Modeling Reformulation Using Passage Analysis

Xiaobing Xue W. Bruce Croft David A. Smith
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts, Amherst, MA, 01003, USA
{xuexb,croft,dasmith}@cs.umass.edu

ABSTRACT

Query reformulation modifies the original query with the aim of better matching the vocabulary of the relevant documents, and consequently improving ranking effectiveness. Previous techniques typically generate words and phrases related to the original query, but do not consider how these words and phrases would fit together in new queries. In this paper, we focus on an implementation of an approach that models reformulation as a distribution of queries, where each query is a variation of the original query. This approach considers a query as a basic unit and can capture important dependencies between words and phrases in the query. The implementation discussed here is based on passage analysis of the target corpus. Experiments on the TREC collection show that the proposed model for query reformulation significantly outperforms state-of-the-art methods.

Categories and Subject Descriptors

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

General Terms

Algorithms, Experimentation, Performance

Keywords

Query Reformulation, Query Substitution, Query Segmentation, Passage Analysis, Information Retrieval

1. INTRODUCTION

Query reformulation is considered as a process of modifying or rewriting the original query to better match the vocabulary of relevant documents. Many previous models of query reformulation [7, 9, 10] have focused on generating related words and phrases to expand the original query. However, they do not consider how the new words and phrases can be used together to form queries that are variations of the original query, thus the important dependencies between those new terms can be missed. Other research on web query reformulation, on the other hand, has tended to focus on generating a single new query (e.g. [2][6]) by applying a specific reformulation operation. Different operations have been studied such as Query Segmentation [2] and Query Substitution [6]. However, little work considers combining these operations from a unified perspective, thus the important information about alternative query reformulations is not captured.

An alternative approach is to transform the original query into a distribution of reformulated queries. Since the reformulated query that involves a particular choice of words and phrases is explicitly modeled, this approach captures dependencies between those query terms. On the other hand, this approach naturally combines different reformulation operations, where all these operations are considered as methods for generating reformulated queries. The detailed analysis of this approach and the comparisons with previous models are provided in Xue and Croft [17].

In this paper, we focus on an implementation of this approach based on passage analysis of the target corpus, avoiding the use of proprietary resources. This implementation allows us to use TREC collections that make fair comparisons with other methods possible.

We make the observation that in the target corpus, passages containing all query words or most of the query words provide a good source of information for reformulating queries. Specifically, passages with all query words provide information about the common ways that people split the original query into different concepts. For example, given the original query "oil industry history", the analysis of passages containing all these three words in the Gov2 collection[1] shows that the original query is split as "(oil industry)(history)" in 51 passages. Similarly, passages only containing some query words can indicate possible ways of substituting missing words. For example, the analysis of passages containing "industry history" on Gov2 shows that "petroleum industry history" appears in 46 passages, which indicates that "petroleum industry history" is a potential substitution for the original query.

2. IMPLEMENTATION BASED ON PASSAGE ANALYSIS

In this section, a passage analysis based implementation is described. This implementation can be divided into three

[1]Gov2 is a TREC collection used for ad-hoc retrieval.

CIKM'10, October 26–30, 2010, Toronto, Ontario, Canada.

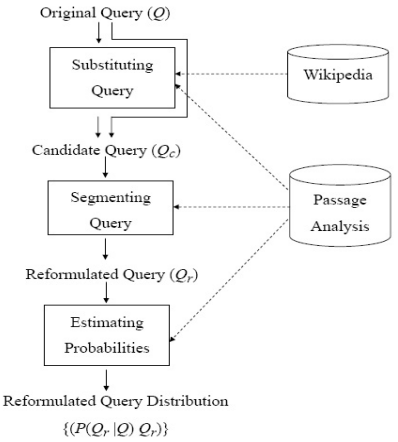

Figure 1: The passage analysis based implementation.

main steps: first, the original query (Q) is substituted through passage analysis and with the help of Wikipedia; second, the candidate queries (Q_c) consisting of the original query and the substituted queries are segmented based on passage analysis; third, the probabilities of the reformulated queries (Q_r) generated from the previous steps are estimated using passage level information. The above framework is illustrated in Fig. 1.

2.1 Generating Query Substitutions

Three different methods are considered to find query substitutions. The first two methods are based on passage analysis and the last uses information from Wikipedia.

2.1.1 Morphologically Similar Words

Morphologically similar words are a reliable way to substitute the original query words using appropriate morphological variants. In this paper, the morphologically similar words are chosen by considering other query words. Specifically, given the original query $Q = (q_1...q_i...q_l)$, for each query word q_i, we extracted all passages containing all query words except q_i. For each extracted passage, if we find a word q_i' which is morphologically similar to q_i, q_i' will be considered as a substitution of q_i and a candidate query is generated as $Q_c = (q_1...q_i'...q_l)$. Note a morphologically similar word is considered as a substitution only when it appears in passages containing all other query words, which further guarantees the quality of the substitution word.

2.1.2 Pattern-based Method

Two types of patterns are derived from the original query and are then used to match qualified passages to find query substitution.

Adding-word patterns are used to find substitutions which replace a bigram of the original query with an n-gram that adds some words in the middle of the query bigram. Specifically, given the original query $Q = (q_1...q_iq_{i+1}...q_l)$, we consider substituting any q_iq_{i+1} with $q_iw_1...w_sq_{i+1}$. $w_1...w_s$ are the added words and s is the number of added words. Here, we only consider adding one or two words. First, for each bigram q_iq_{i+1}, a pattern is designed as $q_i \star q_{i+1}$, where $\star$ denotes any word or any two words. Then, passages containing all query words $q_1...q_l$ are extracted. For each extracted passage, if the designed pattern $q_i \star q_{i+1}$ matches this passage, q_iwq_{i+1} is collected as a substitution of q_iq_{i+1}. Here, w denotes the added word/words. Thus, a candidate query is generated as $q_1...q_iwq_{i+1}...q_l$.

Changing-word patterns are used to find substitutions that replace a trigram of the original query with a new trigram where the middle word is different. Specifically, given the original query $Q = (q_1...q_iq_{i+1}q_{i+2}...q_l)$, we consider substituting $q_iq_{i+1}q_{i+2}$ with q_iwq_{i+2}, where w is a different word. First, for each trigram $q_iq_{i+1}q_{i+2}$ of the original query, a pattern is designed as $q_i \star q_{i+2}$. Second, passages containing all query words except q_{i+1} are extracted. For each extracted passage, if the pattern $q_i \star q_{i+2}$ matches this passage, q_iwq_{i+2} is collected as a substitution for $q_iq_{i+1}q_{i+2}$ and a candidate query substitution is generated as $q_1...q_iwq_{i+2}...q_l$.

2.1.3 Wikipedia Redirect Page

Some query substitutions are difficult to obtain only relying on corpus information, thus the third method uses Wikipedia redirect page as an external resource. A redirect page in Wikipedia is designed to send the user to the article with an alternative title[2].

All redirect pages are organized as a set of tuples $\{(p_{src}, p_{tar})\}$, where the phrase p_{src} is redirected to the phrase p_{tar}. Given the original query $Q = (q_1...q_{i+1}...q_{i+n}...q_l)$, all n-grams ($n > 1$) are extracted. For each n-gram $q_{i+1}...q_{i+n}$, if it matches p_{src} or p_{tar} in any tuple, the corresponding p_{tar} or p_{src} will be collected as a substitution. Then, a candidate query substitution is generated as $(q_1...p_{tar}...q_n)$ or $(q_1...p_{src}...q_n)$.

2.2 Generating Query Segmentations

In this step, phrase structures are detected using passage analysis for all candidate queries including both the original query and query substitutions. The basic idea can be described as follows. Given a candidate query, passages containing all query words are extracted. Then, each extracted passage tells us one way to segment the candidate query. After analyzing all extracted passages, the most frequent ways of segmenting the candidate query can be determined.

2.3 Estimating Probabilities for Queries

The probability $P(Q_r|Q)$ for Q_r is calculated as follows:

$$P(Q_r|Q) = \sum_D P(Q_r|D)P(D|Q) \quad (1)$$

$P(D|Q)$ measures the similarity between D and Q, which is calculated using the language model approach with Dirichlet Smoothing[13, 18]. $P(Q_r|D)$ measures the probability that Q_r is observed in D. $P(Q_r|D)$ is estimated by dividing the number of passages where Q_r appears in by the total number of passages of document D, which is shown in Eq. 2. $\#_{psg}(Q_r, D)$ is the number of passages of D where Q_r has appeared and $\#_{psg}(D)$ is the number of passage of D.

$$P(Q_r|D) = \frac{\#_{psg}(Q_r, D)}{\#_{psg}(D)} \quad (2)$$

After this step, we have generated a distribution of reformulated queries as $\{(P(Q_r|Q), Q_r)\}$. Table 1 provides examples of the reformulated queries generated by different methods and their associated probabilities for the original query "oil industry history".

[2]The definition of redirect pages can be found at `http://en.wikipedia.org/wiki/Redirects_on_wikipedia`

Table 2: More examples of reformulated queries

source	controlling type ii diabetes	pyramid scheme	low white blood cell count
Orig	(controlling)(type ii diabetes)	(pyramid scheme)	(low white blood cell count)
Wiki	(controlling)(type 2 diabetes)	(1)(up)(system)	(low)(leukocyte count)
Morph	(control)(type ii diabetes)	n/a	(lower)(white blood cell count)
Pat-add	n/a	(pyramid promotional scheme)	n/a
Pat-chg	(controlling)(type 2 diabetes)	n/a	(low red blood cell count)
source	hybrid alternative fuel cars	ban human cloning	Enron California energy crisis
Orig	(hybrid)(alternative fuel)(cars)	(ban)(human cloning)	(Enron)(California energy crisis)
Wiki	(hybrid)(alternative fuel)(vehicle)	(ban)(human)(clone)	(Enron)(California electricity crisis)
Morph	(hybrid)(alternative fueled)(cars)	(bans human cloning)	(Enrons)(California energy crisis)
Pat-add	n/a	(ban on human cloning)	n/a
Pat-chg	n/a	(ban reproductive cloning)	(Enron)(California electricity crisis)

Table 1: Examples of the reformulated queries of "oil industry history"

source	Q_r	$P(Q_r\|Q)$
original query	"(oil industry)(history)"	0.28
wiki redirect	"(petroleum industry)(history)"	0.24
morph similar	"(oil)(industrialized)(history)"	0.18
adding pattern	"(oil and gas industry)(history)"	0.20
changing pattern	"(oil)(spill)(history)"	0.10

3. RETRIEVAL MODEL WITH REFORMULATED QUERIES

In this section, we incorporate $\{(P(Q_r|Q), Q_r)\}$ into the retrieval function. The basic idea is to combine the original query Q and the distribution of reformulated queries $\{(P(Q_r|Q), Q_r)\}$ together. We assign the probability α to Q and $1-\alpha$ to $\{(P(Q_r|Q), Q_r)\}$. Here, α is a parameter. The retrieval score of each document is calculated by Eq. 3.

$$\begin{aligned} \text{score}(D) &= \alpha \log(P(Q|D)) + \\ &\quad (1-\alpha) \sum_{i=1}^{k} P(Q_{r_i}|Q) \log(P(Q_{r_i}|D)) \quad (3) \end{aligned}$$

$P(Q|D)$ and $P(Q_{r_i}|D)$ are estimated using the language model approach with Dirchlet Smoothing. k is the number of reformulated queries with the highest probabilities. In the case where the total number of reformulated queries is smaller than k, the actual number will be used.

4. EXPERIMENTS

The Gov2 collection is used for experiments. Two indexes are built, one not stemmed and the other stemmed with the Porter Stemmer. For each topic, the title part is used as the query. The reformulated queries are generated from the non-stemmed index and then the retrieval model with reformulated queries is run on both indexes. The size of passage is fixed to 20. α is set to 0.8 and k is set to 20 for the retrieval model using reformulated queries.

For different substitution methods, "Orig" denotes no substitution of the original query. "Wiki" denotes using the Wikipedia redirect page. "Morph" denotes the method of finding the morphologically similar words. "Pat-add" denotes the method of using adding-word patterns and "Pat-chg" denotes the method of using changing-word patterns.

Several baselines are considered. QL denotes the query likelihood language model [13, 18]. SDM denotes the sequential dependence model [9]. RM denotes the relevance model [7]. SVM-seg denotes a SVM-based query segmentation method [1]. QL-psg denotes a passage-augmented language model [8].

Table 3: The effect of combining Orig(o) with other sources including Wiki(w), Morph(m), Pat-add(a), Pat-chg(c). ^{q} denotes significantly different with QL, $_d$ denotes significantly different with SDM and ^{r} denotes significantly different with RM

	nonstem			pstem		
	MAP	P@5	P@10	MAP	P@5	P@10
QL	22.00	60.00	58.85	29.99	57.12	54.90
SDM	23.49	60.96	57.79	33.40	61.35	59.81
RM	23.60	64.62	**61.06**	31.94	57.50	56.92
SVM-seg	21.83	58.46	56.54	30.07	56.35	54.42
QL-psg	22.12	61.35	58.65	31.11	57.69	55.48
o	23.32^q	62.31	59.42	33.52^q	62.31^{qr}	61.44^{qr}
o,w	23.99^q	64.04^q	60.00	34.00^{qr}	63.27^{qr}	61.44^{qr}
o,m	23.91^q	64.04^q	60.10	33.76^{qr}	60.77^q	61.06^{qr}
o,a	23.85^q	62.69	$\mathbf{61.06}_d$	34.66^{qr}	63.27^{qr}	62.69^{qr}
o,w,m,a	24.65^q_d	$\mathbf{64.81}^q_d$	$\mathbf{61.06}_d$	35.12^{qr}_d	$\mathbf{63.65}^{qr}$	62.40^{qr}
o,w,m,a,c	$\mathbf{24.67}^q_d$	64.62^q_d	$\mathbf{61.06}_d$	$\mathbf{35.19}^{qr}_d$	$\mathbf{63.65}^{qr}$	$\mathbf{62.79}^{qr}$

Mean average precision (MAP), precision at 5 (P@5) and precision at 10 (P@10) are used as performance measures. The two-tailed t-test is used to measure significance.

4.1 Examples

Table 2 shows more examples of the reformulated queries generated by our method. Sometimes, the reformulated queries generated from different sources happen to be the same. Also, some sources can not generate reformulated queries for certain queries (denoted as "n/a" in Table 2).

4.2 Results

We conducted experiments to explore the effect of combining different sources of reformulated queries. Since Orig is the most reliable source of reformulated queries, in order to make a fair comparison, the experiment is conducted over queries that Orig can reformulate. Other sources are combined with Orig. Different baselines are compared including QL, SDM, RM, SVM-seg and QL-psg. The results are reported in Table 3. The best results are bolded.

Table 3 shows that after combining Orig with other sources of reformulated queries, our proposed method beats all baseline methods. In particularly, it performs significantly better than SDM and RM, the state-of-the-art reformulation models, which supports the advantages of the proposed framework that models reformulation as query distribution. When Wiki, Morph and Pat-add are combined with Orig respectively, some improvement can be observed over Orig and

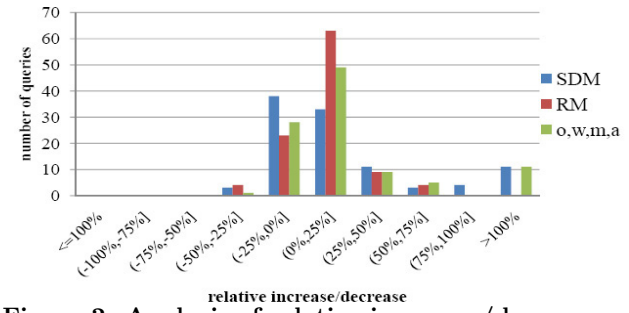

Figure 2: Analysis of relative increases/decreases of MAP over QL on Gov2 with pstem. The bars from left to right indicate SDM, RM and o,w,m,a.

when all three sources are used together with Orig, significant improvement is observed. Pat-chg can bring some slight improvement when it is combined with other sources.

In order to better understand the behaviors of different reformulation models, we analyzed the number of queries each model increases or decreases over QL. Fig 2 demonstrates the histograms of SDM, RM and o,w,m,a based on the relative increases/decreases of MAP over QL on the Porter-stemmed index. The results on the non-stemmed index are similar.

In Fig 2, the improvement of o,w,m,a over SDM mainly comes from decreasing the number of queries that hurt the performance of QL (SDM has 38 queries in the range (-25%, 0%], while o,w,m,a has 28 queries in it). The improvement of o,w,m,a over RM mainly comes from increasing the number of queries that significantly improve the performance of QL (RM has 0 query in the range >100%, while o,w,m,a has 11 queries in it).

5. RELATED WORK

Besides the work mentioned in Section 1, [3][16] expanded the original query with new words and [11] augmented the query representation with phrases.

Tan and Peng [14] proposed a unsupervised query segmentation method using a concept-based language model. Bendersky et al [1] proposed a two-stage query segmentation method.

Wang and Zhai [15] mined the query log to find potential query term substitution and addition patterns. Dang and Croft [4] re-implemented and tested Wang and Zhai's method for TREC collections. The results showed this method does not work well for the well-formed TREC queries.

Peng et al [12] proposed a context-sensitive stemming method for web queries, where query words are stemmed based on the analysis of their context.

Guo et al [5] proposed a CRF-based model for query refinement, which combined several tasks like spelling correction, stemming and phrase detection. Within their model, different tasks can be solved simultaneously instead of sequentially.

6. CONCLUSION

In order to capture the dependencies of query words and phrases, we discuss a novel reformulation framework, where the original query is reformulated as a distribution of queries instead of a bag of components. A passage analysis based implementation is described in this paper and experiments on the TREC collection show that this model significantly improves retrieval performance. Currently, we have focused on short queries. Extending the passage analysis techniques to long queries will be an interesting future issue.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by NSF grant #IIS-0711348. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the sponsor.

7. REFERENCES

[1] M. Bendersky, D. A. Smith, and W. B. Croft. Two-stage query segmentation for information retrieval. In *SIGIR09*, pages 810–811, Boston, MA, 2009.

[2] S. Bergsma and Q. I. Wang. Learning noun phrase query segmentation. In *EMNLP-CoNLL07*, pages 819–826, Prague, 2007.

[3] G. Cao, J. Y. Nie, J. Gao, and S. Robertson. Selecting good expansion terms for pseudo-relevance feedback. In *SIGIR08*, pages 243–250, Singapore, 2008.

[4] V. Dang and W. B. Croft. Query reformulation using anchor text. In *WSDM10*, New York, NY, 2010.

[5] J. Guo, G. Xu, H. Li, and X. Cheng. A unified and discriminative model for query refinement. In *SIGIR08*, pages 379–386, Singapore, 2008.

[6] R. Jones, B. Rey, O. Madani, and W. Greiner. Generating query substitutions. In *WWW06*, pages 387–396, Ediburgh, Scotland, 2006.

[7] V. Lavrenko and W. B. Croft. Relevance based language models. In *SIGIR01*, pages 120–127, New Orleans, LA, 2001.

[8] X. Liu and W. B. Croft. Passage retrieval based on language models. In *CIKM02*, pages 375–382, McLean, VA, 2002.

[9] D. Metzler and W. B. Croft. A markov random field model for term dependencies. In *SIGIR05*, pages 472–479, Salvador,Brazil, 2005.

[10] D. Metzler and W. B. Croft. Latent concept expansion using markov random fields. In *SIGIR07*, pages 311–318, Amsterdam, the Netherlands, 2007.

[11] G. Mishne and M. de Rijke. Boosting web retrieval through query operations. In *ECIR05*, pages 502–516, Spain, 2005.

[12] F. Peng, N. Ahmed, X. Li, and Y. Lu. Context sensitive stemming for web search. In *SIGIR07*, pages 639–646, Amsterdam, the Netherlands, 2007.

[13] J. M. Ponte and W. B. Croft. A language modeling approach to information retrieval. In *SIGIR98*, pages 275–281, Melbourne, Australia, 1998.

[14] B. Tan and F. Peng. Unsupervised query segmentation using generative language models and Wikipedia. In *WWW08*, pages 347–356, Beijing,China, 2008.

[15] X. Wang and C. Zhai. Mining term association patterns from search logs for effective query reformulation. In *CIKM08*, pages 479–488, Napa Valley, CA, 2008.

[16] J. Xu and W. B. Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems*, 18(1):79–112, 2000.

[17] X. Xue and W. B. Croft. Representing queries as distributions. In *SIGIR10 Workshop on Query Representation and Understanding*, pages 9–12, Geneva, Switzerland, 2010.

[18] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *SIGIR01*, pages 334–342, New Orleans, LA, 2001.