Some Experiments with Blind Feedback and Re-ranking for Chinese Information Retrieval

Y.XIAO, R.W.P. LUK
Dept. of Computing
The Hong Kong Polytechnic University
Email:
{csyxiao,csrluk}@comp.polyu.edu.hk

K.F. WONG
Dept. Systems Engineering & Engineering Management
Chinese University of Hong Kong
Email:
kfwong@se.cuhk.edu.hk

K.L. KWOK
Dept. of Computer Science
City University of New York
Email:
kwok@cs.qc.edu

Abstract

In our formal runs, we have experimented with hybrid-term indexing and bigram indexing because hybrid-term indexing is a more distinct type of indexing strategy for better pooling and because bigram indexing usually gives robust (near) good results. We have also used our pseudo-relevance feedback (PRF) methods. In the informal runs, we have experimented with our previous re-ranking strategy, called title re-ranking. This strategy rewards documents which title terms match with the terms in the title query. Title re-ranking is able to improve the effectiveness performance for both short and long queries when bigram indexing is used. For formal runs, our best relax MAP achieved was 36% and 51% using PRF, for title queries and long queries respectively. For informal runs, our best relax MAP achieved was 43% for title queries and 50% for long queries using both PRF and merging retrieval lists.

Keywords: *Chinese information retrieval, indexing, 2-Poisson model, relevance feedback, re-ranking and evaluation.*

1 Introduction

In this year NTCIR-5 participation, our submitted runs are based on bigram indexing and hybrid-term indexing for the single language (Chinese) information retrieval task. We did not use character-based indexing methods in this year's submission because we confirmed that the retrieval effectiveness of character indexing was disappointing in NTCIR-4 against the effectiveness of other indexing strategies.

Pseudo-relevance feedback (PRF) or blind feedback is one of the most well known [1] and widely applied techniques in the open IR evaluation workshops for improving the retrieval effectiveness. Here, we used our previous best PRF method in [1]. We also used our title re-ranking technique in the informal runs to examine its impact on retrieval effectiveness.

The rest of the paper is organized as follows. Section 2 discusses our formal runs as well as other informal runs. Section 3 fine tunes existing methods. Section 4 is our simulated experiments for getting some ideas about how much improvement can be expected if better retrieval techniques can be used. Section 5 is the efficiency of our system. Finally, section 6 summarizes our findings.

2 Standard Runs

In this section, we report our formal runs and the runs using our previous PRF method based on the settings of the formal runs.

2.1 Formal Runs

We used the 2-Poisson model with the Okapi BM11' weighting function [2] as follows:

where q is the query, d_i is the i-th document, q_j is the

$$BM11'(q,d_i) \equiv \sum_j q_j \log\left(\frac{N-n_j+0.5}{n_j+0.5}\right)\left(\frac{t_{i,j}}{t_{i,j}+\frac{len_i}{len}}\right)$$

j-th query term weight, N is the number of documents in the collection, n_j is the document frequency of the j-th term, $t_{i,j}$ is the j-th term frequency in the i-th document and len_i is the the Euclidean document length for the i-th document and len is the average Eucliden document length.

From previous indexing work [3, 4, 5], it is clear that words are the preferred index terms if there is no out-of-vocabulary problem. To solve the out-of-vocabulary problem, words can be extracted automatically [6, 7] but there are concerns about the recall performance of automatic extractions or the concerns about the scope of word formation rules [8]. Instead, we propose to use bigrams to solve the out-

of-vocabulary problem. Bigrams have the advantage that it is a completely data-driven technique, without any rule maintenance problem. Bigrams can be extracted on the fly for each document. There are no requirements to define a somewhat arbitrary threshold (or support) and there is no need to extract and test any templates for word extraction. Therefore, we proposed the hybrid term indexing strategy as in Algorithm A.

Input: Document *d* and the word dictionary *D*
Output: Index terms $\{ w \} \cup \{ b \}$
Method: Hybrid Term Indexing
Step 1 Segment text into sequences s_k
Step 2 **For each** sequence s_k of Chinese characters in the document ***d*** **do**
Step 3 Segment s_k using the word dictionary *D*
Step 4 **For each** word $w \in D$ matched in s_k **do**
Step 5 **if** $|w| > 1$ character **and** *w* is not a stop word **then**
Step 6 Index *w*
Step 7 **end**
Step 8 **For each** single-character segmented substring $s_{k,m}$ in s_k **do**
Step 9 **if** $|s_{k,m}| > 1$ character **then**
Step 10 **For each** bigram *b* in $s_{k,m}$ **do**
Step 11 Index *b*
Step 12 **end**
Step 13 **else**
Step 14 **if** $s_{k,m}$ is not a stop word **then**
Step 15 Index $s_{k,m}$ as a word $w \in D$
Step 16 **end**
Step 17 **end**
Algorithm A. Hybrid term indexing.

Algorithm A combined both word-based indexing and bigram-based indexing. Note that Algorithm A does not index single-character words unless the single-character segmented substring is a single character and it is not a stop word. To secure better recall instead of precision, Algorithm A can be changed to index all single-character words that are not stop words. In this case, step 5 of Algorithm A is modified to:

if *w* is not a stop word **then**,

and steps 13, 14 and 15 can be deleted. In this evaluation, instead of using words, we used just two character words and our indexing strategy is called short hybrid term indexing.

In the formal runs, we adopted two indexing strategies. One is bigram indexing and the other is hybrid-term indexing [9]. We only submitted the results with PRF. In our PRF, we used the previous method [9] to selecte 75 terms from the top 7 documents to expand the original terms. Table 1 shows the retrieval effectiveness of our system.

Query Type	Idx Unit	Rigid (%)		Relax (%)	
		MAP	P@10	MAP	P@10
T	B	31.3	39.2	36.0	52.4
	H	29.0	37.0	31.9	49.2
D	B	33.3	41.2	40.0	55.4
TDCN	B	35.7	45.4	41.9	59.4
	H	43.5	49.4	51.0	62.8

Table 1: Retrieval effectiveness of submitted formal runs. Key: T for title queries, D for description queries, TDCN for long queries, B for bigram indexing and H for hybrid-term indexing

Figure 1 shows the relative rigid MAP of bigram indexing for T queries. The relative rigid MAP is rigid MAP of our system for a particular query minus the average rigid MAP across the MAPs of the different participants for that title query. We observe that the performances of 30 queries of our system are better than the average precision.

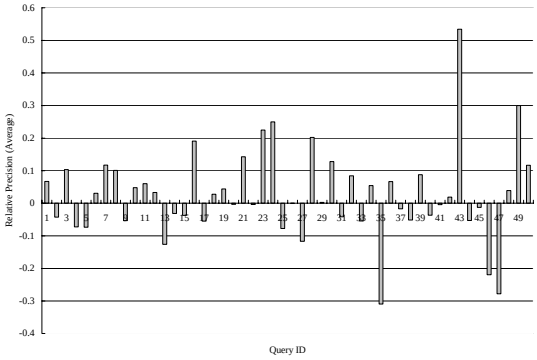

Figure 1: The rigid MAP of different T queries of bigram indexing relative to the rigid MAP of the corresponding queries averaged across all formal runs

Figure 2 shows the relative rigid MAP of bigram indexing for TDCN queries. More than 30 queries obtained the better performance than the average. The rigid MAP of the 47th query is lower than the average by 35%. But for the same query our rigid MAP of hybrid-term indexing is 85.9%, which is higher than the average by 50%.

Figure 3 shows the difference between the rigid MAP of T query and TDCN query of bigram indexing of our system in the formal runs. We observe that the rigid MAP of long queries is not always higher than that of short queries. Especially for the 43rd and 49th queries, the rigid MAP of these two long queries are worse than that of short queries by 59% and 35%. So it can not always improve the performance by adding more terms into the query.

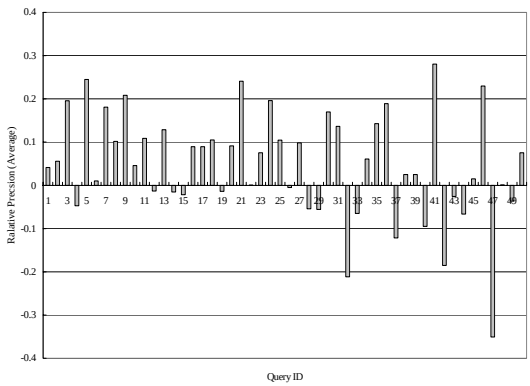

Figure 2: The rigid MAP of different TDCN queries of bigram indexing relative to the rigid MAP of the corresponding queries averaged across all formal runs

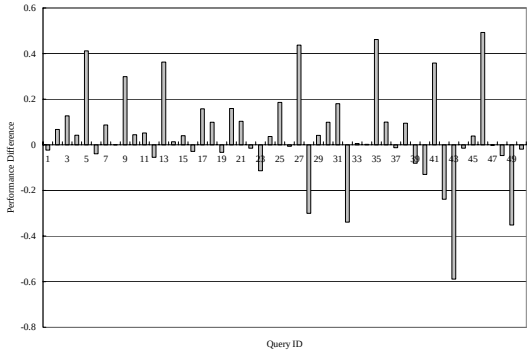

Figure 3: The rigid MAP difference of bigram indexing between TDCN queries and T queries of our formal runs

We found that the poorly performing title queries are not always the same as the poorly performing TDCN queries. For example the rigid MAP of the 35th title query of our system is worse than the average, but the rigid MAP of the same TDCN query is better than the average.

2.2 Results without PRF

In the following informal runs using the same settings as the formal runs, we also obtain the results of T, D and TDCN queries except without PRF. Table 2 shows the retrieval effectiveness of our system. For every query type and for both rigid and relax judgements, the MAP of our system using PRF is better than the corresponging MAP of our system without PRF in Table 2. It appears that PRF can enhance retrieval effectiveness for our retrieval system for NTCIR-5.

Query Type	Idx Unit	Rigid (%)		Relax (%)	
		MAP	P@10	MAP	P@10
T	B	27.8	34.4	33.7	47.2
	H	27.8	38.0	33.7	51.8
D	B	26.7	37.2	31.6	52.0
TDCN	B	32.6	43.0	38.1	56.8
	H	39.4	48.2	45.6	61.2

Table 2: Retrieval effectiveness of our runs (using the same settings at formal runs but) without PRF

3 Fine-tuning existing methods

In this section, we fine-tune our PRF methods, apply title re-ranking and merge retrieval lists using different indexing strategies.

3.1 Title re-ranking plus PRF

In [9], we developed a technique called title re-ranking. It tries to re-rank the documents based on the matching score between the title query and the title of the documents. The re-ranking function *sim'(.)* is:

$$sim'(q,d_i) = \left(sim(q,d_i) - m\right) \times M(q_t, t(d_i)) + m$$

where $sim(q,d_i)$ is the original similarity score, m is the minimum original similarity score in the top n documents, $t(d_i)$ is the title of the i-th document, q_t is the corresponding title query of q, and $M(.)$ is the number matched specific terms between the title query and the title of the document. This re-ranking function guarantees the top n documents will remain in the top n ranks of the re-ranked list because $sim'(q,d_i) \geq m$ for all top n documents.

In NTCIR-4, our title re-ranking strategy can improve the performance of short query but hurt the performance of long query. Afterwards we found a bug in our system. In NTCIR-5, we want to test if this strategy can improve the performance of long query after we fixed the program bug.

Table 3 shows the retrieval performance of our system with title re-ranking plus PRF. In PRF we also select 75 terms from top 7 documents. Title re-ranking strategy can improve the performance of most runs except long query based on hybrid-term indexing. For long query based on bigram indexing, title re-ranking strategy can improve the rigid MAP by 3.6%.

Query Type	Idx Unit	Rigid (%)		Relax (%)	
		MAP	P@10	MAP	P@10
T	B	32.9	39.0	38.6	51.8
	H	32.8	41.6	34.0	51.0
TDCN	B	39.3	45.6	46.5	59.8
	H	42.3	48.6	49.3	62.4

Table 3: Retrieval effectiveness of our bigram indexing with PRF followed by title re-ranking

3.2 PRF Run with new expansion terms

In the formal runs for bigram indexing, we selected 75 bigrams from the top 7 documents in the retrieval list to expand the original query. Because some of these bigrams are the same as the bigrams of the query, we do not know the exact number of the bigrams added to the original query. We want to control the number of expansion bigrams. Therefore, we only count the number of bigrams which do not occur in the query. Table 4 shows the retrieval effectiveness of bigram indexing for T query when we do not count the number of bigrams which occur in the query. We can improve the rigid MAP from 32.9% to 35.0% when we select 75 new expansion bigrams from the top 7 documents.

Query Type	M	Rigid (%)		Relax (%)	
		MAP	P@10	MAP	P@10
T	75	35.0	42.2	41.4	55.4
	100	35.1	41.6	41.5	55.0
TDCN	75	40.8	47.3	48.2	62.1
	100	41.1	48.0	49.0	62.6

Table 4: Retrieval effectiveness of bigram indexing with PRF followed by title re-ranking, not counting the bigrams occurred in the query. Key: M means the number of the bigrams which are added to the original query

3.3 PRF Run with trigrams

We try to use another method to select expansion terms from the top *N* documents. A trigram occurring frequently may hold some specific information. If we can find these trigrams and spit them into bigrams, it is likely to improve the performance to use these bigrams to expand the original query.

The algorithm is as follows. First, we get all the trigrams from the top *N* documents. Because we can not get the document frequency of trigram from the bigram indexing directly, we use the document frequencies of the two component bigrams of the trigram to estimate document frequency of the trigram. In our algorithm we set the minimum document frequency of the two bigrams as an estimate of the document frequency of the trigram. Then, we use our previous term selection method *S3* [9] to select the trigrams. Afterwards, we obtain the top ranked trigrams, we split them into bigrams and use these bigrams to expand the query.

M	Rigid (%)		Relax (%)	
	MAP	P@10	MAP	P@10
5	29.2	36.2	35.4	49.4
20	30.9	37.8	36.7	51.4
35	31.5	39.4	37.2	52.2
50	31.6	38.8	37.4	53.4
75	31.8	39.6	37.3	53.2

Table 5: Retrieval effectiveness of T query of bigram indexing with trigram PRF without title re-ranking. Key: M means the number of the trigrams which are added to the original query

Table 5 shows the retrieval effectiveness of T query of our bigram indexing with trigram PRF. To compare the results with our formal runs, we do not use title re-ranking. We can see the performance can be improved slightly when we selected 75 trigrams.

3.4 Merging Retrieval Lists

We found the performance of bigram indexing is different with that of hybrid term indexing. Figure 4 shows the difference of rigid MAP of T query with PRF following title re-ranking strategy between hybrid term indexing and bigram indexing.

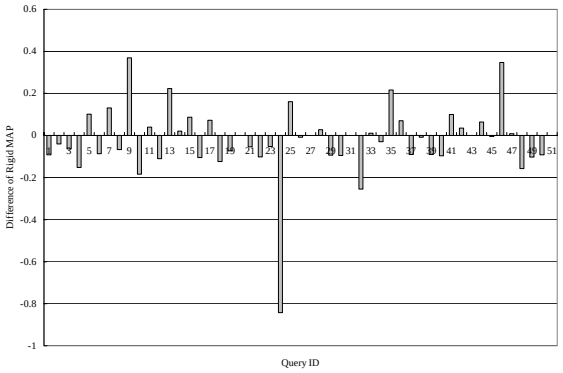

Figure 4: The difference of rigid MAP of T queries of hybrid-term index and bigram index, using PRF followed by title re-ranking

We try to merge the two retrieval lists to get better results. We put the results of bigram indexing and hybrid term indexing together and sort them according to the similarity score. The first 1000 documents for each query are our merge results.

Table 6 shows the results of merging two retrieval lists. We can improve the rigid MAP of title query for bigram indexing in the formal runs by 6%, but only 1% for TDCN query.

Query Type	Rigid (%)		Relax (%)	
	MAP	P@10	MAP	P@10
T	39.4	46.0	43.3	57.4
TDCN	43.2	49.2	50.4	63.8

Table 6: Retrieval effectiveness of merging the results of bigram indexing and hybrid term indexing

4 Simulated Relevance Feedback Run

We want to know what kind of terms can improve the performances and how much the performances can be improved. We can improve the performance if we can detect the relevant documents from the top 7 documents in the retrieval list of the first run and select the terms from these relevant documents to expand the original query. Effectively, we are simulating a single iteration relevance feedback without negative terms from the irrelevant documents.

We use the standard Rochio formula to select the expansion terms:

$$\vec{q}_m = \alpha\vec{q} + \frac{\beta}{|D_r|}\sum_{\forall \vec{d}_j \in D_r}\vec{d}_j - \frac{\gamma}{|D_n|}\sum_{\forall \vec{d} \in D_{n_j}}\vec{d}_j$$

…[4.1]

Because we can delect all the relevant documents and only select the terms from these documents, so here γ is 0, and we set α is 0.3, so β is 0.7.

Table 7 shows the retrieval effectiveness of hybrid-term indexing with PRF for T query. We detect at most *M* relevant documents from top 7 documents based on the rigid relevance judgment file and select 75 expansion terms from the relevant documents.

Another run we have tried is to select the expansion terms from all the relevant documents. First we get all the relevant documents according to the correct answer, and then use the formula [4.1] to select some terms from these documents to expand the query. Table 8 shows the retrieval effectiveness of title queries of the bigram indexing when we select the expansion terms from all the relevant documents.

M	Rigid (%)		Relax (%)	
	MAP	P@10	MAP	P@10
1	47.0	53.0	51.7	66.0
2	48.5	53.6	52.6	66.4
3	48.7	54.2	53.0	66.6
4	48.8	53.6	53.1	66.4

Table 7: Retrieval effectiveness of T query of hybrid-term indexing based on correct answer, selecting terms from the relevant documents in the top 7 documents. Key: M means the maximum number of the identified relevant documents

Idx Unit	M	Rigid (%)		Relax (%)	
		MAP	P@10	MAP	P@10
H	5	47.3	53.8	53.2	66.8
	10	50.4	57.8	54.1	68.8
	25	54.5	61.4	54.3	69.8
	50	55.4	58.4	53.6	66.8
B	5	45.2	50.8	51.7	64.4
	10	50.2	56.6	55.9	67.8
	25	55.8	61.4	60.2	72.4
	50	60.3	65.4	62.1	74.4

Table 8: Retrieval effectiveness of T query using bigram indexing when we select expansion terms from all the relevant documents based on rigid judgment. Key: M is the number of the expansion terms we selected from all the relevant documents

5 Efficiency

5.1 Space efficiency

Table 9 shows the number of Chinese documents of NTCIR5 data collection and the storage of the data in our system. Because there are two bigram indices in our system, we list the number of unique bigrams separately in each bigram index and there is a large number of bigram terms in both indices.

Number of documents	901,446
Storage	3.5G
Number of stop words	105
Number of publishers	4
Number of unique bigrams in each bigram index	2,901,060+ 3,695,672
Total Number of unique bigrams in two bigram indices	4,666,069
Number of unique hybrid-term	3,481,106

Table 9: Summary of the Chinese data collection

Because of the limited memory of our machine we have to keep two separated bigram indices instead of one bigram index.

Table 10 shows the storage cost of the inverted index and the dictionary in gigabytes. It is well known that bigram indexing is larger than the hybrid-term indexing. The last column is the relative storage compared with the storage of the document collection.

Idx Unit	Index Size	Dictionary Size	Total	Relative Storage
B	2.7 G (1.1+1.6)	0.18G (0.08+0.11)	2.88G	82 %
H	1.5G	0.1G	1.6G	46 %

Table 10: Storage cost (in gigabytes) of the inverted index and the dictionary

5.2 Access Time

Figure 5 shows the scatter diagram of the retrieval time per TDCN query with PRF. '+' means the access time of bigram indexing, 'o' means the access time of hybrid-term indexing. We can observe that the access time of bigram indexing is longer than that of hybrid-term indexing because bigram indexing generated more terms. The access time appears to be varying linearly with the number of unique query terms, similar to the findings in [10].

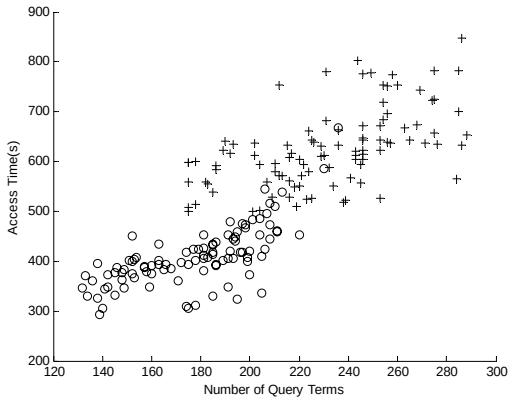

Figure 5: Scatter diagram of the access time against the number of unique query terms of TDCN retrieval based on bigram indexing and hybrid-term indexing with PRF

6 Summary

In this participation, we show that title re-ranking strategy can improve the retrieval effectiveness both short and long queries for bigram indexing. In PRF when we do not count the number of the bigrams which occurs in the query, we can get better performance, which means that adding more terms into the query can improve the performance.

We want to know how much the performance can be improved when we select the expansion terms from the relevant documents from the top 7 documents in the retrieval list in order to simulate a single iteration relevance feedback without negative terms from irrelevant documents. Our experiments show that the rigid MAP of title query is near 49% if we detect at most 4 relevant documents in the top 7 documents. One of our furture work is to find a way to select these terms from the top *N* documents.

Acknowledgement

We would like to thank the Center for Intelligent Information Retrieval, University of Massachusetts (UMASS), for facilitating Robert Luk to develop in part the basic IR system, when he was on leave at UMASS. This work is supported by the CERG Project # PolyU 5183/03E. We are grateful to ROCLING for their word list

References

[1] C. Buckley, G. Salton, J. Allan. Automatic retrieval with locality information using Smart. *Proceedings of TREC 1*, 1992, pp. 59-72.

[2] S.E. Roberston, S. Walker, S. Jones, M. Hancock-Beaulieu and M. Gatford. Okapi at TREC-3, *Proceedings of TREC-3*, 1994, pp. 109-128.

[3] W. Lam, C-Y Wong and K.F. Wong. Performance Evaluation of Character-, Word- and N-Gram-Based Indexing for Chinese Text Retrieval, *Proceedings of IRAL 97*, Japan, 1997.

[4] J-Y. Nie and F. Ren. Chinese information retrieval: using characters or words, *Information Processing and Management*, **35**:443-462, 1997.

[5] M-K. Leong and H. Zhou. Preliminary qualitative analysis of segmented vs bigram indexing in Chinese, *Proceedings of the Sixth Text Retrieval Conference (TREC-6)*, Gaithersburg, Maryland, November, 1997, pp. 19-21.

[6] P. Fung and D. Wu. Statistical Augmentation of a Chinese Machine-readable dictionary, *Proceedings of Workshop on Very Large Corpora*, Kyoto, August, 1994, pp. 69-85.

[7] J. Guo. Critical tokenization and its properties, *Computational Linguistics*, **23:4:** 569-596, 1997.

[8] Z. Wu and G. Tseng. ACTS: An Automatic Chinese Text Segmentation System for Full Text Retrieval, *Journal of the American Society of Information Science*, **46(2)**: 83-96, 1995.

[9] R. W.P. Luk and K.F. Wong. Pseudo Relevance Feedback and Title Re-ranking for Chinese Information Retrieval, *Proceedings of the NTCIR4 Workshop*, Tokyo, 2-4 June, 2004.

[10] P. Vines and J. Zobel. Efficient building and querying of Asian language document databases. *Proceedings of Information Retrieval for Asian Languages*, pp. 118-125, 1999.