An Axiomatic Approach to IR – UIUC TREC 2005 Robust Track Experiments

Hui Fang ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign

Abstract

In this paper, we report our experiments in the TREC 2005 Robust Track. Our focus is to explore the use of a new axiomatic approach to information retrieval.

Most existing retrieval models make the assumption that terms are independent of each other. Although such simplifying assumption has facilitated the construction of successful retrieval systems, the assumption is not true; words are related by use, and their similarity of occurrence in documents can reflect underlying semantic relations between terms. Our new method aims at incorporating term dependency relations into the axiomatic retrieval model in a natural way. In this paper, we describe the method and present analysis of our Robust-2005 evaluation results. The results show that the proposed method works equally well as the KL-divergence retrieval model with a mixture model feedback method. The performance can be further improved by using the external resources such as Google.

1 Introduction

In our recent study [3], we proposed an axiomatic approach to information retrieval, where the relevance is modeled by term-based constraints. Several retrieval functions were derived by using this approach. It has been shown that the performance of the derived function is less sensitive to the parameter setting than the existing retrieval functions with comparable optimal performance; using a fixed parameter value can often achieve near-optimal performance in many test sets. It is thus interesting to evaluate such a new retrieval function in the context of the robust track.

As in most existing retrieval functions, one drawback of the derived new retrieval function is that it makes the assumption that terms are independent of each other. Although such simplifying assumption has facilitated the construction of successful retrieval systems, the assumption is not true; words are related by use, and their similarity of occurrence in documents can reflect underlying semantic relations between terms. Thus, it would be interesting to study how to naturally incorporate such semantic relations into the axiomatic retrieval model.

In this paper, we explain how to extend the existing axiomatic model to take into consideration of the term semantic relations and demonstrate that such extension can also be regarded as a way to do feedback in the axiomatic framework. In both our preliminary experiments with last year's data and the official Robust05 experiments, this method has worked very well. As a pseudo feedback method, it works equally well as the mixture language model feedback method, but it can achieve much better performance when using external resources, such as Google.

The paper is organized as follows. In Section 2, we present the general idea and our new extension for axiomatic approach to information retrieval. In Section 3, we discuss how to implement the proposed extension. In Section 4, we analyze the results of our experiments on Robust-2004 and Robust-2005 data. Finally, we conclude with Section 5.

2 Axiomatic Approach to Information Retrieval

The basic idea of this axiomatic approach is to search in a space of candidate retrieval functions for one that can satisfy a set of reasonable retrieval constraints. To define an axiomatic framework for information retrieval, we need to define (1) a *search space* of possible retrieval functions; and (2) a set of *retrieval constraints* that any reasonable retrieval formula should satisfy. The assumption is that if a retrieval function satisfies all our constraints, the function would likely be effective empirically.

Our recent study [2] demonstrated that the retrieval constraints can be defined as formalized retrieval heuristics. And it demonstrates that the empirical performance of a retrieval function is tightly related to how well it satisfies these constraints. It is also shown that none of the analyzed

retrieval formula can satisfy all the proposed constraints unconditionally. In this paper, we use the constraints defined in the previous work [3] and focus on the other important component in the framework (i.e. function space).

The function space must be large enough to include effective retrieval functions, yet small enough for search. So there is clearly a tradeoff. Following the current retrieval models, we assume that both the documents and the queries are a "bag of terms". Formally, let T be the set of all terms. Let query $Q = \{q_1, ..., q_n\}$ and document $D = \{d_1, ..., d_m\}$ be two bags of terms, where $q_i, d_i \in T$, and it is possible that $q_i = q_j$ and $d_i = d_j$ even if $i \neq j$. Our goal is to define a scoring function $S(Q, D) \in \Re$. To help us search through this function space efficiently and define meaningful constraints on the retrieval functions, we proposed to define a retrieval function inductively [3]. Such inductive definition allows us to decompose a retrieval function into the following three subcomponent functions.

$$\begin{aligned} S(\{q\}, \{d\}) &= f(q, d) \\ S(Q \cup \{q\}, D) &= g(S(Q, D), S(\{q\}, D), q, Q, D) \\ S(Q, D \cup \{d\}) &= h(S(Q, D), S(Q, \{d\}), d, Q, D) \end{aligned}$$

Function f gives the score of a one-term document and a one-term query and is referred to as the *Primitive weighting function*. Function g describes the score change when we add a term to a query, and is called the *Query growth function*. When a new term q is added to a query Q, the score of any document for the new query (i.e. $S(Q \cup \{q\}, D)$) would be mainly determined by the score of the document for the old query (i.e. $S(Q, D)$), the score of the document for the added query term (i.e. $S(\{q\}, D)$), and any possible score adjustment determined by D, Q and q. Similarly, function h describes the score change when we add a term to a document, and is called the *Document growth function*. Clearly, the performance of the axiomatic retrieval model depends on how we instantiate the three component functions. We discussed several possibilities and derived six well-performed retrieval formulas in the previous work [3].

However, one drawback of the derived new retrieval function is that it makes the assumption that terms are independent of each other. The *Primitive weighting function* was defined as

$$S(\{q\}, \{d\}) = f(q, d) = \begin{cases} weight(q) = weight(d) & q = d \\ penalty(q, d) & q \neq d \end{cases}$$

It rewards the document with a score of $weight(q)$ when d matches q and gives it a penalty score of $penalty(q, d)$ otherwise. Even if d is a synonym of q, the retrieval function still penalizes the documents containing d. In other words, the semantic relations between terms are ignored. Although such simplifying assumption has facilitated the construction of successful retrieval systems, the assumption is not true; words are related by use, and their similarity of occurrence in documents can reflect underlying semantic relations between terms. For example, suppose we have a query with a term "car". Intuitively, a single-term document with a term "vehicle" should have a better score than one with a term "software", even though none of these documents has an exact match with the query term "car".

To break the above limitation, we explored how to incorporate term dependency relations into the axiomatic retrieval model in a principled way. We can re-define the *Primitive weighting function* in a more general way as

$$S(Q, D) = f(q, d) = weight(q) \times sim(q, d).$$

where $sim(q, d)$ is a function that measures the similarity between the two terms q and d , $0 \leq sim(q, d) \leq 1$ and $sim(d, d) = 1$. Such definition would reward the documents not only based on the matching query terms, but also the terms that are related to any query term. In this paper, we adopt the normalized mutual information metric for similarity measurement, which is defined as

$$sim(t_1, t_2) = \frac{MI(t_1, t_2)}{MI(t_1, t_1)} \times \frac{1}{\lambda}$$

where λ is a constant and

$$\begin{aligned} MI(t_1, t_2) &= Pr(t_1, t_2) \times log \frac{Pr(t_1, t_2)}{Pr(t_1) \times Pr(t_2)} \\ &\quad + Pr(\overline{t_1}, t_2) \times log \frac{Pr(\overline{t_1}, t_2)}{Pr(\overline{t_1}) \times Pr(t_2)} \\ &\quad + Pr(t_1, \overline{t_2}) \times log \frac{Pr(t_1, \overline{t_2})}{Pr(t_1) \times Pr(\overline{t_2})} \\ &\quad + Pr(\overline{t_1}, \overline{t_2}) \times log \frac{Pr(\overline{t_1}, \overline{t_2})}{Pr(\overline{t_1}) \times Pr(\overline{t_2})} \end{aligned}$$

$Pr(t)$ is a unigram probability for term t, $Pr(t_1, t_2)$ is the joint probabilities for term t_1 and t_2 to co-occur in the same document, and $Pr(\bar{t})$ is the probability for term t not to occur in a document. All the probabilities can be computed by simply counting the document frequency.

Note that the mutual information can be computed over either internal resources (e.g., the target collection itself) or external resource (e.g., the web data) or both. Therefore, the axiomatic framework allows us to exploit the external resources in a more principle way compared with the existing retrieval models.

3 Implementation Issues

In this section, we focus on how to efficiently implement the extension proposed in the previous section.

Based on the re-defined primitive weighting function, we need to compute the similarity between every term in the document and every term in the query. So, the computational cost is high. To solve this problem, we can focus only on the terms that are most similar to a query term. Therefore, the *Primitive weighting function* becomes

$$S(\{q\},\{d\}) = \begin{cases} weight(q) \cdot \frac{MI(q,d)}{MI(q,q)} \cdot \frac{1}{\lambda} & d \in TopKSim(q) \\ 0 & otherwise \end{cases}$$

where λ is a constant and $TopKSim(q)$ is the set of K most similar terms to the term q according to the function $sim()$.

As discussed in [3], there are several ways to instantiate the three component functions. In this paper, we used the following instantiations.

$$\begin{aligned} S(Q \cup \{q\}, D) &= S(Q, D) + S(q, D) \\ S(Q, D \cup \{d\}) &= \sum_{t \in D \cap Q - \{d\}} S(Q, \{t\}) \lambda(|D|+1, C_t^D) \\ &\quad + S(Q, \{d\}) \cdot \lambda(|D|+1, C_d^D + 1) \\ weight(q) &= (\frac{N}{df(q)})^{0.35} \end{aligned}$$

where $\lambda(x, y) = \frac{y}{(\frac{s}{avdl}x+s)+y}$, $0 \leq s \leq 1$, N is the number of documents in the collection, $df(t)$ is the document frequency of t, and C_t^D is the term count of t in D.

Based on the above instantiations, it is not hard to prove that such a way to incorporate the term dependence information can also be regarded as the following way to do pseudo/relevance feedback.

1. Find the top $K = 1000$ most similar terms for every query term based on sim.
2. Pool all the top K similar terms for all the query terms together.
3. Exclude the terms that are only related to one query term if the query length is larger than 2.
4. Add a certain number of terms from the pool to the original query. If all the terms are excluded in the Step 3, then no term will be added to the original query.
5. Compute the weight of the added terms based on the redefined primitive weighting function.

Step 3 is used to guarantee that the added terms will not be biased by the related terms of one query term. In Step 1, the similarity between terms can be computed over any reasonable resources. We consider the following two choices. First, for each query, we use the original axiomatic function to rank the documents and pool the top documents together as a resource. This method will be referred to as pseudo feedback in axiomatic framework. Second, for each query, we use the results returned by Google and pool the top snippets together as a resource. This method will be referred as web-based feedback in axiomatic framework.

Table 1. Performance Comparison on Robust04

	KL-Divergence		F2EXP	
	MAP	gMAP	MAP	gMAP
No FB	0.0955	0.0591	0.0973	0.0629
Pseudo FB	0.1010	0.0609	0.1173	0.0678
Web-based FB	0.1047	0.0634	0.1544	0.0964

4 Experiments and Results

There were two sub-tasks in this year and last year's Robust track: (1) improve the effectiveness for topics that are known to be difficult; (2) predict the query difficulty. For the first task, we only focus on title-only queries, since they are more similar to the actual queries from the web users. For the second task, we ranked all the topics based on the average relevance score returned by the system and the performance of our submitted runs is around the average among all the runs in this year's track.

Our focus is to improve the effectiveness for title-only queries, so we will only present the experiment results for these queries. In our experiments, we used the Web as an external resource. The idea of using web as external resources to improve the performance is not new. Many previous works in Robust track are along this line [7, 6, 4, 1, 5]. But unlike some of them [4, 5], we did not use any NLP techniques to construct the queries for Google.

4.1 Preliminary Experiments

In the preliminary experiments, we compared the performance of pseudo/web-based feedback methods for KL-divergence [8] and F2EXP [3](one retrieval function in the axiomatic retrieval model). In order to compare the performance for a query set over different collections, we tested the robust05 topics over robust2004 data collection and summarized the results in Table 1.

There is no principled approach to use external resources (i.e. Web-based FB) in KL-divergence method. In our experiment, we first used mixture langauge model feedback method to generate an expanded query over the top K Google snippets and ranked the documents in the target collection based on the expanded query.

As seen from the table, the pseudo feedback method in axiomatic framework works equally well as the mixture language model feedback method, but it can achieve much

Table 2. Performance Comparison (PFB) on Robust05

	KL-Divergence		F2EXP	
	MAP	gMAP	MAP	gMAP
No FB	0.1942	0.1275	0.1924	0.1223
Pseudo FB (#Term=20)	0.2389	0.1398	0.2582	0.1235
Pseudo FB (#Term=50)	0.2550	0.1465	0.2629	0.1272
Pseudo FB (#Term=100)	0.2606	0.1523	0.2639	0.1279

Table 3. Parameter Sensitivity of PFB in Axiomatic Framework on Robust05

	#Term=20	#Term=50	#Term=100
#Doc=20	0.2582	0.2629	0.2639
#Doc=100	0.2401	0.2452	0.2468
#Doc=500	0.2103	0.2137	0.2132

better performance when using external resources, such as Google. It demonstrates that the axiomatic framework can offer a more principled and effective way to incorporate the external resources.

4.2 Experiments on Robust 2005

We conducted three sets of experiments on Robust2005 data set.

First, we examined the parameter sensitivity for the proposed method. As mentioned in Section 3, there are three major parameters in the feedback method for axiomatic framework. The parameters are (1)the number of top documents to be included in the resource; (2) the number of terms to be added to the query; (3) λ in the primitive weighting function. Our preliminary experiments show that the optimal value of λ for pseudo feedback method is 0.5 and the optimal value of λ for web-based feedback method is 0.3. Table 3 summarizes the performance (based on MAP) when we change the value for the other two parameters. It shows that the performance is sensitive to the parameter values and the performance is better when the number of document is equal to 20.

Second, we compared the performance of pseudo feedback methods between KL-Divergence and F2EXP. Table 2 summarizes the results when the number of feedback documents is 20. One interesting observation is that the performance of AX+PFB is better than KL+PFB based on MAP, while the performance is worse based on gMAP(i.e. geometric MAP [7]). It might indicate that AX+PFB tends to improve the performance for easy topics instead of difficult ones. But we need to do more experiments and analysis in order to further clarify this.

Table 4. Experiment Results of Web-based FB in Axiomatic Framework on Robust05

Runs	MAP	gMAP
UIUCrAXt0 (No FB)	0.1924	0.1223
UIUCrAXt1 ($\lambda = 0.3$, #Term=20)	0.2677	0.1884
UIUCrAXt2 ($\lambda = 0.5$, #Term=20)	0.2592	0.1827
UIUCrAXt3 ($\lambda = 0.8$, #Term=20)	0.2460	0.1741
UIUCrAXt1 ($\lambda = 0.3$, #Term=20)	0.2677	0.1884
($\lambda = 0.3$, #Term=50)	0.2757	0.1975
($\lambda = 0.3$, #Term=100)	0.2765	0.1987
UIUCrAXt1 ($\lambda = 0.3$, #Term=20)	0.2677	0.1884
UIUCrAXt1 + PFB	0.2767	0.1934

Finally, we report the performance of web-based feedback method in axiomatic framework. Table 4 summarizes the results. Similar to preliminary experiment results, the web-based feedback method can improve the performance significantly. And the performance is sensitive to both the value of λ and the number of added terms. Further performance improvement can be achieved by further expanding the queries, already expanded with the web resource, with a second pseudo-feedback over the target collection.

5 Conclusions

In this paper, we studied how to extend the existing axiomatic model to incorporate the term dependency relations and demonstrated that such extension can also be regarded as a way to do the feedback in the axiomatic framework. In both our preliminary experiments with last year's data and the official Robust05 experiments, this method has worked very well. As a pseudo feedback method, it works equally well as the mixture language model feedback method, but it can achieve much better performance when using external resources, such as Google.

For the future work, we plan to do more experiments on other data sets to find out whether the proposed extension could work consistently well. We will also explore how to tune the parameters for obtaining the optimal results.

References

[1] G. Amati, C. Carpineto, and G. Romano. Fondazione ugo bordoni at trec 2004. In *Proceedings of the Thirteenth Text REtrieval Conference (TREC2004)*, 2005.

[2] H. Fang, T. Tao, and C. Zhai. A formal study of information retrieval heuristics. In *Proceedings of the 2004 ACM SIGIR Conference on Research and Development in Information Retrieval*, 2004.

[3] H. Fang and C. Zhai. An exploration of axiomatic approaches to information retrieval. In *Proceedings of the 2005 ACM SIGIR Conference on Research and Development in Information Retrieval*, 2005.

[4] K. L. Kwok. Trec2004 robust track expeirments using pircs. In *Proceedings of the Thirteenth Text REtrieval Conference (TREC2004)*, 2005.

[5] S. Liu, C. Sun, and C. Yu. Uic at trec-2004. In *Proceedings of the Thirteenth Text REtrieval Conference (TREC2004)*, 2005.

[6] E. M. Voorhees. Overview of the trec 2003 robust retrieval track. In *Proceedings of the Twelfth Text REtrieval Conference (TREC2003)*, 2004.

[7] E. M. Voorhees. Overview of the trec 2004 robust retrieval track. In *Proceedings of the Thirteenth Text REtrieval Conference (TREC2004)*, 2005.

[8] C. Zhai and J. Lafferty. Model-based feedback in the KL-divergence retrieval model. In *Tenth International Conference on Information and Knowledge Management (CIKM 2001)*, pages 403–410, 2001.