Query Aspect Based Term Weighting Regularization in Information Retrieval

Wei Zheng and Hui Fang

Department of Electrical and Computer Engineering
University of Delaware, USA
{zwei,hfang}@ece.udel.edu

Abstract. Traditional retrieval models assume that query terms are independent and rank documents primarily based on various term weighting strategies including TF-IDF and document length normalization. However, query terms are related, and groups of semantically related query terms may form query aspects. Intuitively, the relations among query terms could be utilized to identify hidden query aspects and promote the ranking of documents covering more query aspects. Despite its importance, the use of semantic relations among query terms for term weighting regularization has been under-explored in information retrieval. In this paper, we study the incorporation of query term relations into existing retrieval models and focus on addressing the challenge, i.e., how to regularize the weights of terms in different query aspects to improve retrieval performance. Specifically, we first develop a general strategy that can systematically integrate a term weighting regularization function into existing retrieval functions, and then propose two specific regularization functions based on the guidance provided by constraint analysis. Experiments on eight standard TREC data sets show that the proposed methods are effective to improve retrieval accuracy.

1 Introduction

It has been a long standing challenge to develop robust and effective retrieval models. Many retrieval models have been proposed and studied including vector space models [19], classic probabilistic models [18,23,7], language models [15,25] and recently proposed axiomatic models [5]. These retrieval models rank documents based on the use of various term weighting strategies including term frequency, inverse document frequency and document length normalization [4].

Although these retrieval models rank documents differently, they may fail to return relevant documents for the same reasons. In the previous studies [8,2], researchers conducted failure analysis for the state-of-the-art retrieval models and showed that one of the common failures is that the retrieval models fail to return documents covering all the query aspects. This failure, in a way, is caused by the underlying assumption that query terms are independent to each other. Traditional retrieval models often ignore query term relations in term

C. Gurrin et al. (Eds.): ECIR 2010, LNCS 5993, pp. 344–356, 2010.

weighting and treat every query term as a query aspect. However, such an assumption is not always true. Query terms could be related to each other, and multiple semantically related query terms may form a query aspect. Intuitively, query term relations are useful to identify different aspects in the query and can provide guidance on the term weighting. For example, consider query "stolen or forged art" (i.e., topic 422 in TREC8). The query contains two aspects, i.e., "stolen or forged" and "art". Intuitively, documents covering both aspects should be ranked higher than those covering only one aspect. Thus, a document talking about "stolen or forged money" should not be ranked higher than the one talking about "stolen art". Unfortunately, a query aspect may contain one or multiple terms. Since existing retrieval models treat query terms independently, they may assign lower relevance scores to the documents covering more query aspects. In particular, Buckley [2] reported that all the analyzed retrieval models over-emphasized one aspect of the query, i.e., "stolen or forged", while missing the other aspect, i.e., "art". Clearly, it is important to exploit query aspect information to regularize term weighting and incorporate the term regularization into existing retrieval functions. Despite its importance, the use of query term relations for term weighting regularization has been under-explored in the IR literature. It remains unclear how to regularize term weighting based on query term relations and how to systematically incorporate the term weighting regularization functions into existing retrieval functions.

In this paper, we study the problem of incorporating query term relations into existing retrieval functions. Specifically, we discuss how to utilize term semantic similarities to identify query aspects and how to systematically exploit the query aspect information to regularize term weighting in existing retrieval functions. We first present a general strategy based on the recently proposed inductive definition scheme [5]. We show that the inductive definition provides a natural way of extending an existing retrieval function with the aspect based term weighting regularization - all we need to do is to generalize the query growth function of a retrieval function to incorporate an aspect-based term regularization function. We then propose two term weighting regularization functions that can utilize the query term relations such as query aspects in order to avoid favoring documents that cover fewer query aspects. To evaluate the effectiveness of the proposed methods, we integrate them into four representative retrieval functions (i.e., pivoted normalization retrieval function [21], Okapi BM25 retrieval function [18], Dirichlet prior retrieval function [25] and axiomatic retrieval function [5]), and conduct experiments over eight representative TREC data sets. Experiment results show that, for verbose queries, the proposed methods can significantly and consistently improve the retrieval accuracy on almost all the data sets we experimented with. The rest of the paper is organized as follows. We discuss related work in Section 2 and briefly review the basic ideas of inductive definition and axiomatic approaches in Section 3. We then present our work on aspect-based term weighting regularization in Section 4, and discuss experiment results in Section 5. Finally, we conclude in Section 6.

2 Related Work

Most traditional retrieval models assume that query terms are independent. To improve retrieval accuracy, many studies have recently tried to exploit the relations among query terms. They range from the early studies on the use of phrases in document retrieval [3,14,12] to the recent work on query segmentation [9,16,10], term proximity [22], and term dependencies [13]. Previous studies on the use of phrases in retrieval models [3,14,12] often identified phrases using either statistical or syntactic methods, scored documents with matched phrases, and then heuristically combined the term-based and phrase-based relevance scores. Recent studies [1,11] focused on using supervised learning techniques to support verbose queries. In particular, Bendersky and Croft [1] proposed a probabilistic model for combining the weighted key concepts with the original queries. Query segmentation refers to the problem of segmenting a query into several query concepts. The commonly used methods are based on term co-occurrences [9,16]. Similar to previous work [9,16,6], we assume that term co-occurrences such as mutual information can be used to compute term semantic similarity. But our work focuses on aspect-based term weighting regularization instead of query aspect identification. Kumaran and Allan [10] proposed to interact with users and allow them to extract the best sub-queries from a long query. However, they did not study how to utilize the sub-queries or segmented queries to regularize term weighting. Our work is also related to the studies of term dependencies and term proximity [13,22]. For example, Metzler and Bruce [13] proposed a term dependence model, which can model different dependencies between query terms. However, the proposed model affects the retrieval efficiency, and it remains unclear how to incorporate term dependencies into other retrieval models. Tao and Zhai [22] studied how to exploit term proximity measures, but our work focuses on the semantic relations among query terms.

Although the motivation is similar, our work differs from the previous work in that (1) we attempt to systematically integrate aspect based term weighting regularization into a variety of existing retrieval models; (2) we propose to use constraint analysis to provide guidance on the implementation of term weighting regularization functions; (3) our methods do not rely on the use of external resources and are less computational expensive than the method proposed in the previous study [1]. Moreover, as shown in Section 5, the performance of our methods are comparable to the performance reported in the previous study [1].

3 Axiomatic Approaches to IR

Axiomatic approaches have recently been proposed as a new way of analyzing and developing retrieval functions [4,5]. The basic idea is to search in a space of candidate retrieval functions for the ones that can satisfy a set of desirable retrieval constraints. Retrieval constraints are often defined by formalizing various retrieval heuristics that any reasonable retrieval functions should satisfy. Previous studies proposed several retrieval constraints for TF-IDF weighting, document length normalization, semantic term matching and term proximity [4,5,6,22]. These constraints are shown to be effective to provide guidance on how to improve the performance of an existing retrieval function and how to develop new retrieval functions.

To constrain search space of retrieval functions, an inductive definition of retrieval functions was proposed [5]. The inductive definition decomposes a retrieval function into three component functions: (1)primitive weighting function, which gives the relevance score of a one-term document for a given one-term query; (2)document growth function, which captures the change of relevance scores when a term is added to a document; and (3)query growth function, which captures the score change when a term is added to a query. There are multiple ways of instantiating each of these component functions. In general, different instantiations of the three component functions would lead to different retrieval functions.

Previous study [5] showed that most existing "bag of words" representation based retrieval functions can be decomposed with the proposed inductive definition, and they have similar instantiations of query growth function as follows.

$$S(Q \cup \{q\}, D) = S(Q, D) + S(\{q\}, D) \times \Delta(c(q, Q)) \tag{1}$$

where D denotes a document, Q denotes a query, and $Q \cup \{q\}$ denotes a new query generated by adding a term q to query Q, $S(Q, D)$ denotes the relevance score and $c(q, Q)$ is the term occurrence of q in query Q. Four existing retrieval functions differ in the implementation of $\Delta(c(q, Q))$. In particular, Okapi implements it as $\Delta(x) = \frac{(k_3+1)\times(x+1)}{k_3+x+1} - \frac{(k_3+1)\times x}{k_3+x}$, where k_3 is the parameter in the Okapi BM25 retrieval function, and other functions including Pivoted, Dirichlet and axiomatic retrieval functions implement it as $\Delta(x) = 1$. Note that these query growth functions are only related to query term frequency and do not consider the semantic relations among query terms.

4 Aspect-Based Query Term Regularization

4.1 Problem Formulation

It is known that terms are semantically related. For example, the occurrence of a term in a document may indicate the occurrences of its related terms in the document. Within a query, groups of semantically related terms may form different query aspects. In general, we define a *query aspect* as a group of query terms that are semantically similar to each other. A query may contain one or more query aspects, and a query aspect may contain one or more query terms. For example, query "ocean remote sensing" has two aspects, i.e., "ocean" and "remote sensing". Formally, let $Q = \{q_1, q_2, ..., q_n\}$ be a query with n terms. $\mathcal{A}(q) \subseteq Q$ denotes the aspect of query term q. If the aspect of term q_1 has two terms, i.e., q_1 and q_2, then $\mathcal{A}(q_1) = \{q_1, q_2\}$. $s(t_1, t_2) \in [0, +\infty]$ denotes the semantic similarity between two terms t_1 and t_2. If $\mathcal{A}(q_1) = \mathcal{A}(q_2)$ and $\mathcal{A}(q_1) \neq \mathcal{A}(q_3)$, then $s(q_1, q_2) > s(q_1, q_3)$ and $s(q_1, q_2) > s(q_2, q_3)$. The underlying assumption is that terms within a query aspect should be more semantically similar than those from different query aspects.

Indeed, the definition of query aspects suggests that one possible way of identifying query aspects is to cluster query terms based on a term semantic similarity function. We explore a single-link hierarchical clustering algorithm in the paper. Specifically, we start with each term in a query as a cluster, and then keep combining two clusters when there exist two terms, one from each cluster, whose similarity is higher than a threshold. The threshold can be set as the average similarity of all term pairs for the query. The algorithm stops when no clusters can be further combined. As a result, every cluster can be regarded as a query aspect.

$s(t_1, t_2)$ may be any given term semantic similarity function. Following the previous studies [20,24,6], we assume that co-occurrences of terms reflect underlying semantic relations among query terms, and adopt the expected mutual information measure (EMIM) [23,24] as the term semantic similarity function. Formally, the term semantic similarity function is defined as follows.

$$s(t_1, t_2) = I(X_{t_1}, X_{t_2}) = \sum_{X_{t_1}, X_{t_2} \in \{0,1\}} p(X_{t_1}, X_{t_2}) \log \frac{p(X_{t_1}, X_{t_2})}{p(X_{t_1})p(X_{t_2})}. \tag{2}$$

X_t is a binary random variable corresponding to the presence/absence of term t in each document. We compute the mutual information for query term pairs using the test collection itself and leave other possible term semantic similarity functions and other aspect identification methods as our future work.

Note that most traditional retrieval models [19,23,7,15,25,5] assume that query terms are independent, and each query term corresponds to a query aspect, i.e., $\forall q \in Q, \mathcal{A}(q) = \{q\}$. As shown in the previous studies [2,8], the assumption often leads to non-optimal retrieval performance because the retrieval models may incorrectly assign higher relevance scores to the documents that cover fewer query aspects. For example, for the query "ocean remote sensing" mentioned earlier, all the analyzed retrieval models over-emphasized one aspect "remote sensing" and failed to return documents covering both aspects.

In this paper, we aim to study how to utilize the semantic relations among query terms, such as query aspect information, to regularize term weighting in order to improve the retrieval performance of an existing retrieval function.

4.2 General Strategy

The occurrence of a query term often indicates the occurrences of its semantically related terms. If a query term has many semantically related terms in a query, this term and its query aspect might be over-emphasized because of the matching of these related terms. To solve this problem, we now propose a general strategy that can regularize term weighting based on semantic relations among query terms for existing retrieval functions. Specifically, we first define a constraint for term weighting regularization based on query term relations and integrate the regularization function into existing retrieval functions through the inductive definition scheme under the guidance of constraint analysis [4,5].

The basic idea is to adjust the weights of a query term based on its semantic relations with other query terms so that the documents covering more query aspects would be ranked higher than those covering fewer aspects. We can formalize this idea as a retrieval constraint. Let us first introduce some notations. Q denotes a query and $\mathcal{A}(q) \subseteq Q$ denotes the query aspect of query term q. Query terms q_1 and q_2 belong to different query aspects if $\mathcal{A}(q_1) \neq \mathcal{A}(q_2)$. Let $td(t)$ denote any reasonable measure of term discrimination value of term t (usually based on term popularity in a collection), such as IDF. The term weighting regularization constraint can be defined formally as follows.

Regularization Constraint: Let $Q = \{q_1, q_2, q_3\}$ be a query with three query terms q_1,q_2 and q_3, where $td(q_2) = td(q_3)$. We assume that $\mathcal{A}(q_1) = \mathcal{A}(q_2)$ and $\mathcal{A}(q_1) \neq \mathcal{A}(q_3)$, or equivalently $s(q_1, q_2) > s(q_1, q_3)$ and $s(q_1, q_2) > s(q_2, q_3)$ based on the definition of query aspects. Let D_1 and D_2 be two documents, and $c(t, D)$ denotes the count of term t in document D. If $c(q_1, D_1) = c(q_1, D_2) > 0$, $c(q_2, D_1) = c(q_3, D_2) > 0$, $c(q_3, D_1) = c(q_2, D_2) = 0$, and $|D_1| = |D_2|$, then $S(Q, D_1) < S(Q, D_2)$.

The constraint requires a retrieval function to assign a higher relevance score to the document that covers more query aspects. Thus, even though both D_1 and D_2 match two query terms with the same term discrimination values, we would like D_2 to have a higher score because the matched query terms in D_1 (i.e., q_1 and q_2) are from the same aspect while the matched query terms in D_2 (i.e., q_1 and q_3) are from different aspects.

We analyze four representative retrieval functions with the constraint. The functions are *pivoted normalization function* derived from vector space models [19,21]), *Okapi BM25* derived from classical probabilistic models [23,7,18], *Dirichlet prior* derived from language models [15,25] and *F2-EXP* derived from axiomatic retrieval models [5]. The constraint analysis results show that none of the functions satisfies the constraint because they ignore the query term relations and would assign the same scores to both documents.

To make the retrieval functions satisfy the constraint, we need to incorporate semantic relations among query terms into retrieval functions. As reviewed in Section 3, the inductive definition makes it possible to decompose a retrieval function into three component functions. Clearly, a natural way of incorporating semantic relations among query terms into retrieval functions is to generalize the query growth function so that it is related to not only the query term frequency but also the semantic relations between a query term and other terms in the query. Thus, we propose to define the following *generalized query growth function* by extending Equation (1) with a function $f(q, Q, s(\cdot))$ that regularizes the query term weighting based on the semantic relations between term q and query Q.

$$S(Q \cup \{q\}, D) = S(Q, D) + S(\{q\}, D) \times \Delta(c(q, Q)) \times f(q, Q, s(\cdot)) \tag{3}$$

where $s(\cdot)$ is a term semantic similarity function such as Equation (2).

To integrate term regularization function $f(q, Q, s(\cdot))$ into a retrieval function, we can first decompose the retrieval function into three component functions, and then combine its original primitive weighting function and original document growth function with the generalized query growth function as shown in Equation (3). Thus, the new retrieval function is an extension of the original retrieval function, and it uses the regularization function $f(\cdot)$ to regularize term weighting based on the semantic relations among query terms. The extended versions for the analyzed four retrieval functions are shown as follows.

- Extended Pivoted Normalization:

$$S(Q,D) = \sum_{t \in D \cap Q} \frac{1+\log(1+\log(c(t,D)))}{1-b+b\frac{|D|}{avdl}} \times c(t,Q) \times \log\frac{N+1}{df(t)} \times f(t,Q,s(\cdot)) \quad (4)$$

- Extended Okapi BM25:

$$S(Q,D) = \sum_{t \in Q \cap D} \log\frac{N-df(t)+0.5}{df(t)+0.5} \cdot \frac{(k_3+1)\cdot c(t,Q)}{k_3+c(t,Q)} \cdot \frac{(k_1+1)\cdot c(t,D)}{k_1((1-b)+b\frac{|D|}{avdl})+c(t,D)} \cdot f(t,Q,s(\cdot)) \quad (5)$$

- Extended Dirichlet Prior:

$$S(Q,D) = \sum_{t \in Q \cap D} c(t,Q)\cdot\log(1+\frac{c(t,D)}{\mu \times p(t|C)})\cdot f(t,Q,s(\cdot)) + \log\frac{\mu}{|D|+\mu}\cdot\sum_{q \in Q} f(q,Q,s(\cdot)) \quad (6)$$

- Extended Axiomatic:

$$S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times (\frac{N}{df(t)})^{0.35} \times f(t,Q,s(\cdot)) \times \frac{c(t,D)}{c(t,D)+b+\frac{b\times|D|}{avdl}} \quad (7)$$

$c(t,D)$ is the count of term t in document D, $c(t,Q)$ is the count of term t in query Q, $df(t)$ is the number of documents with term t, $|D|$ is the length of document d, $avdl$ is the average document length of the document collection, and $p(t|C)$ is the probability of term t in collection C. k_1 is set to 1.2 and k_3 is set to 1000. μ and b are parameters in the original retrieval functions. Note that the original retrieval functions are the special cases of their corresponding extensions when $f(q,Q,s(\cdot)) = 1, \forall q \in Q$.

The proposed general strategy provides a systematical way of incorporating query term relations into any retrieval functions that can be decomposed with the inductive definition scheme. Moreover, although we discussed how to compute term semantic similarity and how to identify aspect, the proposed strategy is generally applicable for any other reasonable term semantic similarity functions and aspect identification methods.

The remaining challenge of the proposed general strategy is to select appropriate implementation for function $f(q,Q,s(\cdot))$, which regularizes the term weighting based on the semantic relations among query terms so that the extended retrieval functions satisfy the defined constraint. We discuss two term weighting regularization functions in the next subsection.

4.3 Term Weighting Regularization Functions

To make extended retrieval functions satisfy the constraint, we need to implement the regularization function f in a way so that it would demote the weights of terms that either belong to large query aspects or have many semantically related terms. Thus, we propose the following two term weighting regularization functions, with the first one explicitly capturing the query aspect information and the second one implicitly capturing aspect information through the semantic relations among query terms.

Aspect size based regularization: The proposed regularization constraint is to avoid over-favoring documents covering fewer aspects. One possible solution is to penalize terms from larger aspects. The rationale is that a larger query aspect needs to be penalized more harshly because the aspect is more likely to be over-favored due to the larger number of query terms in the aspect. Thus, we propose an aspect size based regularization function, f_{size}, which explicitly uses the query aspect information and regularizes the term weighting based on the size of its aspect. Formally,

$$f_{size}(q, Q, s(\cdot)) = 1 - \alpha + \alpha \cdot \left(\frac{|\mathcal{A}_s(q)|}{|Q|} \right)^{-\beta} \tag{8}$$

where $|\mathcal{A}_s(q)|$ is the number of terms in query aspect $\mathcal{A}_s(q)$ identified based on term similarity function $s(\cdot)$ and $|Q|$ is the number of terms in query Q. Clearly, the value of f_{size} for a query term q is inversely correlated with the size of its query aspect. It means that if a query term is in a larger query aspect, the weights of the query term should be penalized more because the matching of its semantic related terms may also contribute to the relevance score. There are two parameters in the regularization function: β controls the curve shape of the regularized function, and α balances the original and the regularized term weighting. We will examine the parameter sensitivity in the experiment section.

Semantic similarity based regularization: The aspect size based regularization function requires us to explicitly identify query aspects. However, the accuracy of aspect identification may greatly affect the retrieval performance for the regularization based retrieval function. To overcome the limitation, we propose a semantic similarity based regularization function, i.e., f_{sim}, which does not require the explicit aspect identification. Specifically, f_{sim} exploits the semantic similarity between term q and its query Q, which is computed by taking the average of the semantic similarity between q and other query terms in Q. Formally, we have

$$f_{sim}(q, Q, s(\cdot)) = 1 - \alpha + \alpha \times (-\log(\frac{\sum_{q' \in Q - \{q\}} s(q, q')}{|Q| - 1})) \tag{9}$$

where $|Q|$ is the number of terms in query Q and α is a parameter that balances the original term weighting and the regularized term weighting. If a query term is more semantically related to the query, f_{sim} would decrease the term weighting so that the term and its related terms would not be over-emphasized by the retrieval functions. f_{sim} does not require the identification of query aspects. Instead, it implicitly assumes that the relations between query aspects can be approximated by the relations between query terms.

Summary: We incorporate the proposed regularization functions, $f_{size}(\cdot)$ or f_{sim} $(\cdot)$, into the four extended retrieval functions shown in Equation (4)-(7). After analyzing the extended retrieval functions, we can show that all of them satisfy the defined retrieval constraint because both regularization functions satisfy

$$f(q_2, Q, s(\cdot)) < f(q_3, Q, s(\cdot)),$$

which leads to $S(Q, D_1) < S(Q, D_2)$ for all extended retrieval functions. It means that the proposed regularization functions can demote the weights of terms that are from a larger query aspect or have more semantically related terms in the query.

5 Experiments

5.1 Experiment Setup

We evaluate the proposed methods on eight representative TREC data sets: the ad hoc data used in the ROBUST track of TREC 2004 (Robust04), the ad hoc data used in the ROBUST track of TREC 2005 (Robust05), the ad hoc data used in TREC7 (Trec7), the ad hoc data used in TREC8 (Trec8), the Web data used in TREC8 (Web), news articles (Ap88-89), technical reports(Doe) and government documents (Fr88-89). We use two types of queries: keyword queries (i.e., title-only) and verbose queries (i.e., description-only). Table 1 shows some statistics of the test sets, including the collection size, the number of documents, the number of queries and average number of terms per keyword query, and average number of terms per verbose query.

The preprocessing only involves stemming with Porter's stemmer. No stop words is removed for two reasons: (1) A robust retrieval model should be able to discount the stop words appropriately; (2) Removing stop words would introduce at least one extra parameter, i.e., the number of stop words into the experiments. The performance is measured in terms of MAP (mean average precision).

We now explain the notations for different methods. BL is the original retrieval function without regularization. f_{size} and f_{sim} denote the aspect size based and semantic similarity based regularization functions respectively. As shown in Equation (4)-(7), we can integrate the proposed functions into four retrieval functions, i.e., Pivoted (**Piv.**), Okapi BM25 (**Okapi**), Dirichlet Prior (**Dir.**) and axiomatic function (**AX**). Okapi is known to perform poorly for verbose queries [17,4], so instead we report the performance of the modified Okapi (**Mod. Okapi**) [4], which is a stronger baseline for verbose queries. Each of

Table 1. Statistics of Test Collections

Collection	Size	#d	#q	#t/kq	#t/vq	Collection	Size	#d	#q	#t/kq	#t/vq
Robust04	2GB	528K	249	2.75	15.5	Robust05	3GB	1,033K	50	2.7	17.5
Trec7	2GB	528K	50	2.5	14.3	Trec8	2GB	528K	50	2.42	15.8
Web	2GB	247K	50	2.42	15.8	Ap88-89	491MB	165K	146	3.76	18.1
Doe	184MB	226K	35	3.69	18.8	Fr88-89	469MB	204K	42	3.5	19.6

these retrieval functions has one retrieval parameter, and we tune the retrieval parameters as well as the new parameters introduced in the proposed methods and report the optimal performance. In all the tables, ‡ and † indicate that the improvement is statistically significant according to the Wilcoxin signed rank test at the level of 0.05 and 0.1 respectively.

5.2 Comparison of Proposed Methods

Table 2 shows the optimal performance comparison of the proposed methods for verbose queries. Clearly, both proposed term regularization functions can significantly and consistently improve the retrieval performance for all four retrieval functions over almost all of the eight data collections. Moreover, f_{sim} performs better than f_{size}. After analyzing the results, we find that the performance difference might be caused by the fact that f_{sim} is not as dependent to the accuracy of aspect identification as f_{size}. The aspect identification method used in this paper can correctly identify aspects for some queries but not for

Table 2. Performance Comparison for Verbose Queries (MAP)

Function		**Robust04**	**Robust05**	**Trec7**	**Trec8**	**Web**	**Ap88-89**	**Doe**	**Fr88-89**
Piv.	BL	0.2145	0.1406	0.1461	0.2032	0.2122	0.1931	0.1031	0.1424
	f_{size}	**0.2277**‡	0.1408	**0.1706**‡	**0.2208**‡	**0.2481**‡	0.1931	**0.1237**‡	**0.1854**‡
	f_{sim}	**0.2422**‡ (+13.1%)	**0.1517**‡ (+7.8%)	**0.1782**‡ (+21.9%)	**0.2336**‡ (+15.3%)	**0.2555**‡ (+20.8%)	**0.2047**‡ (+6.22%)	**0.1216**‡ (+17.3%)	**0.1637**‡ (+15.5%)
Mod. Okapi	BL	0.2114	0.1391	0.1527	0.2014	0.2371	0.1812	0.1037	0.1526
	f_{size}	**0.2404**‡	0.1428	**0.1785**‡	**0.2311**‡	**0.2758**‡	0.1821	**0.1173**	**0.1942**‡
	f_{sim}	**0.2532**‡ (+12.9%)	**0.1547** ‡ (+8.4%)	**0.1843**‡ (+17.2%)	**0.2408**‡ (+12.0%)	**0.2814**‡ (+17.1%)	**0.1936**‡ (+5.91%)	**0.1141**‡ (+10.2%)	**0.1772**‡ (+13.3%)
Dir.	BL	0.2326	0.1598	0.1811	0.2279	0.2693	0.1990	0.1253	0.1522
	f_{size}	**0.2448**‡	0.1598	**0.1902**‡	**0.2393**‡	**0.2938**‡	0.1990	0.1253	**0.1856**‡
	f_{sim}	**0.2578**‡ (+10.3%)	**0.1623** (+1.25%)	**0.1971**‡ (+7.64%)	**0.2507**‡ (+10.1%)	**0.2946**‡ (+8.39%)	0.1990 (+0%)	**0.1270** (+1.6%)	**0.1801**‡ (+13.9%)
AX	BL	0.2421	0.1612	0.1864	0.2357	0.2715	0.2016	0.1161	0.1674
	f_{size}	**0.2531**‡	0.1612	**0.1881**	**0.2434**‡	**0.2896**‡	0.2016	**0.1245**	**0.2013**‡
	f_{sim}	**0.2534**‡ (+4.5%)	**0.1620** (+1.24%)	**0.1904**† (+2.15%)	**0.2446**‡ (+3.81%)	**0.2866**‡ (+5.51%)	**0.2071**‡ (+4.55%)	**0.1232** (+6.03%)	**0.1958**‡ (+17.4%)

Table 3. Performance Comparison for Keyword Queries (MAP)

Function		**Robust04**	**Robust05**	**Trec7**	**Trec8**	**Web**	**Ap88-89**	**Doe**	**Fr88-89**
Piv.	BL	0.2406	0.1999	0.1762	0.2438	0.2883	0.2267	0.1788	0.2183
	f_{sim}	**0.2432**‡	0.1999	**0.1780**†	**0.2442**	**0.2892**	0.2267	0.1788	**0.2205**
Okapi	BL	0.2477	0.2013	0.1857	0.2512	0.3105	0.2255	0.1847	0.2247
	f_{sim}	**0.2507**‡	0.2013	**0.1892**‡	**0.2518**	0.3105	0.2255	0.1847	**0.2267**

Table 4. Performance Comparison for Verbose Queries on Robust04

Function		**MAP**	**Prec@5**	**Prec@10**	**Function**		**MAP**	**Prec@5**	**Prec@10**
Piv.	BL	0.2145	0.4610	0.3964	**Dir.**	BL	0.2326	0.4554	0.4032
	f_{sim}	**0.2422**‡	**0.4956**‡	**0.4241** ‡		f_{sim}	**0.2578**‡	**0.4859**‡	**0.4237**
Mod. Okapi	BL	0.2114	0.4827	0.4068	**AX**	BL	0.2421	0.4859	0.4253
	f_{sim}	**0.2532**‡	**0.5044**‡	**0.4293**‡		f_{sim}	**0.2534** ‡	**0.4956**	**0.4281**

all queries. Thus, the performance of f_{size} may be affected more by the inaccurate aspect identification, which leads to the relatively worse performance of f_{size}. Table 3 shows the results of two functions for keyword queries, and the results for the other two functions are similar and not included due to the space limit. Clearly, the performance improvement for keyword queries is not as significant and consistent as for verbose queries. Our result analysis suggests that the smaller improvement is caused by the smaller number of terms in keyword queries, because the keyword queries often lead to the smaller query aspects and incorrect aspect identifications for some short queries.

Table 4 shows the performance of the proposed f_{sim} method for verbose queries on ROBUST04. In addition to MAP, we also report the performance measured with Prec@5 and Prec@10. We compare our results with the results of another recently proposed retrieval method for verbose queries [1]. The MAP of their baseline method for verbose queries is 0.2450, and the MAP of their proposed method is 0.2620. Prec@5 of their baseline method is 0.4726 and the Prec@5 of their proposed method is 0.4854. Due to the different pre-processing strategies and different baseline functions, these numbers cannot be directly compared with the results reported in this paper. However, it is quite encouraging to see that our proposed methods can achieve comparable performance with much less computational cost and without the use of external resources.

Finally, we conduct an additional set of experiments when stop words are removed in the pre-processing stage. With the stop word removal, the performance (MAP) of BL for the representative retrieval functions on Robust04 are 0.2196 (Piv.), 0.2351 (Mod. Okapi), 0.2323 (Dir.) and 0.2473 (AX) respectively. Clearly, the results show that stopword removal can improve retrieval performance a little bit but not that much, which suggests that the baseline method we used in the performance comparison is a strong baseline.

5.3 Parameter Sensitivity

As indicated in Equation (8)-(9), f_{sim} has one parameter α and f_{size} has two parameters α and β. Figure 1 shows the sensitivity curve for these parameters on Robust04. $\alpha = 0$ means that no regularization is used. The better performance

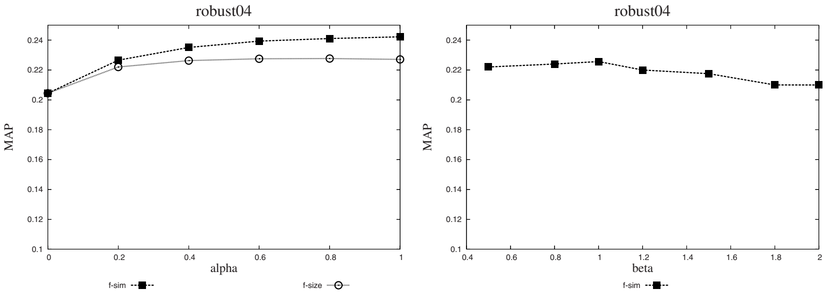

Fig. 1. Performance Sensitivity: α in f_{sim} and f_{size} (Left) and β in f_{size} (Right)

at larger values of α indicates that the proposed term regularization methods are effective. We also observe that the performance is not very sensitive to the values of β. We only show results on one data set due to the space limit. However, the results are similar for other data sets. In general, setting $\alpha = 0.6$ and $\beta = 1$ would lead to good performance for most data collections.

6 Conclusions and Future Work

In this paper, we study the problem of exploiting semantic relations among query terms to regularize term weighting in retrieval functions. Assuming that groups of semantically related query terms form query aspects, we present a general strategy that can systematically incorporate a term weighting regularization function into a variety of existing retrieval functions. Specifically, we propose two term regularization functions based on term semantic similarity, and then discuss how to integrate the regularization functions into existing retrieval functions through the inductive definition scheme. The proposed methods are incorporated into four representative retrieval functions and evaluated on eight representative TREC retrieval collections. Experiment results show that, for verbose queries, the proposed methods can significantly improve the retrieval performance of all the four retrieval functions across almost all the eight test collections. Note that the proposed methods do not require training data and external resources, and the computational cost is low when the MI values are stored in the index offline.

There are several interesting future research directions. First, we will explore other aspect identification methods and compute term semantic similarity using other resources, such as WordNet, query logs and external collections. Second, the semantic relations among query terms could be different for different domains. It would be interesting to explore whether the proposed methods work well in some specific domains, such as biomedical literature search or legal document search. Finally, we focus on only semantic relations among query terms in this work. It would be interesting to study the combination of query term semantic relations, semantic term matching between query and documents, the proximity of different query terms and any other reasonable properties of terms.

References

1. Bendersky, M., Croft, W.B.: Discovering key concepts in verbose queries. In: Proceedings of SIGIR 2008 (2008)
2. Buckley, C.: Why current ir engines fail. In: Proceedings of SIGIR 2004 (2004)
3. Croft, W., Turtle, H., Lewis, D.: The use of phrases and structured queries in information retrieval. In: Proceedings of SIGIR 1991 (1991)
4. Fang, H., Tao, T., Zhai, C.: A formal study of information retrieval heuristics. In: Proceedings of SIGIR 2004 (2004)
5. Fang, H., Zhai, C.: An exploration of axiomatic approaches to information retrieval. In: Proceedings of SIGIR 2005 (2005)
6. Fang, H., Zhai, C.: Semantic term matching in axiomatic approaches to information retrieval. In: Proceedings of SIGIR 2006 (2006)

7. Fuhr, N.: Probabilistic models in information retrieval. The Computer Journal 35(3), 243–255 (1992)
8. Harman, D., Buckley, C.: Sigir 2004 workshop: Ria and where can ir go from here? SIGIR Forum 38(2) (2004)
9. Jones, R., Rey, B., Madani, O., Greiner, W.: Generating query substitutions. In: Proceedings of WWW 2006 (2006)
10. Kumaran, G., Allan, J.: A case for shorter queries, and helping users create them. In: Proceedings of HLT 2006 (2006)
11. Lease, M.: An improved markov rndom field model for supporting verbose queries. In: Proceedings of SIGIR 2009 (2009)
12. Liu, S., Liu, F., Yu, C., Meng, W.: An effective approach to document retrieval via utilizing wordnet and recognizing phrases. In: Proceedings of SIGIR 2004 (2004)
13. Metzler, D., Croft, W.B.: A markov random field model for term dependencies. In: Proceedings of SIGIR 2005 (2005)
14. Mitra, M., Buckley, C., Singhal, A., Cardie, C.: An analysis of statistical and syntactic phrases. In: Proceedings of RIAO 1997 (1997)
15. Ponte, J., Croft, W.B.: A language modeling approach to information retrieval. In: Proceedings of the ACM SIGIR 1998, pp. 275–281 (1998)
16. Risvik, K.M., Mikolajewski, T., Boros, P.: Query segmentation for web search. In: Proceedings of the 2003 World Wide Web Conference (2003)
17. Robertson, S., Walker, S.: On relevance weights with little relevance information. In: Proceedings of SIGIR 1997, pp. 16–24 (1997)
18. Robertson, S.E., Walker, S., Jones, S., Hancock-Beaulieu, M.M., Gatford, M.: Okapi at TREC-3. In: Proceedings of TREC-3 (1995)
19. Salton, G.: Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer. Addison-Wesley, Reading (1989)
20. Schutze, H., Pedersen, J.O.: A co-occurrence based thesaurus and two applications to information retrieval. Information Processing and Management 33(3), 307–318 (1997)
21. Singhal, A., Buckley, C., Mitra, M.: Pivoted document length normalization. In: Proceedings of SIGIR 1996 (1996)
22. Tao, T., Zhai, C.: An exploration of proximity measures in information retrieval. In: Proceedings of SIGIR 2007 (2007)
23. van Rijbergen, C.J.: A theoretical basis for theuse of co-occurrence data in information retrieval. Journal of Documentation, 106–119 (1977)
24. van Rijsbergen, C.J.: Information Retrieval. Butterworths (1979)
25. Zhai, C., Lafferty, J.: A study of smoothing methods for language models applied to ad hoc information retrieval. In: Proceedings of SIGIR 2001 (2001)