Natural Language Engineering **8** (2/3): 193–207. © 2002 Cambridge University Press
DOI: 10.1017/S1351324902002899 Printed in the United Kingdom

Selecting effective index terms using a decision tree

TOKUNAGA TAKENOBU, KIMURA KENJI,
OGIBAYASHI HIRONORI and TANAKA HOZUMI
Department of Computer Science, Tokyo Institute of Technology, Tokyo, Japan

(*Received 27 June 2001; revised 4 February 2002*)

Abstract

This paper explores the effectiveness of index terms more complex than the single words used in conventional information retrieval systems. Retrieval is done in two phases: in the first, a conventional retrieval method (the Okapi system) is used; in the second, complex index terms such as syntactic relations and single words with part-of-speech information are introduced to rerank the results of the first phase. We evaluated the effectiveness of the different types of index terms through experiments using the TREC-7 test collection and 50 queries. The retrieval effectiveness was improved for 32 out of 50 queries. Based on this investigation, we then introduce a method to select effective index terms by using a decision tree. Further experiments with the same test collection showed that retrieval effectiveness was improved in 25 of the 50 queries.

1 Introduction

Indexing is a key technology in Information Retrieval (IR), and converts a natural language text (a document) into a representation that properly describes the content of the document and can be handled efficiently by computers. Significant properties of indexing are exhaustivity and specificity. Exhaustivity is a property of index descriptions, and indicates the extent to which an index description covers the document content. Specificity is a property of an individual index term, and indicates the extent to which each index term is specific to a document (Sparck Jones 1972).

In conventional IR techniques, a document is represented in terms of a set of index terms, which are often single words or word stems. Index terms can be weighted on the basis of their frequency in order to rank retrieved documents. Using single words as index terms generally has good exhaustivity, but poor specificity due to word ambiguity.

A commonly used example of this problem is the word 'bank', which has two distinct meanings: a financial institution and the side of a river. In an IR system using single words as index terms, a query including the word 'bank' will retrieve all documents including 'bank' irrespective of the meaning of 'bank' in the query. One

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

approach to solving the ambiguity problem is to introduce index terms more complex than single words, such as phrases. In the previous example, we can distinguish the two meanings by using phrasal index terms such as 'bank of the Seine' and 'Bank of Japan'.

There have been many attempts to introduce complex index terms into IR systems (Strzalkowski 1995; Mitra *et al.* 1997; Voorhees 1999). Some have tried to analyze documents using natural language processing (NLP) techniques to extract linguistically motivated constructions such as phrases or predicate-argument structures. Others have tried to extract useful chunks of words on a statistical basis, with the chunks often referred to as 'statistical phrases' (Keen and Hartley 1994). Statistical phrases can be obtained with less computational cost than linguistically motivated constructions, but they have obvious limitations, such as there being no guarantee that each index term has a genuine meaning, the possibility that relations between distant words will be difficult to capture, and so on.

The performance of these attempts to include complex index terms, however, has not been consistently good. A primary reason for this is that introducing complex index terms increases the diversity of index terms, thus increasing mismatches between the index terms. Using complex (more specific) index terms increases specificity, but at the expense of exhaustivity.

To gain both specificity and exhaustivity at the same time, we have considered retrieval that consists of two phases and uses different types of index terms in each phase. In the first phase, we use a conventional indexing method to obtain a certain number of documents as retrieval output. Here, we concentrate on maintaining recall by relying on the exhaustivity of conventional single-word index terms. In the second phase, we analyze the retrieved documents more precisely using NLP techniques, and rerank these documents if necessary. In this phase, we aim to gain precision by introducing more complex index terms.

If documents retrieved by conventional methods include many documents relevant to a user's query, we can avoid starting from scratch in applying NLP techniques in the second phase by using the results from the first phase. In addition, to solve the diversity problem concerning index terms, we concentrate on analyzing the results of the first phase, instead of analyzing all documents at one time in the manner of Strzalkowski (1995). Therefore, in our approach, NLP techniques are used to improve the results of conventional retrieval methods, not to replace the conventional methods (Metzler and Haas 1989; Kwok and Chan 1998).

Important issues concerning the two-phase retrieval framework are as follows:

- How many highly ranked documents from the first retrieval should be used in the second retrieval?
- How should we combine single-word and complex index terms in the second phase of retrieval?

In this paper, we focus particularly on the second issue. As mentioned, we use complex index terms not to replace single-word index terms, but to complement them. It is important to identify the cases in which introducing complex index terms can improve the retrieval effectiveness. Mitra, Buckley, Singhal and Cardie (1997)

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

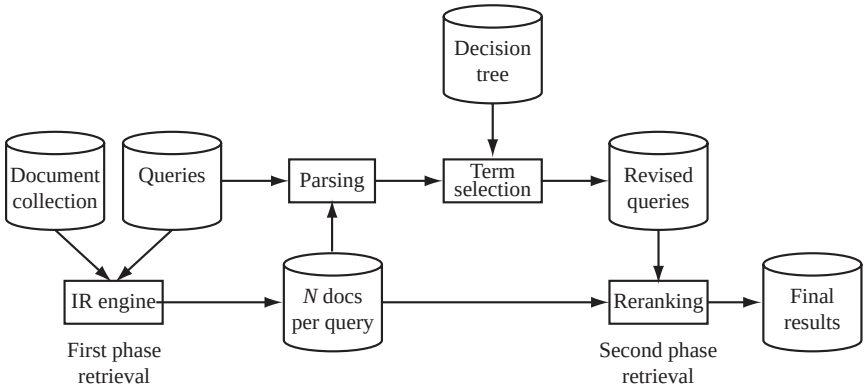


Fig. 1. System overview.

claim their experiments have shown that complex index terms (phrases) are useful when the results of conventional retrieval based on single-word terms are 'moderate' (Mitra *et al.* 1997). However, what constitutes 'moderate' is still an open question.

To answer this question here, we analyze cases in which complex index terms are effective, and also explore the upper bound of improvement enabled by applying complex index terms and the effectiveness of different types of complex index terms. Based on this analysis, we propose a method to select promising index terms from among various types of index terms in the second-phase retrieval. A decision tree is used to achieve this goal; that is, the effectiveness of each index-term candidate is evaluated by referring to its various features.

Figure 1 shows an overview of the system design. In the first-phase retrieval, the top N documents are retrieved as output. These documents and the query are syntactically analyzed and complex index terms are extracted from the analysis result. A decision tree is used to select effective index terms from both these complex index terms and the single-word index terms. The selected index terms constitute a revised query, which is used to rerank the N documents.

In the next section, we describe the syntactic parsing tool used in this research and the methods to extract complex index terms from the parsing results. We look at the effectiveness of index terms and the criteria for formulating a query in section 3. The effectiveness of index terms was evaluated with actual data through our experiments. The decision-tree-based index term selection method is introduced in section 4, and the experimental details are also described in this section. We conclude the paper and look at future work in section 5.

2 Extracting complex index terms

We employ the Apple Pie Parser to parse the query and documents retrieved in the first phase. The Apple Pie Parser is a probabilistic chart parser developed by Sekine at New York University (Sekine and Grishman 1995). The grammar and lexicon of the Apple Pie Parser were automatically constructed from the Penn Treebank

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

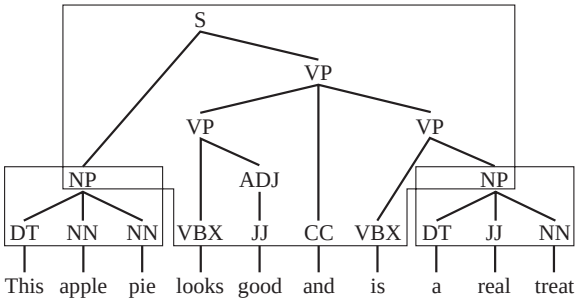


Fig. 2. Example of a parse tree.

(Marcus *et al.* 1993); the grammar uses only two non-terminal symbols, S and NP. This feature enables parser robustness and wide coverage.

The following is an example of a grammar rule used by the Apple Pie Parser:

```
    S → NP VBX JJ CC VBX NP
:structure ''(S ⟨1⟩ (VP (VP ⟨2⟩ (ADJ ⟨3⟩)) ⟨4⟩ (VP ⟨5⟩ ⟨6⟩)))'';
```

As this rule shows, the right side of a rule is a sequence of terminal symbols and either of the non-terminal symbols NP and S; in other words, the rule structure is flattened. To supplement detailed structure, each rule is associated with a structural description, in which the place holder $\langle i \rangle$ corresponds to the ith symbol of the right hand side of the grammar rule. Figure 2 shows an example of a parse tree, where each boxed structure corresponds to a grammar rule of the Apple Pie Parser.

The performance of the Apple Pie Parser is reported as about 70% in both recall and precision on the basis of constituent boundaries (Sekine and Grishman 1995). The average of the cross brackets is 2.64.

From parse trees, the following syntactic structures are extracted using extraction rules:

- subject-verb relation (`sv`): when a noun phrase is followed by a verb, the `sv`-relation is identified between the head of the noun phrase and the verb.
- verb-object relation (`vo`): when a verb is followed by a noun phrase, the `vo`-relation is identified between the verb and the head of the noun phrase.
- adjective-noun relation (`an`): when an adjective is followed by a noun phrase, the `an`-relation is identified between the adjective and the head of the noun phrase.
- noun-noun relation (`nn`): when a noun is followed by another noun, the `nn`-relation is identified between the two nouns. When more than two nouns are consecutive, the rule is applied to each adjacent pair.

For instance, from the parse tree shown in Figure 2, these rules extract the four syntactic relations '`nn`: apple+pie', '`sv`: pie+looks', '`vo`: is+treat', and '`an`: real+treat'. Each word in these relations is stemmed and the relations used as index terms. This procedure is basically the same as that of Strzalkowski (1995).

In addition to these syntactic relation-based index terms, we also use single-word index terms, which are extracted according to the following procedure. From parse

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

trees generated by the Apple Pie Parser, words tagged with 'noun', 'proper noun', 'adverb', 'verb' and 'adjective' are extracted. Stemming and stop-word deletion are then applied to these tagged words. At this stage, each word has part-of-speech information. We consider two types of single-word index terms: one with part-of-speech information and one without. In summary, we extract three types of index term: syntactic relations, single words with part-of-speech information, and conventional single-word index terms.

Term weights are calculated for these index terms according to their type. The weight of a single-word index term without part-of-speech information is calculated using the TF·IDF formula with normalization in a way similar to with the SMART system (Salton 1988). The IDF value is calculated based on the term occurrence in the entire document collection. This weighting scheme is also applied to terms in a query. The weight of a single-word index term with part-of-speech information is also calculated based on TF·IDF with normalization. Here however, the IDF value is calculated based on the term occurrence in the retrieved top N documents. This is because part-of-speech information is not assigned to all documents.

The weight of a syntactic relation-based index term is calculated based on its normalized term frequency. The IDF factor is not used in this case, because such index terms are inherently specific, unlike single-word index terms. Our preliminary experiments showed that introducing the IDF factor into the weight of syntactic relation-based index terms degrades the retrieval performance.

3 Index term effectiveness

To evaluate the effectiveness of each type of index term described in the previous section, we define the effectiveness of each index term and the criteria to decide on an optimal index term set (query).

As an effectiveness measure for index terms, we consider two measures, the term precision and the term F-measure. The term precision is defined as the ratio of relevant documents that include the pertinent index term to all documents that include the index term. In other words, it is the precision of a Boolean retrieval using only that index term as a query. Term recall is similarly defined as the ratio of relevant documents that include the index term to all relevant documents for a given query. From term precision P_t and term recall R_t, the term F-measure F_t is calculated from the following formula (van Rijsbergen 1979):

$$F_t = \frac{2P_tR_t}{P_t + R_t}$$

Given a relevance judgment for documents retrieved by a given query, the effectiveness of each index term can be calculated as described above. The index terms can be ranked according to their effectiveness and we can select effective index terms from this ranked list to formulate a query. We need to decide the number of index terms to be included in a query. For this purpose, we consider two criteria that can be used to fix a cutoff for the index term list.

The first criterion is based on retrieval precision. A sequence of queries is con-

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

Table 1. *Distribution of index terms extracted from documents*

Syntactic relations			Single words with POS		
Index term	Token	Type	Index term	Token	Type
sv	5,698,396	1,157,436	Noun	14,631,645	64,848
vo	2,867,959	537,297	Proper noun	6,669,830	174,897
an	3,594,571	555,213	Verb	4,675,920	7,946
nn	5,302,704	812,161	Adjective	4,357,386	97,156
			Adverb	534,932	3,694

structed by adding one index term at a time to the query from the ranked index term list. The non-interpolated average precision is calculated for each retrieval result and the cutoff is set as that query giving the maximum average precision. To calculate the average precision, it is necessary to rank the retrieved documents. The vector space model is used for this purpose (Salton 1988).

The second criterion is based on the retrieval F-measure. Similar to the precision-based criterion, a sequence of queries is constructed and the F-measure is calculated for the retrieval result of each query. The cutoff is set as that query giving the maximum F-measure. Note that we perform a Boolean retrieval in this case.

In summary, we have four options when formulating a query using the following combinations of index term effectiveness and cutoff criteria.

- E_pC_p: Precision-based effectiveness and precision-based cutoff
- E_FC_p: F-measure-based effectiveness and precision-based cutoff
- E_pC_F: Precision-based effectiveness and F-measure-based cutoff
- E_FC_F: F-measure-based effectiveness and F-measure-based cutoff

To explore the effectiveness of different types of index terms as described in section 2, we conducted experiments using the TREC-7 information retrieval test collection (Voorhees and Harman 1999). The TREC-7 test collection consists of 50 topics (#351–#400) and 528,155 documents from several sources: the Financial Times (FT), Federal Register (FR94), Foreign Broadcast Information Service (FBIS), and LA Times. Each topic consists of three sections, the 'Title', 'Description' and 'Narrative'. All three sections are used for query formulation.

As explained in section 1, we do not use complex index terms for the entire document collection. Instead, complex index terms are introduced after the first-phase retrieval. We used the results of the Okapi system from the TREC-7 conference as the first-phase retrieval output, given that Okapi was shown to be one of the best performing systems in the conference (Robertson, Walker and Beaulieu 1999).

For each query, the top 1000 documents retrieved by the Okapi system were parsed by the Apple Pie Parser. Different types of terms were then extracted and assigned term weights as described in section 2. Statistics on the extracted index terms are shown in Table 1.

From these index terms, effective index terms were selected to formulate a query

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

Table 2. *Non-interpolated average precision*

Query	Okapi	E_FC_F	E_FC_p	E_pC_F	E_pC_p
351	77.24	32.93	45.84	33.49	45.63
352	47.40	29.91	34.01	38.34	39.36
353	32.37	34.99	36.62	34.78	36.05
354	23.55	18.09	25.34	22.21	24.36
355	29.30	24.76	26.96	22.24	29.15
356	6.55	9.98	13.01	9.98	13.01
357	36.07	32.13	36.46	29.33	36.03
358	30.49	32.46	38.64	32.46	38.64
359	2.54	7.51	22.61	5.31	22.65
360	36.19	46.71	47.15	47.71	49.01
361	49.50	26.11	44.92	26.11	44.87
362	23.19	10.15	8.90	11.99	11.99
363	8.68	28.72	30.55	28.72	30.22
364	49.63	51.08	52.60	51.08	52.60
365	85.54	94.95	94.95	94.95	94.95
366	48.22	47.12	47.12	47.19	47.21
367	14.79	8.66	13.47	11.76	13.26
368	66.59	51.47	51.47	58.82	58.88
369	41.77	24.01	27.06	24.01	27.06
370	30.16	29.98	47.12	36.31	43.26
371	8.37	2.07	4.99	2.07	4.83
372	13.99	9.74	15.41	9.74	15.41
373	42.77	30.61	34.74	30.61	34.74
374	39.52	35.47	35.47	34.64	34.99
375	32.39	42.18	41.40	43.40	42.16
376	10.34	21.52	21.52	21.52	21.52
377	33.22	15.14	34.73	15.14	34.95
378	1.42	7.14	7.72	7.43	7.43
379	33.49	26.71	28.72	26.71	28.72
380	38.32	33.64	42.65	33.64	42.65
381	6.35	7.34	6.82	4.38	6.70
382	58.03	15.63	25.35	11.64	25.96
383	2.90	3.26	3.41	3.43	3.50
384	22.99	16.73	26.64	17.21	26.63
385	40.12	24.71	31.57	28.19	32.47
386	4.06	7.39	7.39	7.39	7.39
387	22.89	49.91	49.91	49.94	50.87
388	20.79	23.50	31.65	20.75	29.03
389	0.85	1.15	1.15	1.45	1.73
390	27.08	11.86	20.27	19.97	25.00
391	52.61	35.07	44.03	32.67	32.74
392	42.16	17.70	46.45	17.70	46.45
393	16.84	16.60	17.45	18.21	18.15
394	9.52	12.41	15.73	12.41	15.73
395	27.83	21.85	26.45	22.23	26.65
396	47.19	42.08	47.70	43.54	48.40
397	32.93	43.42	43.42	43.42	43.42
398	29.54	41.25	57.54	51.96	57.15
399	18.26	20.20	27.93	20.86	27.54
400	39.78	43.20	46.95	44.13	48.66
Ave.	30.33	26.42	31.80	27.26	32.00

as described in section 2. This query was used to rerank the 1000 documents to give the results of the second-phase retrieval. Reranking was done by the vector space model, i.e. based on the cosine measure between a query vector and the 1000 document vectors.

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

Table 3. *Distribution of selected terms*

Query	E_fC_f	E_fC_p	E_pC_f	E_pC_p
351	0/0/2	2/3/2	0/0/4	2/3/4
352	1/0/0	12/8/0	4/5/4	8/14/6
353	1/1/0	2/3/0	1/1/4	2/2/5
354	0/1/0	5/3/0	3/3/2	4/4/2
355	2/1/1	7/11/2	2/4/2	7/14/2
356	0/0/1	1/1/1	0/0/1	1/1/1
357	1/0/0	4/2/0	3/2/4	4/4/4
358	0/0/1	1/2/1	0/0/1	1/2/1
359	0/1/0	8/9/2	0/1/1	9/9/2
360	0/1/0	4/3/1	0/1/3	0/3/3
361	0/1/0	4/5/1	0/1/0	5/5/1
362	0/0/1	6/6/1	0/0/2	0/0/2
363	0/1/0	8/8/0	0/1/0	8/9/0
364	0/1/0	1/1/0	0/1/0	1/1/0
365	0/0/1	0/0/1	0/0/1	0/0/1
366	1/1/0	1/1/0	1/1/2	1/2/2
367	0/1/0	5/6/0	2/4/2	6/9/2
368	3/2/1	3/2/1	2/1/1	2/2/1
369	0/0/1	4/3/3	1/0/1	4/3/3
370	1/0/0	8/3/0	6/11/9	21/21/9
371	1/1/0	6/4/1	1/1/0	7/5/1
372	0/0/1	6/5/1	0/0/1	6/5/1
373	1/1/0	4/4/0	1/1/0	4/4/0
374	0/1/0	0/1/0	0/1/1	1/1/1
375	0/1/0	1/2/0	0/1/3	1/2/3
376	0/1/1	0/1/1	0/1/1	0/1/1
377	0/1/1	2/3/1	0/1/1	1/5/1
378	0/1/0	4/4/1	0/1/1	0/1/1
379	0/1/0	0/1/1	0/1/0	0/1/1
380	0/1/0	1/2/0	0/1/0	1/2/0
381	0/1/0	3/5/1	0/2/1	4/5/1
382	1/0/2	7/6/4	0/0/2	10/9/4
383	0/1/0	2/3/1	0/2/1	2/4/1
384	0/0/1	16/16/1	1/0/2	17/18/2
385	0/1/0	11/11/0	1/2/1	7/12/2
386	0/1/0	0/1/0	0/1/0	0/1/0
387	1/0/0	1/0/0	1/1/2	1/0/2
388	1/0/0	1/1/0	1/1/1	2/3/2
389	1/0/0	1/0/0	2/1/2	0/1/2
390	0/1/0	4/5/0	1/3/3	5/8/3
391	0/1/0	3/4/0	0/1/3	0/2/3
392	0/1/0	0/2/0	0/1/0	0/2/0
393	0/0/1	2/2/1	0/0/4	1/1/5
394	0/1/0	4/5/2	0/1/1	4/5/2
395	0/1/0	21/20/0	2/5/8	13/15/10
396	2/2/2	9/8/5	2/2/4	11/11/6
397	0/1/0	0/1/0	0/1/0	0/1/0
398	0/1/0	2/4/1	2/3/2	3/4/6
399	1/1/0	5/6/0	0/1/5	7/10/8
400	0/1/0	5/3/0	0/1/5	3/3/6
Ave.	0.38/0.72/0.36	4.14/4.2/0.76	0.8/1.5/1.98	3.94/5/2.52
Ratio	1/1.89/0.95	1/1.01/0.18	1/1.88/2.48	1/1.27/0.64

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

Table 4. *Distribution of index terms used in queries (E_pC_p)*

Syntactic relations			Single words with POS		
Index term	Token	Type	Index term	Token	Type
sv	52	45	Noun	244	153
vo	15	15	Proper noun	28	16
an	31	27	Verb	55	41
nn	59	39	Adjective	54	40
			Adverb	0	0

Table 2 shows the non-interpolated average precision of each combination of the index term effectiveness and the cutoff criteria. The column 'Okapi' shows the performance of the first-phase retrieval; that is, of the Okapi system. Underlined figures indicate the best performance for each query.

Table 2 shows that the cutoff criteria affects the retrieval effectiveness (average precision) more strongly than the index term effectiveness measure affects it. It also shows that introducing different types of index terms improves the performance in 32 queries out of 50. This suggests that introducing complex index terms might further improve the retrieval effectiveness of a state-of-the-art system based on conventional indexing.

Table 3 shows the number of different types of index terms used in the query. In the notation '$x/y/z$', x, y, and z signify, respectively, the number of single words without part-of-speech information (conventional index terms), single words with part-of-speech information, and syntactic relation-based index terms. The last row denotes the ratio of syntactic relation-based index terms and single-word index terms with part-of-speech information to the conventional index terms. From this table, we can see that precision- based cutoff (C_p) tends to select more index terms than F-measure-based cutoff (C_F).

Comparing index term effectiveness measures, precision-based effectiveness (E_p) tends to select more complex index terms (syntactic relations and single words with part-of-speech information) than F-measure-based effectiveness (E_F). This tendency is reasonable because complex index terms might improve precision but degrade recall. Note that the F-measure takes recall into account as well.

Table 4 summarizes the total number of different types of complex index terms in queries constructed from the E_pC_p combination. Comparing Table 4 with Table 1 shows that noun phrases (an, nn-relations) tend to be selected as effective index terms. This result provides experimental support for previous methods in which noun phrases are singled out as complex index terms (Mitra *et al.* 1997; Arampatzis, Tsoris, Koster and van der Weide 1998).

These results indicate there is an upper bound on improvement, since we used relevance judgment information to formulate optimal queries that included different types of index terms. In the next section, we introduce a method to select effective index terms without referring to relevance judgment information.

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

4 Index term selection

As shown in the previous section, optimal queries can be constructed by referring to the relevance judgment of each query. However, relevance judgments are not available in real-world settings. We need some criterion to select effective index terms to revise the original query. For this purpose, we adopt a decision tree to judge if an index term should be included in the revised query.

The decision tree is constructed based on three features of each index term: the type of the index term, the position of the index term in the original query, and the weight of the index term. As described in section 2, we introduce three types of index terms: syntactic relations, single words with part-of-speech information, and conventional single-word index terms. The first feature distinguishes between these three types.

In the TREC data collection used in our experiments, a topic (query) consists of three sections, the 'Title', 'Description' and 'Narrative'. The second feature is used to distinguish between these three sections.

The third feature takes a term weight as its value. Term weights are calculated as described in section 2 depending on the type of index term.

Given a training data set consisting of a set of pairs of a query and an optimal index term set calculated as described in section 3, a decision tree is constructed. The decision tree takes a query and classifies each index term in the query into the classes 'use' and 'do not use'.

To explore the effectiveness of the index term selection method described above, we conducted experiments using the TREC-7 information retrieval test collection (Voorhees and Harman 1999), the same data used in the experiments discussed in the previous section. The Okapi system results from the TREC-7 conference were again used as the first-phase retrieval output.

For each query, the top 1000 documents retrieved by the Okapi system were parsed by the Apple Pie Parser, and different types of terms extracted and assigned term weights as described in section 2.

As described in section 3, an optimal index term set was selected for each query by referring to a relevance judgment over that query. In this experiment, the E_pC_p combination of effectiveness and cutoff criteria was used since this combination worked best in the previous experiments. Index terms included in this optimal set were marked 'use' and other index terms were marked 'do not use'. Since the frequency of 'use' instances was lower than that for 'do not use', we adopted the *boosting* technique when constructing a decision tree, that is, each occurrence of a 'use' instance was duplicated seven times. In preliminary experiments, we tried several boosting factors, ranging from 1–10, and settled on a value of 7 because of its superior performance.

This training data was fed into the C4.5 system (Quinlan 1993) to construct a decision tree for each query on the basis of the *one out of n* method. That is, one query was held out for testing and the remaining 49 queries were used as training data to construct the decision tree for the held out query. We obtained one decision tree to test each query out of a total of 50 decision trees. The constructed decision

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

Table 5. *Result of reranking after index term selection*

Query	Okapi prec.	Optimal query prec.	Optimal query term dist.	Revised query prec.	Revised query term dist.
351	77.24	45.63	2/3/4	34.96	7/8/7
352	47.40	39.36	8/14/6	36.34	9/17/14
353	32.37	36.05	2/2/5	34.71	4/4/9
354	23.55	24.36	4/4/2	24.40	4/4/3
355	29.30	29.15	7/14/2	24.23	9/15/9
356	6.55	13.01	1/1/1	10.64	4/5/2
357	36.07	36.03	4/4/4	32.93	5/11/5
358	30.49	38.64	1/2/1	31.13	2/4/4
359	2.54	22.65	9/9/2	18.21	10/9/9
360	36.19	49.01	0/3/3	33.51	4/4/8
361	49.50	44.87	5/5/1	38.93	7/7/7
362	23.19	11.99	0/0/2	7.94	0/1/6
363	8.68	30.22	8/9/0	29.04	11/11/7
364	49.63	52.60	1/1/0	51.82	2/2/1
365	85.54	94.95	0/0/1	64.37	2/2/1
366	48.22	47.21	1/2/2	34.04	2/4/5
367	14.79	13.26	6/9/2	12.04	6/9/7
368	66.59	58.88	2/2/1	46.89	6/4/6
369	41.77	27.06	4/3/3	21.24	5/4/6
370	30.16	43.26	21/21/9	38.75	21/31/24
371	8.37	4.83	7/5/1	1.55	7/4/6
372	13.99	15.41	6/5/1	15.39	8/7/3
373	42.77	34.74	4/4/0	32.04	4/7/5
374	39.52	34.99	1/1/1	34.35	1/1/5
375	32.39	42.16	1/2/3	39.86	1/3/9
376	10.34	21.52	0/1/1	17.00	0/2/1
377	33.22	34.95	1/5/1	34.78	1/7/3
378	1.42	7.43	0/1/1	3.84	1/3/3
379	33.49	28.72	0/1/1	20.11	1/3/3
380	38.32	42.65	1/2/0	42.60	1/3/1
381	6.35	6.70	4/5/1	6.53	3/5/1
382	58.03	25.96	10/9/4	26.51	10/11/9
383	2.90	3.50	2/4/1	3.36	1/3/5
384	22.99	26.63	17/18/2	25.96	18/21/6
385	40.12	32.47	7/12/2	29.08	11/14/5
386	4.06	7.39	0/1/0	1.17	2/2/2
387	22.89	50.87	1/0/2	17.51	1/0/2
388	20.79	29.03	2/3/2	22.65	2/8/5
389	0.85	1.73	0/1/2	0.62	1/4/9
390	27.08	25.00	5/8/3	26.64	6/8/5
391	52.61	32.74	0/2/3	24.37	0/3/7
392	42.16	46.45	0/2/0	46.22	0/2/1
393	16.84	18.15	1/1/5	17.63	3/1/4
394	9.52	15.73	4/5/2	11.94	5/8/5
395	27.83	26.65	13/15/10	26.39	13/18/16
396	47.19	48.40	11/11/6	48.09	11/11/7
397	32.93	43.42	0/1/0	10.39	2/6/6
398	29.54	57.15	3/4/6	54.72	5/6/8
399	18.26	27.54	7/10/8	27.34	8/16/11
400	39.78	48.66	3/3/6	44.88	4/4/10
Ave.	30.33	32.00	3.9/5/2.5	26.79	5.0/6.9/6.1
#Imp.		31		25	

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

trees tended to test features in the order of index term types, positions, and then term weights.

Each decision tree was used to select index terms to define a revised query, which was then used to rerank the 1000 documents to give the output of the second-phase retrieval. Reranking was done based on the vector space mode.

Table 5 shows the non-interpolated average precision of Okapi, reranked according to both the optimal queries and the decision-tree-based revised queries. The table also shows the number of different types of index terms used in the final query. In the notation '$x/y/z$', x, y, and z again signify, respectively, the number of single words without part-of-speech information (conventional index terms), single words with part-of-speech information, and syntactic relation-based index terms. Underlined figures indicate cases in which the performance was improved by reranking. The last row shows the number of cases in which retrieval performance was improved by reranking.

Table 5 shows that introducing different types of index terms improved the performance in 31 queries out of 50 when using optimal queries. And queries automatically constructed with decision trees produced improvement in 25 out of these 31 cases. Underlined figures indicate that the performance was improved compared to the baseline (Okapi). Note that when constructing these queries, the relevance judgments were not referred to this time. However, overall performance fell by 10%. Future research is thus needed to identify more effective features that can be used to construct a decision tree.

We further investigated the six cases in which no improvement was achieved (queries 360, 365, 386, 387, 389 and 397) despite it being possible with optimal queries. This revealed that it is rather difficult for humans to think of good index terms for these queries. In addition, in some cases adding less useful index terms degraded performance. For instance, in query 365 concerning the *El Nino* phenomenon, 'El Nino' is the decisive index term and adding any other index term always degraded performance. This suggests that deleting less useful index terms is important in retrieval that requires high precision.

In this paper, we focused on selecting effective index terms from among those which can be extracted from a query. This can be considered a form of local selection, since we used information only in the query when selecting index terms. Mitra *et al.* claim that complex index terms are useful when the results of conventional retrieval are 'moderate' (Mitra *et al.* 1997). This suggests another possible approach when introducing complex index terms based on global context – merging the highly ranked documents from a conventional method and the moderately ranked documents from the proposed method. Such an approach is often called 'data fusion'.

We define 'moderate rank' as follows. For two ranked document lists given for each query, one from the Okapi output and the other from our output, the rank shift of each relevant document is calculated. For example, when a relevant document is ranked 1st by Okapi and 10th by our method, the rank shift becomes −9. These rank shifts are accumulated from documents which are ranked highly by Okapi, and 'moderate rank' is defined as that which maximizes the accumulation. The merged

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

Table 6. *Result of merging two ranking lists*

Query	Okapi	moderate rank	Merged
351	77.24	140	77.36
352	47.40	367	47.59
353	32.37	83	34.14
354	23.55	283	24.66
355	29.30	78	33.58
356	6.55	187	6.86
357	36.07	270	36.26
358	30.49	371	30.44
359	2.54	187	3.54
360	36.19	355	36.08
361	49.50	15	52.64
362	23.19	444	23.17
363	8.68	15	18.32
364	49.63	72	50.31
365	85.54	59	85.54
366	48.22	225	48.22
367	14.79	354	14.84
368	66.59	187	68.07
369	41.77	147	41.77
370	30.16	359	33.36
371	8.37	169	8.56
372	13.99	77	16.32
373	42.77	751	42.77
374	39.52	401	39.71
375	32.39	212	34.53
376	10.34	175	11.76
377	33.22	162	32.78
378	1.42	322	1.76
379	33.49	39	35.52
380	38.32	27	39.92
381	6.35	188	6.43
382	58.03	183	58.01
383	2.90	324	3.10
384	22.99	94	24.18
385	40.12	329	40.83
386	4.06	296	4.06
387	22.89	183	24.39
388	20.79	272	21.22
389	0.85	595	0.84
390	27.08	271	27.36
391	52.61	400	53.20
392	42.16	285	43.21
393	16.84	518	16.88
394	9.52	528	9.55
395	27.83	361	29.00
396	47.19	52	57.56
397	32.93	227	33.24
398	29.54	175	36.50
399	18.26	24	24.17
400	39.78	419	40.41
Ave.	30.33	245	31.69

ranking is constructed by joining documents ranked high to moderate by Okapi and all those documents from our method not already contained in the Okapi ranking.

Table 6 shows the result of such merging. The third column shows 'moderate ranks' calculated as described above. We obtained an improvement over the results

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

of Okapi, but the performance is slightly below that with the optimal queries (Table 5). This result supports Mitra's suggestion. Note that we referred to the relevance judgment information here, as we did in section 3. Criteria to determine the 'moderate rank' without referring to the relevance judgment information is required for practical applications.

5 Concluding remarks

This paper introduced an index term selection method that uses a decision tree in the context of a two-phase retrieval framework. Effective index terms are selected from among different types of index terms – that is, syntactic relations – and single words with or without part-of-speech information. Experiments using the TREC-7 test collection showed that the retrieval effectiveness was improved in 25 out of 50 queries. Overall performance, however, was degraded. We conducted experiments whereby we merged retrieval results, one set from a conventional method and one from our proposed method, and thus improved retrieval performance. Further investigation is required to determine appropriate merging criteria.

Future research issues include the need for methods to find more effective features when constructing decision trees, ways to combine index term selection with relevance feedback techniques, and ways to take into account the relation between complex index terms and their component single-word terms.

References

Arampatzis, A. T., Tsoris, T., Koster, C. H. A., and van der Weide, T. P. (1998) Phrase-based information retrieval. *Infor. Process. & Manage.* **34**(6): 693–707.

Keen, E. M. and Hartley, R. J. (1994) Phrase processing in text retrieval. *J. Document & Text Manage.* **2**(1): 23–34.

Kwok, K. L. and Chan, M. (1998) Improving two-stage ad-hoc retrieval for short queries. *Proceedings Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 250–256.

Marcus, M. P., Santorini, B. and Marcinkiewicz, M. A. (1993) Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics* **19**(2): 313–330.

Metzler, D. P. and Haas, S. W. (1989) The constituent object parser: Syntactic structure matching for information retrieval. *Proceedings Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 117–126.

Mitra, M., Buckley, C., Singhal, A. and Cardie, C. (1997) An analysis of statistical and syntactic phrases. *Proceedings RIAO '97*, pp. 200–214.

Quinlan, J. (1993) *C4.5 : Programs for Machine Learning*. Morgan Kaufmann.

Robertson, S. E., Walker, S. and Beaulieu, M. (1999) Okapi at TREC-7: automatic ad hoc, filtering, VLC and interactive track. *Proceedings 7th Text REtrieval Conference*, pp. 253–264. NIST Special Publication, SP 500-242.

Salton, G. (1988) *Automatic Text Processing*. Addison-Wesley.

Sekine, S. and Grishman, R. (1995) A corpus-based probabilistic grammar with only two non-terminals. *Proceedings International Workshop on Parsing Technologies.*

Sparck Jones, K. (1972) A statistical interpretation of term specificity and its application in retrieval. *J. Documentation* **28**(1): 11–21.

Strzalkowski, T. (1995) Natural language information retrieval. *Infor. Process. & Manage.* **31**(3): 397–417.

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press

van Rijsbergen, C. J. (1979) *Information Retrieval*, 2nd ed. Butterworths.
Voorhees, E. M. (1999) Natural language processing and information processing. In: Pazienza, M. T., editor, *Information Extraction*, pp. 32–48. Springer.
Voorhees, E. M. and Harman, D. (1999) Overview of the seventh text retrieval conference (TREC-7). *Proceedings 7th Text REtrieval Conference*, pp. 1–23. NIST Special Publication, SP 500-242.

https://doi.org/10.1017/S1351324902002899 Published online by Cambridge University Press