user. A way to normalize the scores, was to consider the mutual information  $I(t_1, t_2)$  between the two characters  $t_1$  and  $t_2$  contained in a bigram  $t$  as a cutoff factor:

$$w_t = pf_t \cdot \log\left(\frac{N}{f_t \cdot (\text{loc}(H_i, t) + 1)}\right) \cdot G \quad (5.4)$$

where

$$G = \begin{cases} I(t_1, t_2), & \text{if } t \text{ is a bigram} \\ 7, & \text{(the threshold of } I), \text{ if } t \text{ contains more than 2 characters} \end{cases}$$

With formula 5.4, a highly ranked bigram has better likelihood of being a real word. If two or more top bigrams were extracted from the same passage, and these bigrams overlapped, they were concatenated with the overlap eliminated. Each retrieved passage was examined until all such n-grams ( $n \geq 2$ ) were obtained. The term scores of those new n-grams were calculated and all extracted terms were ranked to be voted for the final answers.

To evaluate the basic versions of our system, the main measures we used were *mean reciprocal rank* (MRR) and *accuracy*. QA at TREC originally used MRR to evaluate systems that returned 5 answer passages for each question, and later on used accuracy when only one answer was allowed for each question. Our systems looked for exact answers, but for a more in-depth investigation we had a ranked list of 5 answers returned for each question. Accordingly we used MRR to evaluate all 5 answers, while used accuracy to only examine the top answer.

Table 5.6 shows the results of the basic system using the combinations of Bigrams and LDC segmentation schemes, where in addition to MRR and Accuracy, it gives the “#incorrect”, the number of questions that suggested no correct answers in any of the five returns, as well as the “%correct”, or “ $c@5$ ”, which is the fraction of questions that suggested at least one correct answer among the five returns. The four runs were denoted by *Bigrams-Bigrams* (Bigrams for passage retrieval, Bigrams for answer extraction), *LDC-Bigrams* (LDC for passage retrieval, Bigrams for answer

extraction), *Bigrams-LDC*, and *LDC-LDC*. The test collection, as mentioned before, contained only the TREC Chinese corpus and the 51 UMass questions.

| Basic Runs      | %Correct<br>( $c@5$ ) | MRR   | Accuracy | #Incorrect |
|-----------------|-----------------------|-------|----------|------------|
| Bigrams-Bigrams | 29.41                 | 0.259 | 0.235    | 36         |
| LDC-Bigrams     | 25.49                 | 0.232 | 0.216    | 38         |
| Bigrams-LDC     | 33.33                 | 0.320 | 0.314    | 34         |
| LDC-LDC         | 33.33                 | 0.333 | 0.333    | 34         |

**Table 5.6: Running Results of Basic QA System with 4 Segmentation Combinations**

Apparently, runs using LDC for answer extraction produced much better results than runs with Bigrams for the same component. When using bigrams for answer extraction, the heuristic formula 5.4 had successfully formulated several correct answers longer than bigrams, such as “曹雪芹” (Cao, Xue-qin) from “曹雪” and “雪芹” for question 36 “红楼梦的作者是谁?” (Who is the author of *the Dream of the Red Chamber?*), however, it increased the complexity of the weighting scheme and required more time on processing terms, and its power was even limited when answers became more rare and complicated. Therefore, LDC was a better choice for answer extraction. As for passage retrieval, Bigrams and LDC had close performance. In order to allow for simplicity of adding more natural language processing on top of the basic system, such as more sophisticated question analysis, correctly segmented words were better suited. Thus we kept the basic LDC-LDC version as the baseline of our QA system.

### 5.2.2 Impact of Using Answer Patterns

To find more accurate answers, we augmented the LDC-LDC baseline with question classification and pattern matching. When a specific category was assigned to a question, the answer extraction component took advantage of this information by restricting the candidate terms to instances matching the patterns corresponding to the designated category. As mentioned in Chapter 3, we defined seven question categories, similar to the Marsha system. The heuristics for determining instances for PERSON, NUMBER, DATE and TIME were described in Chapter 3. For LOCATION and ORGANIZATION types, the candidates were lists of terms appearing in the LOCATION and ORGANIZATION names, respectively. It was difficult to include all instances in these categories, but with the aid of the statistics-based heuristic (formula 3.8), there was still a good chance to locate the correct answer even though it was not included as an instance of the category it belonged to.

Using the same test collection as in Section 5.2.1, the result of running the augmented system is shown in Table 5.7:

| Runs                  | # Correct          | MRR                | Accuracy          | # Incorrect    |
|-----------------------|--------------------|--------------------|-------------------|----------------|
| Baseline<br>(LDC-LDC) | 33.33              | 0.326              | 0.314             | 34             |
| Augmented             | 70.59<br>(+211.8%) | 0.660<br>(+102.5%) | 0.627<br>(+99.7%) | 15<br>(-55.9%) |

**Table 5.7: QA Result with Question Classification and Pattern Matching**

Table 5.7 shows a significant improvement of our system by the use of question categories and answer patterns. This indicated that natural language processing was

of great importance for QA in the Chinese language. A typical example was question 15 “卢沟桥上有多少个石狮子?” (How many stone lions are there on the Lu-Gou Bridge?). The correct answer—“四百八十五个” (where “四百八十五” is composed by numerals and means 485; “个” is just a unit word to fit in the context)—was in the NUMBER category, was impossible to be found without defining the answer patterns, because neither statistic heuristics nor a dictionary could locate such a character sequence as a word. However, with answer patterns, it was found and ranked highest in the answer list. The other answers in the top five, as shown below also matched the NUMBER category:

- 2). 七七 (77)
- 3). 451
- 4). 一个 (one)
- 5). 2

Among the 51 questions, six belonged to the NUMBER category. None of them suggested correct answers in the baseline system. The augmented version returned correct answers at top rank for three of the questions, and at rank 5 for another one. The impact of using our name recognizer for the PERSON category was even significant. The baseline system only answered five of the 14 questions correctly in the PERSON category, while our augmented version returned correct answers for all of them, 13 of which were at the top rank and the remaining one was at the second rank. Both systems worked well on LOCATION, ORGANIZATION, and OTHER questions, which might imply that system performance on these types of questions

was relying on the correct segmentation and the statistical answer selection heuristics more than pre-defined answer patterns. Both our baseline and augmented systems lost points for DATE and TIME questions. We defined answer patterns similar to the NUMBER types, except that we restricted the set of unit words and the particular formats for expressing time and date. For instance, the only unit words in the DATE category were “年”(year), “月”(month), “日”(date), “星期”(week), “礼拜”(week), “公元”(A.D.), “公元前”(B.C.). The most common format in expressing a date is “x年 y月 z日”. The reason that our augmented system failed to return correct answers for many of these questions, was both because of the complexity in expressing time and dates, and because in reality, time and dates are subject to change and are likely to be expressed relative to a historical time or date. This was also observed by Li and Croft [73]. An example they gave was the question “谢军在哪一年战胜了前苏联选手第一次获得国际象棋世界冠军” (In which year did Jun Xie defeat a Russian player and win the world chess championship for the first time?). The answer we produced was the same as Marsha: “今天” (today). This incorrect answer actually referred to the date of October 29, 1991 in the context of the supporting document.

### **5.2.3 Comparison with the Marsha Chinese QA System**

Since our QA system used the same document and question collections as Marsha did, we could compare the performance of the two systems.

Marsha answered 24 questions correctly. As it only returned one answer for each question, the accuracy was equal to MRR: 0.47. It might have the potential to answer more questions correctly if it suggested 5 answers per question, thus it is hard to compare the MRR and  $c@5$  between our system and Marsha. But by only looking at the top returns, our system produced 32 correct answers with accuracy 0.627, which significantly improved Marsha's results by 33.4%. The reasons, as analyzed in Chapter 3, mainly lay in the effective passage retrieval algorithm and answer selection heuristics. The answer patterns we defined were also helpful in extracting answers that were unable to be located by a named entity markup tool such as *IdentiFinder*.

#### **5.2.4 Evaluation with New Questions and the Web Corpus**

Compared to the real QA tracks in English, the test collection used for our Chinese system was relatively too limited. The document corpus was only 170MB in size, and the UMass question set contained only 51 questions. Even though our system answered about 70% of questions correctly, we were uncertain that the system could scale up for real applications. Accordingly, we extended our evaluations with a secondary document collection—the 17GB Web corpus, as well as a larger question set—the 149 new questions as described in Chapter 3.

The experiments were conducted as follows: All runs used the new questions. The first run was still based on the TREC corpus only; the second run used the Web to reinforce the selection of candidates extracted from the TREC corpus; and the last run

used the Web collection only for passage retrieval and answer extraction. The results are given in Table 5.8:

| Runs with new questions        | % Correct (c@5) | MRR   | Accuracy | # Incorrect |
|--------------------------------|-----------------|-------|----------|-------------|
| TREC Corpus                    | 56.38           | 0.519 | 0.497    | 65          |
| TREC Corpus +Web Reinforcement | 57.72           | 0.516 | 0.483    | 63          |
| Web only                       | 53.02           | 0.484 | 0.463    | 70          |

**Table 5.8: Evaluation of Chinese QA with New Questions and Web Corpus**

Our system seemed still effective in general when new questions were tested. In fact, using the Wilcoxon test on the reciprocal ranks, the differences were not significant between any pair of the runs ( $p > 0.3$ ). However, the significance test does not fully characterize the systems' performance. Compared to the run with TREC corpus only, Web reinforcement could influence the answer selection component only by changing the term passage frequency (term redundancy factor). In our experiments, the Web data affected the system by having fewer of the new questions answered correctly at the top rank, while acquiring more correct answers between rank 2 and 5. However, overall the use of Web did not produce gains in the performance. The problem might be explained by the quality of the Web data. As mentioned in Section 5.1, there was a mismatch about the age and styles between the TREC and Web data. For example, for question “中国国家主席是谁?” (Who is the president of China?), in the TREC corpus, where the articles were written between 1991 and 1995, the answer should be “杨尚昆” (Yang Shang-Kun) or “江泽

民”(Jiang Ze-Min). However, the Web data would suggest the current president’s name: “胡锦涛” (Hu Jin-Tao). We treated all these answers to be correct. Runs with TREC data only and TREC with Web reinforcement would only return “杨尚昆” (Yang Shang-Kun) or “江泽民”(Jiang Ze-Min) to the user, since only terms appearing in the TREC corpus were allowed to be voted as answer candidates. Therefore, in the run with Web reinforcement, the Web data would only degrade the scores of the candidates found in TREC Corpus, which might result in none of the desired answers being returned to the user. Another problem regarding the Web data was that most pages we retrieved were from commercial sites, and thus the advertisements took up a considerable fraction of the corpus and produced a lot of noise. In most cases, TREC passages were ranked ahead of Web passages. This might also explain the reason that the “Web only” run did not produce a gain either.

## **Chapter 6**

### **Conclusion and Future Work**

Typical modern QA systems employ NLP strategies on top of IR. In the course of developing a Chinese QA system with MultiText, there arose two questions: Are the IR techniques developed by MultiText suitable for retrieving information from Chinese texts? What specific NLP problems need to be addressed for Chinese QA?

To answer these questions, first of all there was a need to re-visit a traditional Chinese IR task—full-text retrieval. During MultiText’s participation in the TREC-6 Chinese track, the character-based indexing and the use of the Shortest Substring Ranking applied to structured long manual queries produced the best manual submission. However, retrieval with automatic queries and pseudo-relevance feedback was not investigated. Chinese NLP issues, such as automatic word segmentation, were therefore also left unexplored. In the full-text retrieval

experiments described in this thesis, we used both short term and bigram queries that were constructed by To[67] as well as a set of automatic queries that were extracted from “title”, “description”, or “title+description” segmented with three different schemes: Bigrams, BVN, and LDC, from the simplest method to the most advanced. The document ranking algorithms we examined included both traditional and new passage-based strategies, including CD, Tiered, QAP and CDR, and a variant of a well-known probabilistic document retrieval method Okapi BM25.

In general, most retrieval techniques were also shown appropriate for Chinese. The overall comparison among the five different ranking techniques indicated the same trends of retrieval effectiveness in Chinese as in English: most techniques produced comparable performance, while Okapi BM25 worked slightly better. In particular, for short manual queries, using overlapping bigrams were better than using as-is terms. QAP performed less successfully than other algorithms, probably because the terms’ within-document frequency was not considered. For automatic queries that were usually longer, however, it was hard to identify the best passage-based algorithms, as most of them were initially designed to achieve high-performance for very short queries. When a query contains more terms, there is a less likelihood that the terms would appear in the same context in the documents as in the original topic fields. To compare three segmentation schemes, we noticed that LDC outperformed Bigrams and BVN, which might imply that more accurate segmentation could be beneficial for longer queries. To compare the use of different topic fields for query formulation, “title+description”, which resulted in the longest queries with duplicate search terms,

worked the best. “Description only” queries were in general a little bit longer than “title only” ones, but not necessarily true for some of the topics. Because the description area contained a set of keywords while the topic area was a sentence or phrase that was harder to segment and contained more noise, “description only” queries produced consistently better results than “title only” ones, as opposed to the results in English ad hoc retrieval. Further improvement to our results was made using pseudo-relevance feedback adapted from the QAP algorithm and the QA term selection heuristics to identify feedback terms and weights for query expansion. The fusion of several best runs produced significant gains and was competitive with most of the TREC-6 Chinese track submissions.

When stepping into the actual building of a Chinese QA system, we incorporated the pipeline architecture similar to the MultiText English QA system. The MultiText QAP and term extraction heuristics were both used in our system and proved effective in our experiments. Among the different segmentation schemes, LDC was more suitable for the QA task as more NLP was required than in full-text retrieval. The heuristics designed specifically for Chinese question categorization and pattern matching, such as the name and number recognizers considerably improved the system performance.

Both full-text retrieval and question answering experiments involved the use of the Web corpus. However, the Web data did not produce desired gains in the performance of our systems. The most probable reasons were that the Web data were collected from commercial sites whose quality might not be high, and most Web

content was about 10 years newer than the newswire articles in the TREC corpus, which led to a mismatch between the two collections.

For future work, for text retrieval, we need to investigate in depth the relationship between query length and the effectiveness of each ranking and segmentation algorithm. This research might be the key to find out the factors related to retrieval effectiveness and thus the best representation of Chinese queries. For QA, more efforts are necessary in question categorization and pattern matching. The classification of our current question categories is too general. In TREC QA, it has been observed that there was a trade-off between the use of a few very broad categories and the use of many specialized categories [31]. We may consider using a hierarchical typology to exploit this trade-off in the future. The current answer patterns we designed are far from complete for real QA applications. For example, the recognition of time and date was not successful. In future we may consider combining the BBN *IdentiFinder* with our own pattern recognizers to help identify more answer patterns properly. Furthermore, our current QA system could only solve open-domain, factoid questions. Techniques dealing with temporal, domain-specific, list and definitional questions can be exploited in the future. We also plan to incorporate existing techniques for both English and Chinese QA systems into a cross-lingual QA system in which the user can input questions in one language while obtaining the answer in the other. In addition, a standard test collection with high quality of question sets and document corpus need to be built as well.

# Bibliography

- [1]A.Chen, J.He, L.Xu, F.C.Gey, and J.Meggs. Chinese Text Retrieval Without Using a Dictionary. In *Proceedings of the 20th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 42-49. 1997. Philadelphia, PA.
- [2]A.Singhal, S.Abney, M.Bacchiani, M.Collins, D.Hindle, and F.Pereira. AT&T at TREC-8. In *The 8th Text REtrieval Conference*. 1999. Gaithersburg, MD.
- [3]A.Smeaton and R.Wilkinson. Spanish and Chinese Document Retrieval in TREC-5. In *The 5th Text REtrieval Conference*. 1997. Gaithersburg, MD.
- [4]C.Buckley. trec\_eval IR Evaluation Package. Available from <ftp://ftp.cs.cornell.edu/pub/smart>.
- [5]C.Buckley, A.Singhal, and M.Mitra. New Retrieval Approaches Using SMART: TREC 4. In *The 4th Text REtrieval Conference*. 1995. Gaithersburg, MD.
- [6]C.Buckley, M.Mitra, J.Walz, and C.Cardie. Using Clustering and SuperConcepts Within SMART: TREC 6. In *The 6th Text REtrieval Conference*. 1997. Gaithersburg, MD.

- [7]C.Fellbaum, e. *WordNet: An Electronic Lexical Database*. 1998. The MIT Press.
- [8]C.Kwok, O.Etzioni, and D.S.Weld. Scaling Question Answering to the Web. In *Proceedings of the 10th International World Wide Web Conference (WWW 10)*. 2001.
- [9]C.L.A.Clarke and G.V.Cormack. Interactive Substring Retrieval (MultiText Experiments for TREC-5). In *The 5th Text Retrieval Conference (TREC-5)*, pages 267-278. 1996. Gaithersburg, MD.
- [10]C.L.A.Clarke, G.V.Cormack, D.I.E.Kisman, and T.R.Lynam. Question Answering by Passage Selection (MultiText Experiments for TREC-9). In *The 9th Text REtrieval Conference (TREC 9)*. 2000. Gaithersburg, MD.
- [11]C.L.A.Clarke, G.V.Cormack, and E.A.Tudhope. Relevance Ranking for One to Three Term Queries. *Information Processing and Management*, 36(2), pages 291-311. 2000.
- [12]C.L.A.Clarke, G.V.Cormack, and F.J Burkowski. An Algebra for Structured Text Search and a Framework for Its Implementation. *The Computer Journal*, 38(1), Pages 43-56. 1995.
- [13]C.L.A.Clarke, G.V.Cormack, and F.J Burkowski. Shortest Substring Ranking (MultiText Experiments for TREC-4). In *4th Text REtrieval Conference (TREC-4)*, Pages 295-304. 1995. Washington, DC.

- [14]C.L.A.Clarke, G.V.Cormack, G.Kemkes, M.Laszlo, T.R.Lynam, E.L.Terra, and P.L.Tilker. Statistical Selection of Exact Answers (MultiText Experiments for TREC 2002). In *The 11th Text REtrieval Conference (TREC 2002)*. 2002. Gaithersburg, MD.
- [15]C.L.A.Clarke, G.V.Cormack, and T.R.Lynam. Exploiting Redundancy in Question Answering. In *24th Annual ACM SIGIR Conference on Research and Development in Information Retrieval*. 2001. New Orleans.
- [16]C.L.A.Clarke, G.V.Cormack, T.R.Lynam, C.M.Li, and G.L.McLean. Web Reinforced Question Answering (MultiText Experiments for TREC 2001). In *The 10th Text REtrieval Conference (TREC 2001)*. 2001. Gaithersburg, MD.
- [17]D.Harman. Overview of The First Text REtrieval Conference (TREC-1). In *1st Text REtrieval Conference (TREC-1)*. 1992. Gaithersburg, MD.
- [18]D.Harman. Data Preparation. In *the Proceedings of the TIPSTER Text Program--Phase I*. 1994. San Mateo, CA, Morgan Kaufmann.
- [19]D.Harman. Overview of the Second Text REtrieval Conference (TREC-2). In *the 2nd Text REtrieval Conference*. 1994. Gaithersburg, MD.
- [20]D.Knaus, E.Mttendorf, P.Schauble, and P.Sheridan. Highlighting Relevant Passages for Users of the Interactive SPIDER Retrieval System. In *4th Text Retrieval Conference (TREC-4)*, Pages 233-243. 1995. Washington.

- [21]D.L.Yeung, C.L.A.Clarke, G.V.Cormack, T.R.Lynam, and E.L.Terra. Task-Specific Query Expansion (MultiText Experiments for TREC 2003). In *The 12th Text REtrieval Conference (TREC 2003)*. 2003. Gaithersburg, MD.
- [22]D.Moldovan, S.Harabagiu, R.Girju, P.Morarescu, F.Lacatusu, A.Novischi, A.Badulescu, and O.Bolohan. LCC Tools for Question Answering. In *The 11th Text Retrieval Conference (TREC 2002)*. 2002. Gaithersburg, MD.
- [23]E.A.Fox, M.P.Koushik, J.Shaw, R.Modlin, and D.Rao. Combining Evidence from Multiple Searches. In *the 1st Text REtrieval Conference (TREC-1)*, pages 319-328. 1993. Gaithersburg, MD.
- [24]E.Brill, J.Lin, M.Banko, S.Dumais, and A.Ng. Data-Intensive Question Answering. In *The 10th Text REtrieval Conference (TREC 2001)*. 2001. Gaithersburg, MD.
- [25]E.Hovy, U.Hermjakob, and C.Lin. The Use of External Knowledge in Factoid QA. In *The 10th Text REtrieval Conference*. 2001. Gaithersburg, MD.
- [26]E.Mttendorf and P.Schauble. Document and Passage Retrieval Based on Hidden Markov Models. In *Proceedings of the 17th Annual International ACM Conference on Research and Development in Information Retrieval*, Pages 318-327. 1994. Dublin, Ireland.
- [27]E.Voorhees. The TREC-8 Question Answering Report. In *The 8th Text REtrieval Conference*. 1999. Gaithersburg, MD.

[28]E.Voorhees. The TREC-8 Question Answering Track Evaluation. In *The 8th Text REtrieval Conference*. 1999. Gaithersburg, MD.

[29]E.Voorhees. The TREC-8 question answering track report. In *8th Text REtrieval Conference*. 1999. Gaithersburg, MD.

[30]E.Voorhees. Overview of the TREC-9 Question Answering Track. In *The 9th Text REtrieval Conference*. 2000. Gaithersburg, MD.

[31]E.Voorhees. Overview of the TREC2001 Question Answering Track. In *The 10th Text REtrieval Conference*. 2001. Gaithersburg, MD.

[32]E.Voorhees. Overview of the TREC 2002 Question Answering Track. In *The 11th Text REtrieval Conference*. 2002. Gaithersburg, MD.

[33]E.Voorhees. Overview of the TREC 2003 Question Answering Track (Draft). In *The 12th Text REtrieval Conference*. 2003. Gaithersburg, MD.

[34]E.Voorhees. Overview of TREC 2003 (Draft). In *the 12th Text REtrieval Conference*. 2003. Gaithersburg, MD.

[35]E.Voorhees and D.Harman. Overview of the Sixth Text Retrieval Conference (TREC-6). In *The 6th Text REtrieval Conference*. 1997. Gaithersburg, MD.

[36]E.Voorhees and D.Harman. Overview of the Eighth Text Retrieval Conference (TREC-8). In *The 8th Text REtrieval Conference*. 1999. Gaithersburg, MD.

- [37]E.Voorhees and D.Harman. Overview of the Eighth Text Retrieval Conference (TREC-8). In *The 8th Text REtrieval Conference*. 1999. Gaithersburg, MD.
- [38]E.Voorhees and D.Harman. Overview of the Ninth Text Retrieval Conference (TREC-9). In *9th Text REtrieval Conference*. 2000. Gaithersburg, MD.
- [39]E.Voorhees and D.M.Tice. Building a Question Answering Test Collection. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 200-207. 2000.
- [40]G.Salton. Automatic Information Organization and Retrieval. 1968. McGraw-Hill.
- [41]G.Salton. *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. 1989. MA, Addison-Wesley Series in Computer Science. Addison-Wesley Longman Publ. Co., Inc., Reading .
- [42]G.Salton and C.Buckley. Automatic Text Structuring and Retrieval-- Experiments in Automatic Encyclopedia Searching. In *Proceedings of the 14th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Pages 21-30. 1991. Chicago, IL.
- [43]G.Salton, J.Allan, and C.Buckley. Approaches to passage Retrieval in Full Text Information Systems. In *Proceedings of ACM-SIGIR International Conference on Research and Development in Information Retrieval* , Pages 49-58. 1993. Pittsburg.

- [44]G.Salton, J.Allan, and C.Buckley. Approaches to passage Retrieval in Full Text Information Systems. In *Proceedings of ACM-SIGIR International Conference on Research and Development in Information Retrieval* , Pages 49-58. 1993. Pittsburg.
- [45]G.V.Cormack, C.L.A.Clarke, C.R.Palmer, and D.I.E.Kisman. Fast Automatic Passage Ranking (MultiText Experiments for TREC-8). In *The 8th Text REtrieval Conference (TREC 8)*. 1999. Gaithersburg, MD.
- [46]G.V.Cormack, C.L.A.Clarke, C.R.Palmer, and S.S.L.To. Passage-based Query Refinement. In *6th Text REtrieval Conference (TREC-6)*, Pages 303-319. 1997. Gaithersburg, MD.
- [47]G.V.Cormack, C.R.Palmer, M.Biesbrouck, and C.L.A.Clarke. Deriving Very Short Queries for High Precision and Recall. In *the 7th Text REtrieval Conference (TREC-7)*. 1998. Gaithersburg, MD.
- [48]I.H.Witten, A.Moffat, and T.C.Bell. *Managing Gigabytes Compression and Indexing Documents and Images*. 1999. Morgan Kaufmann.
- [49]J.Allan, L.Ballesteros, J.P.Callan, W.B.Croft, and Z.Lu. Recent Experiments with INQUERY. In *The 4th Text REtrieval Conference*. 1995. Gaithersburg, MD.
- [50]J.Chu-Carroll, J.Prager, C.Welty, K.Czuba, and D.Ferrucci. A Multi-Strategy and Multi-Source Approach to Question Answering. In *The 11th Text REtrieval Conference (TREC 2002)*. 2002. Gaithersburg, MD.

[51]J.H.Lee. Combining Multiple Evidence from Different Properties of Weighting Schemes. In *Proceedings of the 18th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 180-188. 1995. Seattle, WA.

[52]J.P.Callan. Passage-level Evidence in Document Retrieval. In *Proceedings of ACM-SIGIR International Conference on Research and Development in Information Retrieval*, Pages 302-309. 1994. Dublin, Ireland.

[53]J.Prager, E.Brown, A.Coden, and D.Radev. Question-Answering by Predictive Annotation. In *23th Annual ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49-58. 2000. Athens.

[54]J.Zobel and A.Moffat. Exploring the similarity space. *SIGIR Forum* 32, Pages 18-34. 1998.

[55]J.Zobel, A.Moffat, and R.Wilkinson. Efficient Retrieval of Partial Documents. *Information Processing & Management*, 31(3): 361-377. 1995.

[56]K.S.Jones and C.V.Rijsbergen. Report on the Need for and Provision of an "Ideal" Information Retrieval Test Collection. British Library Research and Development Report 5266. 1975. Computer Laboratory, University of Cambridge.

[57]M.A.Hearst and C.Plaunt. Subtopic Structuring for Full-length Document Access. In *Proceedings of ACM-SIGIR International Conference on Research and Development in Information Retrieval*, Pages 59-68. 1993. Pittsburg.

- [58]M.Kaszkiel and J.Zobel. Passage Retrieval Revisited. In *Proceedings of the 20th Annual International ACM Conference on Research and Development in Information Retrieval*, Pages 178-185. 1997. Philadelphia, PA.
- [59]M.Najork and J.L.Wiener. Breath-First Search Crawling Yields High-Quality Pages. In 10th International World Wide Web Conference, pages 114-118. 2001.
- [60]R.Srihari and Z.Lu. Information Extraction Supported Question Answering. In *The 8th Text REtrieval Conference*. 1999. Gaithersburg, MD.
- [61]R.Wilkinson. Effective Retrieval of Structured Documents. In *Proceedings of ACM-SIGIR International Conference on Research and Development in Information Retrieval* , Pages 311-317. 1994. Dublin, Ireland.
- [62]R.Wilkinson. Chinese Document Retrieval at TREC-6. In *The 6th Text REtrieval Conference*. 1998. Gaithersburg, MD.
- [63]S.Abney, M.Collins, and A.Singhal. Answer Extraction. [In *Conference on Applied Natural Language Processing*]. 2000.
- [64]S.Harabagiu, D.Moldovan, M.Pasca, R.Mihalcea, M.Surdeanu, R.Bunescu, R.Girju, V.Rus, and P.Morarescu. The Role of Lexico-Semantic Feedback in Open-Domain Textual Question-Answering. In *Proceedings of the Association for Computational Linguistics*, pages 274-281. 2001.
- [65]S.Robertson and S.Walker. Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval. In Proceedings of the 17th

Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, Pages 232-241. 1994. Dublin, Ireland.

[66]S.Robertson, S.Walker, S.Jones, M.M.Hancock-Beaulieu, and M.Gatford. Okapi at TREC-3. In *The 3rd Text REtrieval Conference (TREC-3)*, Pages 109-126. 1994. Gaithersburg, MD.

[67]S.S.L.To. Passage-Based Chinese Text Retrieval. Master Thesis. 1998. Department of Computer Science, University of Waterloo.

[68]S.Sproat and C.L.Shih. A Statistical Method for Finding Word Boundaries in Chinese Text. *Computer Processing of Chinese and Oriental Languages*, 4(4), pages 336-351. 1990.

[69]S.Tellex, B.Katz, J.Lin, A.Fernandes, and G.Marton. Quantitative Evaluation of Passage Retrieval Algorithms for Question Answering. In Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Pages 41-47. 2003. Toronto, Canada.

[70]S.Walker, S.Robertson, M.Boughanem, G.Jones, and K.S.Jones. Okapi at TREC-6 Automatic Ad Hoc, VLC, Routing, Filtering and QSDR. In *the 6th Text REtrieval Conference (TREC-6)*, Pages 125-136. 1997. Gaithersburg, MD.

[71]W.B.Frakes and R.Baeza-Yates. *Information Retrieval: Data Structures and Algorithms*. 1992. Upper Saddle River, NJ, Prentice-Hall Inc.

[72]X.Huang and S.Robertson. Okapi Chinese Text Retrieval Experiments at TREC-6. In *The 6th Text REtrieval Conference (TREC-6)*. 1997. Gaithersburg, MD.

[73]X.Li and W.B.Croft. Evaluating Question Answering Techniques in Chinese. In *Proceeding of HLT 2001*. 2001. San Diego, CA.

[74]Z.Wu and G.Tseng. Chinese Text Segmentation for Text Retrieval: Achievements and Problems. *Journal of the American Society for Information Science*, 44, pages 532-542. 1993.