Query Substitution based on N-gram Analysis

Xiaobing Xue, Van Dang and W. Bruce Croft
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts, Amherst, MA, 01003, USA
{xuexb,vdang,croft}@cs.umass.edu

ABSTRACT

Query substitution replaces original query words with a new words that express the same meaning. In this paper, the technique of n-gram analysis is proposed to find the synonyms or quasi-synonyms of the original query word. The synonyms found are then incorporated into the original query with different methods. Experiments show that the proposed n-gram analysis techniques can obtain interesting synonyms, which help to improve retrieval effectiveness.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Experimentation, Performance

Keywords

Query Substitution, Query Reformulation, N-gram Analysis, Synonym Finding, Information Retrieval

1. INTRODUCTION

Query transformation techniques are designed to improve retrieval effectiveness by capturing alternative ways of expressing the same concepts. Query expansion techniques, such as relevance feedback or pseudo-relevance feedback, add words to the query that are generally related to the query topic. Query substitution techniques are very similar but attempt to identify words that are related to specific query words, rather than the general topic. As an example, "aeroplane" would be a possible substitute word for "airplane" in most queries. Stemming is a special case of query substitution where the related words are restricted to morphological variants. Query substitution and query expansion are different aspects of query transformation and, as such, may be complementary in terms of effectiveness.

SIGIR'09, July 19–23, 2009, Boston.

Previous research on query substitution has focused mainly on query logs [1, 3]. In this paper, a more general method based on n-gram analysis is proposed to find the "quasi-synonyms" or substitute words. Specifically, we assume that if two words have many common n-gram contexts in a large collection of text, they will have a high probability of being quasi-synonyms. Several methods are proposed to transform the original query using the synonyms to reduce the vocabulary mismatch between the query and the document. N-gram based techniques have been used for query segmentation [2], but the approach used is somewhat different from that presented here.

2. WORD SUBSTITUTION BASED ON N-GRAM ANALYSIS

An n-gram is a sequence of words that appear in the document. After scanning the whole collection, it is easy to obtain a set of n-grams $S = \{ngram_1, ngram_2, ..., ngram_m\}$, where $ngram_i = w_1 w_2 ... w_n$. Here, w_j represents a word. An n-gram context is defined as the neighborhood word pattern in an n-gram. Specifically, w_j's n-gram context c in $ngram_i$ is defined as $c = w_1 w_2 ... w_{j-1} \# w_{j+1} ... w_n$. # indicates the position where w_j appears. For example, in a five-gram 'buy up to 100 airbus', the context of the word 'airbus' is 'buy up to 100 #'. Based on the common contexts of two words, the substitution probability is defined as follows.

$$P(w|t) = \frac{P(w,t)}{P(t)} = \frac{\sum_c P(w,t,c)}{P(t)} \quad (1)$$

$$P(w,t,c) = \sum_c P(w|c)P(t|c)P(c) \quad (2)$$

Here, $P(t)$ is a normalizer. $P(w|c)$ is estimated by dividing the frequency that the word w appears in the context c by the frequency of the context c. $P(c)$ is the prior probability of the context, which measures the importance of a context in deciding the substitution relations of two words. We estimate $P(c)$ by dividing 1 by the number of unique words appearing in the context c, which penalizes common contexts such as 'a #' and 'the #'.

3. TRANSFORMING THE QUERY

Several methods have been considered for adding the synonyms obtained into the original query Q.

First, we use the weighted synonym operator '#wsyn' of the Indri[1] query language, which combines a set of synonyms

[1]http://www.lemurproject.org/indri/

Table 1: Indri query after adding synonyms. λ is taken as 0.5.

WSyn	#weight(0.5 #combine(rail strike) 0.5 #combine(#wsyn(1 rail 0.34 railroad 0.09 railway) #wsyn(1 strike 0.09 walkout 0.08 protest)))
QGen1	#weight(0.5 #combine(rail strike) 0.5 #weight(0.34 #combine(railroad strike) 0.09 #combine(railway strike) 0.09 #combine(rail walkout) 0.08 #combine(rail protest)))

with different weights. These weights can be assigned by the substitution probability. This method is denoted as WSyn.

Second, we consider generating a set of new queries Q' based on the generation probability $P(Q'|Q)$, which can be further used as the weights of the generated queries. $P(Q'|Q)$ can be directly estimated by the substitution probability $P(q'|q)$, i.e. the probability of substituting the original query word q with q'. This method is denoted as QGen1. It is important to consider the compatibility or context of q' with regard to the other query words in Q, thus QGen2 estimates $P(Q'|Q)$ by multiplying the substitution probability with the compatibility of Q', which is estimated based on the probability that q' co-occurs with other query words.

The final query is a linear combination of the original query and the reformulated one with the parameter λ. An example is provided here. Given the query 'rail strike', the synonyms found for the words 'rail' and 'strike' are 'railroad (0.34) railway(0.09)' and 'walkout(0.09) protest(0.08)', respectively. The scores in parentheses denote the substitution probabilities. The Indri queries generated by WSyn and QGen1 using these synonyms are shown in Table 1. The Indri query of QGen2 has the same form as QGen1, but the weights are different.

4. EXPERIMENTS

Experiments were conducted on the TREC AP and WSJ collections. AP contains 242,918 documents and WSJ contains 173,252 documents. For each collection, all n-grams with n up to 5 were extracted. The word substitution probabilities were estimated based on the extracted n-grams. 100 queries were used to test the retrieval performance on each collection: 50 queries were used for training and the rest were used for testing. Mean average precision (MAP) and precision at 10 (P@10) were used as the performance measures. The query likelihood language model (LM) and relevance model (RM) ranking algorithms were the baselines.

Table 2 shows query word substitutions generated by the n-gram analysis technique and we compare them with expansion words generated by the pseudo-relevance feedback technique RM. Table 2 clearly shows that our query substitution method tends to find synonyms, while RM tends to find words that co-occur with the whole query.

The retrieval performance of WSyn, QGen1 and QGen2 on AP and WSJ are shown in Table 3. These results show that WSyn, QGen1 and QGen2 perform better than LM, which verifies the benefit of the proposed query substitution methods. Among the proposed query substitution methods, WSyn and QGen2 perform better than QGen1. Also, WSyn, QGen1 and QGen2 all perform worse than RM. The main reason appears to be that, for these collections, the number of queries that can benefit from expansion is larger than the number of queries that can benefit from substitution. However, since the proposed query substitution methods and expansion methods such as RM improve different aspects of the queries, they are potentially complementary.

Table 2: Examples of query substitution. Each column shows the top 5 words with highest substitution (expansion) probability.

Query Substitution		Query Expansion
rail	strike	rail strike
railroad	walkout	union
train	attack	pay
railway	protest	percent
air	demonstrate	train
bus	raid	service
airbus	subsidy	airbus subsidy
plane	support	aircraft
jetliner	pay	govern
jet	program	germany
boee	aid	europe
airline	fund	west

Table 3: Retrieval Performance of Query Substitution methods

		LM	RM	WSyn	QGen1	QGen2
AP	MAP	0.200	0.248	0.205	0.204	0.226
	P@10	0.424	0.446	0.446	0.436	0.458
WSJ	MAP	0.303	0.328	0.315	0.307	0.310
	P@10	0.448	0.476	0.460	0.446	0.460

5. CONCLUSION

The n-gram analysis techniques are proposed in this paper to find quasi-synonyms for query word substitution. Different methods of transforming the query to include the synonyms were evaluated. Experiments showed that the proposed n-gram analysis techniques can effectively find quasi-synonyms for query words and, after combining the original query with the synonyms, retrieval effectiveness can be improved. The next step will be to explore how substitution and expansion techniques can be used together to further improve effectiveness.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval, in part by NSF grant #IIS-0711348, in part by NSF CLUE IIS-0844226, and in part by NSF grant #IIS-0534383, and in part by Yahoo!. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

6. REFERENCES

[1] Jones, R., Rey, B., Madani, O., Greiner, W.: Generating query substitutions. In: WWW'06, Einburgh, Scotland, 2006, 387-396.

[2] Tan, B., Peng, F.: Unsupervised query segmentation using generative languge models and wikipedia. In: WWW '08, Beijing, China,2008, 347–356
[3] Wang, X., Zhai, C.: Mining term association patterns from search logs for effective query reformulation. In: CIKM'08, Napa Valley, CA, 2008, 479-488.