Structuring and expanding queries in the probabilistic model

OGAWA Yasushi, MANO Hiroko, NARITA Masumi, HONMA Sakiko
Software Research Center, RICOH Co., Ltd.
1-1-17 Koishikawa, Bunkyo-ku, Tokyo 112-0002, JAPAN
{yogawa,mano,narita,honma}@src.ricoh.co.jp

1 Introduction

This is our second participation in TREC, following the last year's ad-hoc, and five runs were submitted for the main web track.

Our system is based on our Japanese text retrieval system [3], to which English tokenizer/stemmer has been added to process English text. Our indexing system stores term positions, thus providing proximity-based search, in which the user can specify the distance between query terms.

What our system does is outlined as follows:

1. Query construction

   The query constructor accepts each topic, extracts words in each of the appropriate fields and constructs a query to be supplied to the ranking system.

2. Initial retrieval

   The constucted query is fed into the ranking system, which then assigns term weights to query terms, scores each document and turns up a set of top-ranked documents assumed to be relevant to the topic (pseudo-relevant documents).

3. Query expansion

   The query expander collects and ranks the words in the pseudo-relevant documents and the words ranked the highest are added to the original query, with the words already in the query re-assigned new term weights.

4. Final retrieval

   The ranking system performs final retrieval using the modified query.

The basic features of the system are mostly the same as those implemented last year for the TREC-8 ad-hoc track [2]. In what follows, we explain each of the steps in more detail, both the features retained from last year and new enhancements we added this year for the TREC-9 main web track.

2 Query construction

We have employed automatic query processing to construct queries using single-word and phrasal search terms. In what follows, we describe how single and phrasal search terms are created by linguistic methods and represented as a structured query.

2.1 Basic features

2.1.1 Single term selection

Natural language text in each topic is processed by our English tokenizer and stemmer to output stemmed word tokens.[1] From the stemmed tokens, the query constructor selects single-word search terms by eliminating stopwords. We have used two kinds of stopword lists, the Fox's [1] word list for the <title> field and its augmented word list we created for the <desc> field.

2.1.2 Phrasal term selection

Noun phrases consisting of two or more single words are extracted for use in search terms. Syntactic phrases are recognized in the natural language text by applying the syntactic chunker LT_CHUNK developed at the Edinburgh Language Technology Group. Each noun phrase is then tokenized and stemmed. Phrases consisting of three or more single words are decomposed into sets of word pairs. For example, the noun phrase "industrial waste disposal" is decomposed to derive three word pairs "industrial waste," "waste disposal," and "industrial disposal."

2.1.3 Query representation

Single and phrasal search terms are combined into a query using syntax of our query language. Phrasal terms are represented using a proximity operator #WINDOW. For example, the phrasal search term "waste disposal" is represented as:

```
#WINDOW[1,1,o](waste,disposal)
```

where #WINDOW[1,1,o] specifies that the two component words are to be found adjacent and in the described order in a document. To deal with possible changes in word order and the number of intervening words between the component words of a phrasal term, we prepared a variant of the basic query representation, #WINDOW[2,*num*,u](A,B). The variant allows the words A and B to co-occur not in adjacency but within a specified number of words (*num*) of each other, in any order. We have also introduced a scaling operator #SCALE to phrasal term representation to adjust its term weight.

To sum, our sample queries are expressed as follows:

```
A query from the <title> and <desc> fields:
     #OR(killer,bee,attack,human,africanise,
         #SCALE[0.4](#WINDOW[1,1,o](killer,bee)),
         #SCALE[0.25](#WINDOW[2,50,u](killer,bee)))
```

[1] Text was used as is, with no spelling-correction applied.

2.2 Multiple fields

In TREC-8, the fields from which a word or phrase was extracted were not taken into account when a query is constructed. In TREC-9, when words are repeated across multiple fields in the topic, we increase their term weights to reflect their relative importance. To adjust the weights of these repeated words, the scaling operator #SCALE is applied and optimized to maximize the retrieval effectiveness.

```
Words extracted from the topic:
<title>: lava, lamp
<desc>: origin, operation, lava, lamp

Query:
#OR(origin,operation,#SCALE[3.0](lava),#SCALE[3.0](lamp))
```

3 Initial retrieval

For each query constructed, the ranking system ranks the documents in the target document collection and retrieves top-ranked documents. To rank documents, the system uses term weighting and document scoring formulae similar to Okapi's [4] but with some modifications.

The weight of each term is calculated by using the formula

$$w_t = \log\left(k_4' \cdot \frac{N}{n} + 1\right),$$

where N is the number of documents in the collection, n is the number of the documents in which the term occurs and k_4' is a parameter, with $0 \leq k_4'$.

Note that unlike Okaipi's, with our formula, the term weights never get negative. By keeping the term weights positive, the quality of retrieval is maintained even in the worst case.

With each term weighted according to the above formula, the ranking score for each document is calculated using the formula

$$s_{d,q} = \sum_{t \in q} \frac{w_t}{\log\left(k_4' \cdot N + 1\right)} \cdot \frac{f_{t,d}}{k_1((1-b) + b\frac{l_d}{l_{ave}}) + f_{t,d}}$$

where $f_{t,d}$ is the within-document frequency of the term, l_d is the document length, l_{ave} is the average document length, k_1 and b are parameters.

4 Query expansion

4.1 Basic features

The query expander, regarding the top-ranked documents as relevant documents, collects all single terms except stopwords in them and ranks the terms according to its Term Selection Value (TSV) while reweighting query terms, using formulae similar to Okapi's.

For each term collected, a new weight based on the feedback from the retrieved documents is assigned. The term reweighting formula reflects term weighting during initial retrieval mentioned above.

$$\begin{aligned} w_t &= \frac{k_5}{k_5+\sqrt{R}} \log\left(k_4' \frac{N}{N-n} + \frac{n}{N-n}\right) + \frac{\sqrt{R}}{k_5+\sqrt{R}} \log \frac{r+0.5}{R-r+0.5} \\ &\quad - \frac{k_6}{k_6+\sqrt{S}} \log \frac{n}{N-n} - \frac{\sqrt{S}}{k_6+\sqrt{S}} \log \frac{s+0.5}{S-s+0.5}, \end{aligned}$$

where R is the number of relevant documents, r is the number of relevant documents containing the term, S is the number of non-relevant documents, s is the number of non-relevant documents containing the term, and k_5 and k_6 are parameters. (S was set to 0 in the experiment.)

The query expander then calculates TSV for each reweighted term to select the terms to augument the query by using the formula

$$TSV = \left(\sum_{d \in R} \frac{f_{t,d}}{k_1((1-b)+b\frac{l_d}{l_{ave}})+f_{t,d}} \Big/ R - \beta \cdot \sum_{d \in S} \frac{f_{t,d}}{k_1((1-b)+b\frac{l_d}{l_{ave}})+f_{t,d}} \Big/ S \right) \cdot w_t$$

where β is a parameter.

Note the TSV formula has been changed from Okapi's to incorporate the within-document frequency.

The top-ranked single terms are then added to the original query with their respective term weights. The single terms and phrasal terms origianlly included in the query are also given re-assigned term weights, multiplied with a bonus factor.

4.2 Duplicates-resistant term selection

Compared with the ad-hoc track document collection we used last year, the web track document collection used this year seemed to contain far more documents that are exact or partial copies of some other documents. Although their presence in the retrieved documents may not affect retrieval effectiveness as measured by the current evaluation measures [6], it can degrade performance when the retrieved documents are used for automatic query expansion since this could give terms that appear in duplicates or near-duplicates an unjustifiably higher document frequency, thus boosting the likelihood of being selected.

To alleviate ill effects of duplicates and near-duplicates in the documents retrieved in initial retrieval, two work-arounds are devised:

1. During initial retrieval, eliminate documents scored the same as previously retrieved documents. If two documents have the same score, it is highly likely that they are exact copies of each other.

2. When selecting terms for expansion, for terms with a low document frequency among the retrieved documents, terms that appear in the same set of documents as those from which previously selected terms were drawn are excluded. These terms

are unlikely to co-occur in the same set of documents, and if they do, that may indicate the set shares the same piece of text.

4.3 Phrasal term addition

This year, we extend the idea of using phrasal terms from only in query generation to both in query generation and expansion. That is, when expanding queries, not only single words but also pairs of contiguous words in top-ranked documents are collected, evaluated and selected for use as expansion terms. The goal was to find a way to select the right pairs of words, with the right balance of weights, while keeping the expansion process simple and quick with minimal overhead.

In the experiments, we tested the same weighting/selection technique as used for single words for its applicability to word pairs and found it promising when the following adjustments were made:

- Set first the minimum Term Selection Value, which is set several times as high as that for single words.
- Give more importance to the document frequency in top-ranked documents compared with that in the document collection, when assigning a weight.
- Adjust the weights for word pairs that contain the same single words as those selected and those supplied in the original query.

5 Final retrieval

The expanded-and-reweighted query is sent to the ranking system and documents are retrieved as final result. Document ranking is done just as in initial retrieval, except that the term weights are supplied in the query.

6 Results

We tuned up the formulae using the WT2g document collection and mainly queries generated from the TREC-8 topics. Parameter values were chosen for each of the four categories below and are listed in Table 1 – 3.

- Queries using only <title> and queries using <title> and <desc>
- Queries using no phrasal search terms and queries using phrasal search terms

Note that in Table 1 and Table 2, we chose different sets of parameter values for the same retrieval parameters. This is because, for retrieval in a run with no query expansion, we wanted parameter values that would maximize average precision, whereas for initial retrieval for a run with query expansion, we looked for parameter values that would maximize precision at ten retrieved documents.

Table 1: Parameters for runs without expansion

	title only		title + desc	
	no phrases	phrases	no phrases	phrases
k_1	0.75	0.5	1.0	0.75
b	0.25	0.25	0.25	0.25
k_4'	0.5	0.5	0.5	0.5
Scale for multi-field query terms	–	–	3.0	3.0
num in #WINDOW[2,*num*,u]	–	50	–	10
Scale for #WINDOW[1,1,o]	–	0.25	–	0.4
Scale for #WINDOW[2,*num*,u]	–	0.1	–	0.25

Table 2: Parameters for runs with expansion (initial retrieval)

	title only		title + desc	
	no phrases	phrases	no phrases	phrases
k_1	1.5	0.75	1.5	0.75
b	0.25	0.25	0.25	0.25
k_4'	0.5	0.1	0.5	0.5
Scale for multi-field query terms	–	–	2.75	3.0
num in #WINDOW[2,*num*,u]	–	50	–	10
Scale for #WINDOW[1,1,o]	–	0.25	–	0.2
Scale for #WINDOW[2,*num*,u]	–	0.1	–	0.1

Table 3: Parameters for runs with expansion (final retrieval)

	title only		title + desc	
	no phrases	phrases	no phrases	phrases
Maximum number of documents used for expansion	10	10	10	10
k_1	0.75	0.75	1.0	0.75
b	0.25	0.25	0.25	0.25
k_4'	0.01	0.01	0.01	0.01
k_5 for single expansion terms	6.0	6.0	6.0	6.0
k_5 for phrasal expansion terms	2.0	3.0	3.0	3.0
Scale for multi-field query terms	–	–	3.0	3.0
num in #WINDOW[2,*num*,u]	–	50	–	10
Scale for #WINDOW[1,1,o]	–	0.4	–	0.3
Scale for #WINDOW[2,*num*,u]	–	0.25	–	0.2
Maximum number of terms to be added	25	25	30	30
Minimum number of terms to be added	10	10	10	10
Minimum r for term to qualify	2	2	2	2
Maximum r for near-duplicate checking	3	3	3	3
Scale for phrasal expansion terms	0.3	0.2	0.3	0.3
Minimum TSV factor for phrasal expansion terms	5	3.3	2.5	2.5
Bonus factor for query terms	10.0	10.0	7.0	5.0

Table 4: Average precision for TREC-8 topics

	title only		title + desc	
	no phrases	phrases	no phrases	phrases
baseline	0.3184	0.3247	0.3017	0.3113
above + multi-field scaling	–	–	0.3321	0.3420
above + expansion	0.3493	0.3536	0.3637	0.3703
above + phrasal expansion terms	0.3538	0.3533	0.3671	0.3716

The experimental runs using the above parameters, the TREC-8 topics and the WT2g document collection resulted in the following average precision measurements (Table 4). The table shows improvement in performance as more features are added.

Using the same conditions as above, the results for the TREC-9 topics and the WT10g document colletion are shown in Table 5. The runs submitted are indicated by asterisks.[2] (The numbers in the parentheses below the table are from the original submissions, which contained incorrect results due to program bugs. The numbers in the table were obtained after bug fixes and a minor modification in which the minimum number of words to be added was lowered to 0.)

[2]ric9dpxL is a linear combination of ric9dpx and HITS [5]. This run is yet to be analyzed.

Table 5: Average precision for TREC-9 topics

	title only		title + desc	
	no phrases	phrases	no phrases	phrases
baseline	0.2025	0.2073	0.2135	0.2407
above + multi-field scaling	–	–	0.2489	0.2608*3
above + expansion	0.1740	0.2021	0.2212*2	0.2427
above + phrasal expansion terms	0.1788	0.2034*1	0.2211	0.2411*4

*1: ric9tpx (0.1787), *2 ric9dsx (0.2201), *3: ric9dpn (0.2616), *4: ric9dpx (0.2267)

The results show that use of phrasal search terms and multi-field scaling worked well in TREC-9, as in TREC-8 above. However, unlike in TREC-8, we see that query expansion, whether with phrasal expansion terms or not, made unexpectedly negative effect in TREC-9. Why the difference?

One thing we noticed is the difference in the size of the target collection from which documents were retrieved. The WT10g document collection used for the TREC-9 submission had more than six times the number of documents in the WT2g document collection used for the TREC-8 experiments. The largeness of the WT10g collection led to rare words getting term weights that were extremely high because of the way we calculated the term weights, thus making it easier for these words to be selected as expansion terms.

Another difference is in the number of relevant documents for each of the topics. Although on the average, the number of relevant documents in TREC-9 is greater than that in TREC-8, there are more topics having a very few relevant documents in TREC-9 than in TREC-8; topics having fewer than 10 relevant documents add up to 11 in TREC-9, as opposed to only 2 in TREC-8. The system, on the other hand, retrieved just as many documents for these topics despite this, turning up fewer relevant documents in the top-ranked documents in initial retrieval before expansion. For example, P@10, or precision after 10 documents are retrieved, is 0.3520 in TREC-9, compared to 0.5000 in TREC-8, in runs in "title + desc, phrases." What this means is that the query expander had to select expansion terms from mostly non-relevant documents, which in addition may contain a near-duplicate or two skewing the term selection statistics even further in the wrong direction.

In the follow-up experiments conducted after the submission, we tested an alternative approach where expansion terms were selected without regard to their term weights so that their influence on term selection would be eliminated. The average precision we obtained from this approach in a run in "title + desc, phrases," was 0.2629, showing an 8% improvement from the submitted run.

References

[1] C. Fox. A stop list for general text. *ACM SIGIR Forum*, Vol. 24, No. 2, pp. 19–35, 1991.

[2] Y. Ogawa, H. Mano, M. Narita, and S. Honma. Structuring and expanding queries in the probabilistic model. In *The Eighth Text REtrieval Conference (TREC-8)*, 2000.

[3] Y. Ogawa and T. Matsuda. An efficient document retrieval method using n-gram indexing (in Japanese). *Transactions of IEICE*, Vol. J82-D-I, No. 1, pp. 121–129, 1999.

[4] S.E. Robertson and S. Walker. On relevance weights with little relevance information. In *Proc. of 20th ACM SIGIR Conf.*, pp. 16–24, 1997.

[5] M. Toyoda and M. Kitsuregawa. Finding related communities on the Web. In *Poster in WWW-9 Conf.*, 1999.

[6] D. Hawking, E. Voorhees, N. Craswell, and P. Bailey. Overview of the TREC-8 Web Track. In *The Eighth Text REtrieval Conference (TREC-8)*, 2000.