Okapi Chinese text retrieval experiments at TREC–6

Xiangji Huang* S E Robertson*

1 Introduction

The focus of the Okapi TREC–6 Chinese experiments is on investigating the effectiveness of different automatic indexing methods and phrase weighting for retrieval based on probabilistic models over Chinese text. We compare different probabilistic weighting methods based on a range of word and single character approaches.

There are two indexing methods used in our experiments. One indexing method is to use linguistic units (words, compound words and phrases) in texts as indexing terms to represent the texts. We refer to this method as the word approach. For this approach, text segmentation, which divides text into linguistic units, is regarded not only as a necessary precursor but also as a bottleneck of this kind of system [1]. The other method for indexing texts is based on single Chinese characters, in which texts are indexed by the characters appearing in the texts [2]. By using single character approaches, a search could be conducted for any multi-character word or phrase identified at search time, no matter whether this word or phrase is in the dictionary.

Three automatic runs city97c1, city97c2 and city97c3 were submitted in TREC–6. All the three runs were based on the whole topic. City97c1 and city97c3 are for word indexing approach with different parameter values and city97c2 is for character indexing approach.

The runs reported here are all on the TREC–6 collection of 26 new Chinese topics and 164768 documents. The Chinese dictionary we use for our word approach retrieval system contains about 70,000 Chinese words and phrases. Most of these words and phrases come from a manually constructed dictionary in China. We expanded this dictionary while working on the Chinese TREC experiments.

2 Chinese text segmentation

2.1 Concepts and methods

A Chinese word is the minimal linguistic unit that can be used independently. Most modern Chinese words consist of more than one ideographic character and the number of characters in a word varies. Since a Chinese text is a linear sequence of non-spaced or equally spaced ideographic characters, we must either apply a dictionary-based Chinese word segmentation method to the text, or index and search in terms of single Chinese character.

There are different requirements on Chinese text segmentation for different applications. For example, machine translation and natural language processing require the correct segmentation of Chinese text according to Chinese syntax. For the purpose of information retrieval, we may not have to segment Chinese text correctly.

Segmentation based on a big dictionary can be classified into three groups. The first is the longest match, for which text is sequentially scanned to match the dictionary. The longest matched strings are taken as indexing and search tokens and shorter tokens within it are discarded. Since longer tokens in the dictionary are more specific, longest match will generate fewer tokens with more specific meaning.

The second is the shortest match, for which text is sequentially scanned to match the dictionary. The first matched tokens are taken and the match process started from the next character. With the shortest match method, the segmentation process will generate more tokens with less specific meaning. The third is the overlap match, for which tokens generated from the text can overlap each other across the matching boundary.

For the word approach retrieval system in our experiments, we used the longest match algorithm to segment Chinese texts. By applying this algorithm to Chinese TREC collections, approximately 43.6 million words and phrases were identified. Since no dictionary is expected to include all the words, there must be some unmatched

*Centre for Interactive Systems Research, Department of Information Science, City University, Northampton Square, London EC1V 0HB, UK. email {xjh,ser}@is.city.ac.uk

strings left with dictionary based segmentation algorithms. Unmatched strings can be used as tokens for indexing and search. They can also be used to expand the dictionary. In our experiments, we take each character in the string as a token.

For the single character approach, it is a purely mechanical procedure to segment TREC Chinese texts into single characters. An inverted file of about one gigabyte is generated for the character approach retrieval systems in our experiments.

2.2 Query processing

There are two kinds of automatic methods for query processing in our Chinese text retrieval system. One is character-based method, which uses characters, character pairs and multi-character adjacencies as retrieval keywords. Character pairs and multi-character adjacencies are similar to the bigrams and n-grams investigated by some other researchers [3, 4]. The other is word-based method. Given a word segmentation system, similar methods for characters can be applied to words as a way of allowing phrases to contribute to the matching. We refer to this method as the word-based query processing. Table 1 shows nine methods which have been implemented in our systems.

Algorithms	Query processing	Database	Number of weighting methods
M_0	characters	character	1
M_1	characters+character-pairs	character	8
M_2	characters+multi-character adjacencies	character	8
M_3	words	character	8
M_4	words	word	1
M_5	words+word-pairs	character	8
M_6	words+word-pairs	word	4
M_7	words+multi-word adjacencies	character	8
M_8	words+multi-word adjacencies	word	4

Table 1: Illustration of retrieval algorithms

We use M_5 and M_6 for all the TREC experiments reported here. Thus the query-processing is word-based, but both forms of document processing are used. The text in all parts of the Chinese topics were treated in the same way. Text segmentation is applied to the topics first. Only pairs of the adjacent segmented terms which occur in the same subfield of the topic are regarded as new potential phrases. All these segmented terms and new potential phrases are ranked by the values of their weights multiplied by the within-query frequencies. The top 19 terms are used as keywords for searching the word index and for searching the character index.

3 Probabilistic model

3.1 Basic weighting function

The basic weighting functions are based on the Robertson-Sparck Jones weight [5], which approximates to inverse collection frequency (ICF) shown in equation 1 when there is no relevance information.

$$ICF = \log \frac{N - n + 0.5}{n + 0.5} \tag{1}$$

The weighting function we use for Okapi TREC–6 Chinese experiments is given as follows. This function is extended from the BM25 function [7].

$$w = \frac{(k_1 + 1)tf}{K + tf} \log \frac{N - n + 0.5}{n + 0.5} . \frac{(k_3 + 1)qtf}{k_3 + qtf} \oplus k_d y \tag{2}$$

where

N is the number of indexed documents in the collection,
n is the number of documents containing a specific term,
R is the number of known relevant documents for a specific topic,
r is the number of known relevant documents containing a specific term,
tf is within-document term frequency,
qtf is within-query term frequency,

$$y = \begin{cases} \ln(\frac{dl}{avdl}) + \ln(x_1) & \text{if } 0 < dl \leq rel\_avdl; \\ (\ln(\frac{rel\_avdl}{avdl}) + \ln(x_1))(1 - \frac{dl - rel\_avdl}{x_2.avdl - rel\_avdl}) & \text{if } rel\_avdl < dl < \infty. \end{cases}$$

where dl is the length of the document, $avdl$ is the average document length rel_avdl is the average relevant document length in TREC–5, x_1 and x_2 are two parameters to be set,

the k_i are tuning constants, which depend on the database and possibly on the nature of the topics and are empirically determined,[1]
K equals $k_1((1-b) + b.dl/avdl)$, and
the $\oplus$ in the formula indicates that the following component is added only once per document, rather than for each term.

This formula now defines the weight of a term (that is, the contribution of that term to the total score for the document) in the context of an individual document.

The y bit in BM25 as defined for TREC–5 equals $|Q|.\frac{avdl-dl}{avdl+dl}$, where Q is a query, containing terms T (although it was not actually used in TREC–5, as k_d was set to zero). That means y will decrease with dl, from a maximum as $dl \to 0$, through zero when $dl = avdl$, and to a minimum as $dl \to \infty$. But it should be better that y will reach a maximum as $dl \to rel\_avdl$, through zero when $dl = avdl/x_1$ (or $dl = x_2.avdl$), and to a minimum as $dl \to 0$ (or $dl \to \infty$). So a new y bit was designed for TREC–6, where x_1 and x_2 were set to 3 and 26 respectively, after some experiments on the TREC–5 collection. We may get better results by setting other values for x_1 and x_2.

3.2 Phrase weighting for Chinese

We need weighting functions which will enable us to cope with phrases in a word-based system, and with words or phrases in a character-based system. Suppose, then, that we have a sequence of j adjacent units $t_1t_2...t_j$ (characters or words) constituting a single larger unit (word or phrase). In the Robertson/Sparck Jones model, each unit (large or small) has a "natural" weight, given by the formula; let these be w_{t_i} and $w_{t_1t_2...t_j}$ respectively. Then we can suggest a number of weighting functions which satisfy (or will probably satisfy) the above condition or something like it. Table 2 gives a few such functions which have been implemented.

Weight methods	$w(t_1\ adj\ t_2\ adj \cdots adj\ t_j)$	$w(t_1 \wedge t_2 \wedge \cdots \wedge t_j)$
$Weight_0$	$\sum_{i=1}^{j} w_{t_i} + j^k * w_{t_1t_2...t_j}$	$\sum_{i=1}^{j} w_{t_i}$
$Weight_1$	$w_{t_1t_2...t_j}$	$\sum_{i=1}^{j} w_{t_i} - j^k$
$Weight_2$	$w_{t_1t_2...t_j}$	$\sum_{i=1}^{j} w_{t_i}$
$Weight_3$	$\sum_{i=1}^{j} w_{t_i} + log\frac{\#(t_1 \wedge t_2 \wedge \cdots \wedge t_j)}{\#(t_1\ adj\ t_2\ adj \cdots adj\ t_j)}$	$\sum_{i=1}^{j} w_{t_i}$
where $\#(t)$ indicates the number of documents containing the term t and k is another tuning constant		

Table 2: Weight methods

None of these functions has a very strong probabilistic basis, beyond the attempt to satisfy the condition. Each has two versions, one applied to words and the other to characters, so there are eight functions in all. The "natural" weights come from equation 2: w_{t_i} is obtained by applying the equation to term t_i, and $w_{t_1t_2...t_j}$ comes from applying the same equation to the combined term $t_1t_2...t_j$.

In our experiments, the value of k was 0.5. The average document length is 891 bytes, The average relevant document length in TREC–5 is 1399 bytes.

[1]For our experiments, the k_i values will be given in Section 5

4 Experimental results and performance comparison

Extensive experiments have been done on an SGI Challenging L machine to investigate: 1. the effect of probabilistic approach on Chinese text retrieval; 2. the difference between word approach and character approach; 3. the effect of different phrase weighting functions and of varying their parameters.

All the results reported here are from Chinese ad hoc experiments on the various TREC collections. These experiments represent only a small proportion of the range suggested by the foregoing discussion. Experiments are still continuing. In our experiments, the relevance judgements for each topic come from the human assessors of NIST. Statistical evaluation was done by means of the latest version TREC evaluation program, for which we are grateful to Chris Buckley. The abbreviated captions in the statistical tables have the following meanings: Average Precision: average precision over all 11 recall points (0.0, 0.1, 0.2,..., 1.0); R Precision: precision after the number of documents retrieved is equal to the number of known relevant documents for a query; Precision 100 docs: precision after 100 documents have been retrieved.

Three automatic runs city97c1, city97c2 and city97c3 were submitted for evaluation in TREC–6. All the three runs were based on the whole topic. City97c1 and city97c3 are for word indexing approach with different parameter values and city97c2 is for character indexing approach. For these three submitted runs, the values of k_1 and k_3 were set to 2.0 and 5.0. For city97c1 and city97c3, the k_d were set to 6 and 15 respectively. The k_d was set to 6 for city97c2.

The three submitted Chinese results for TREC ad hoc experiments are shown in Table 3 and Table 4. All of these three runs for the whole 26 topics are around the median score of the groups participating in the Chinese track.

The results for the different weighting methods of the character approach are presented in Table 5. Table 6 gives some more evaluation results for the word approach. The values of k_1, k_3 and b for the above runs are 2.0, 5.0 and 0.75 respectively. In these two tables, the method of $Weight_1$ gives the best result comparing to the other methods for both the word and character approaches.

Table 7 gives more details of the evaluations of the four runs for the word and character approaches by setting the values of parameter $k_d = 0$ and 15 respectively. From this table, we can see that the character approach city97c02 gives almost identical performance to the word approach city97w02. By setting k_d to 15, the word and character approach perform 7.20% and 11.28% better than the word approach city97w02 with k_d equal to 0.

Run	> median	= median	< median
city97c1	7	0	19
city97c2	10	1	15
city97c3	8	0	18

Table 3: Comparative Chinese Results

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
city97c1	0.4838	2495	0.5119	0.4804
city97c2	0.5047	2589	0.5221	0.4900
city97c3	0.4943	2461	0.5178	0.4835

Table 4: Chinese Ad hoc Results

5 Discussion

Our overall results are a little disappointing. However, we have a great deal more testing to do, which we hope will enable us to make up some of the ground.

One result of particular interest to us relates to the relative success of Weight1. Although, as stated earlier, the formula does not have a strong justification in terms of the probabilistic model, there is at least a qualitative argument which might explain the result. There are two parts to this argument.

It may be seen from Table 2 that Weight1 does not assign the usual sum of individual term weights to the conjunction of a set of terms. Instead it reduces that sum by a certain amount. If we consider the occurrence of a

Run	Weighting Method	k_d	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
city97c01	$Weight_0$	0	0.4563	2525	0.4618	0.4565
city97c29	$Weight_1$	0	0.4808	2565	0.4972	0.4738
city97c03	$Weight_2$	0	0.4730	2519	0.4972	0.4669
city97c04	$Weight_3$	0	0.4223	2423	0.4335	0.4319
city97c05	$Weight_0$	2	0.4596	2532	0.4650	0.4600
city97c30	$Weight_1$	2	0.4900	2579	0.5111	0.4812
city97c07	$Weight_2$	2	0.4832	2541	0.5096	0.4750
city97c08	$Weight_3$	2	0.4271	2424	0.4397	0.4385
city97c09	$Weight_0$	6	0.4632	2538	0.4697	0.4627
city97c32	$Weight_1$	6	0.5047	**2589**	0.5221	0.4900
city97c11	$Weight_2$	6	0.4964	2547	0.5178	0.4819
city97c12	$Weight_3$	6	0.4324	2436	0.4523	0.4819
city97c13	$Weight_0$	8	0.4649	2544	0.4718	0.4638
city97c33	$Weight_1$	8	0.5084	2588	0.5244	0.4927
city97c15	$Weight_2$	8	0.5011	2542	0.5198	0.4854
city97c16	$Weight_3$	8	0.4343	2436	0.4526	0.4504
city97c17	$Weight_0$	10	0.4666	2547	0.4756	0.4638
city97c34	$Weight_1$	10	0.5113	2580	**0.5288**	0.4958
city97c19	$Weight_2$	10	0.5034	2536	0.5205	0.4865
city97c20	$Weight_3$	10	0.4358	2426	0.4551	0.4485
city97c21	$Weight_0$	15	0.4697	2549	0.4787	0.4658
city97c35	$Weight_1$	15	**0.5129**	2560	0.5274	**0.4981**
city97c23	$Weight_2$	15	0.5068	2522	0.5209	0.4888
city97c24	$Weight_3$	15	0.4374	2413	0.4553	0.4488
city97c25	$Weight_0$	20	0.4718	2543	0.4816	0.4662
city97c36	$Weight_1$	20	0.5095	2526	0.5227	0.4950
city97c27	$Weight_2$	20	0.5052	2498	0.5209	0.4888
city97c28	$Weight_3$	20	0.4360	2395	0.4551	0.4504

Table 5: Phrase weighting for character approaches

phrase as a subset of the occurrence of the conjunction of the corresponding terms, and if we give some extra weight to the phrase, the probabilistic model suggests that we should also reduce the scores of the remaining documents in the conjunction (those that do not include the phrase). This in effect is achieved by Weight1.

This result suggests a new look at phrase weighting schemes in English as well as in Chinese.

Acknowledgments

This research is supported by ORS award from Committee of Vice-Chancellors and Principals of United Kingdom and Centenary Scholarship from City University.

References

[1] Wu, Z.; Tseng, G. Chinese text segmentation for text retrieval: Achievement and Problems. *Journal of the American Society for Information Science* 44 (9): 532-541; 1993

[2] Chen, G. On single Chinese character retrieval system. *Journal of Information* 11 (1): p11-18; 1992 (in Chinese)

[3] Chien, L.F. Fast and quasi-natural language search for gigabits of Chinese texts. In *SIGIR 95*: p112-120; 1995.

[4] Willett, P. Document retrieval experiments using indexing vocabularies of varying size. II. Hashing, truncation, digram and trigram encoding of index terms. *Journal of Documentation* 35 (4): p296-305; 1979

[5] Robertson, S.E.; Sparck Jones, K. Relevance weighting of search terms. *Journal of the American Society for Information Science* 27: p129-146; 1976

Run	Weighting Method	k_d	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
city97w01	$Weight_0$	0	0.4195	2419	0.4341	0.4188
city97w02	$Weight_1$	0	0.4609	2434	0.4852	0.4542
city97w03	$Weight_2$	0	0.4586	2425	0.4832	0.4508
city97w04	$Weight_3$	0	0.4561	2447	0.4746	0.4550
city97w05	$Weight_0$	2	0.4251	2433	0.4380	0.4262
city97w06	$Weight_1$	2	0.4704	2469	0.4946	0.4688
city97w07	$Weight_2$	2	0.4683	2464	0.4929	0.4669
city97w08	$Weight_3$	2	0.4630	2476	0.4802	0.4627
city97w09	$Weight_0$	5	0.4321	2436	0.4472	0.4373
city97w10	$Weight_1$	5	0.4810	2490	0.5062	0.4773
city97w11	$Weight_2$	5	0.4792	2485	0.5035	0.4762
city97w12	$Weight_3$	5	0.4702	2493	0.4844	0.4669
city97w13	$Weight_0$	6	0.4340	2440	0.4484	0.4388
city97w14	$Weight_1$	6	0.4838	**2495**	0.5119	0.4804
city97w15	$Weight_2$	6	0.4823	2492	0.5062	0.4800
city97w16	$Weight_3$	6	0.4714	2491	0.4848	0.4681
city97w17	$Weight_0$	8	0.4371	2436	0.4546	0.4462
city97w18	$Weight_1$	8	0.4879	2489	0.5142	0.4812
city97w19	$Weight_2$	8	0.4868	2489	0.5114	0.4800
city97w20	$Weight_3$	8	0.4737	2486	0.4901	0.4696
city97w21	$Weight_0$	10	0.4395	2432	0.4597	0.4477
city97w22	$Weight_1$	10	0.4904	2478	0.5166	0.4812
city97w23	$Weight_2$	10	0.4897	2478	0.5166	0.4800
city97w24	$Weight_3$	10	0.4756	2478	0.4920	0.4712
city97w25	$Weight_0$	15	0.4434	2422	0.4667	0.4492
city97w26	$Weight_1$	15	**0.4943**	2461	**0.5178**	**0.4835**
city97w27	$Weight_2$	15	0.4935	2465	0.5143	0.4819
city97w28	$Weight_3$	15	0.4772	2458	0.4915	0.4723
city97w29	$Weight_0$	20	0.4435	2411	0.4690	0.4535
city97w30	$Weight_1$	20	0.4927	2440	0.5107	0.4808
city97w31	$Weight_2$	20	0.4919	2437	0.5099	0.4808
city97w32	$Weight_3$	20	0.4730	2427	0.4892	0.4731
city97w33	$Weight_0$	50	0.4309	2245	0.4661	0.4446
city97w34	$Weight_1$	50	0.4414	2201	0.4789	0.4581
city97w35	$Weight_2$	50	0.4397	2193	0.4791	0.4565
city97w36	$Weight_3$	50	0.4339	2212	0.4671	0.4485

Table 6: Phrase weighting for word approaches

Run	k_d	Indexing Method	Average Precision
city97w02	0	*word*	0.4609
city97c29	0	*character*	0.4808 (+4.32%)
city97w26	15	*word*	0.4943 (+7.24%)
city97c35	15	*character*	0.5129 (+11.28%)

Table 7: Chinese Ad hoc Results Comparison

[6] Beaulieu, M.M., Gatford, M, Huang, X., Robertson, S.E., Walker, S. and Williams, P. Okapi at TREC–5". In D.K.Harman, editor, *Proceedings of the Fifth Text REtrieval Conference (TREC–5)*. NIST Special Publication 1997.