Diagnostic Evaluation of Information Retrieval Models

HUI FANG
University of Delaware
TAO TAO
Microsoft Corporation
CHENGXIANG ZHAI
University of Illinois at Urbana-Champaign

Developing effective retrieval models is a long-standing central challenge in information retrieval research. In order to develop more effective models, it is necessary to understand the deficiencies of the current retrieval models and the relative strengths of each of them. In this article, we propose a general methodology to *analytically* and *experimentally* diagnose the weaknesses of a retrieval function, which provides guidance on how to further improve its performance. Our methodology is motivated by the empirical observation that good retrieval performance is closely related to the use of various retrieval heuristics. We connect the weaknesses and strengths of a retrieval function with its implementations of these retrieval heuristics, and propose two strategies to check how well a retrieval function implements the desired retrieval heuristics. The first strategy is to formalize heuristics as constraints, and use constraint analysis to analytically check the implementation of retrieval heuristics. The second strategy is to define a set of relevance-preserving perturbations and perform diagnostic tests to empirically evaluate how well a retrieval function implements retrieval heuristics. Experiments show that both strategies are effective to identify the potential problems in implementations of the retrieval heuristics. The performance of retrieval functions can be improved after we fix these problems.

Categories and Subject Descriptors: H.3.3 [**Information Search and Retrieval**]: Retrieval models

General Terms: Algorithms, Experimentation, Measurement

Additional Key Words and Phrases: Retrieval heuristics, constraints, formal models, TF-IDF weighting, diagnostic evaluation

Note: part of the material appeared in Proceedings of ACM SIGIR 2004.
Authors addresses: H. Fang, Department of Electrical and Computer Engineering, University of Delaware, Newark, DE 19716; email: hfang@ece.udel.edu; T. Tao, Microsoft Corporation, Redmond, WA 98052; email: taotao@microsoft.com; C. Zhai, Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801; email: czhai@cs.uiuc.edu.

1. INTRODUCTION

The study of retrieval models is central to information retrieval. Many different retrieval models have been proposed and tested, including vector space models [Salton et al. 1975; Salton and McGill 1983; Salton 1989; Singhal et al. 1996], probabilistic models [Robertson and Sparck Jones 1976; van Rijbergen 1977; Turtle and Croft 1991; Fuhr 1992; Ponte and Croft 1998; Lafferty and Zhai 2003; Amati and Rijsbergen 2002], and logic-based models [van Rijsbergen 1986; Wong and Yao 1995; Fuhr 2001]. Despite this progress in the development of formal retrieval models, none of the state of the art retrieval functions can outperform other functions consistently, and seeking an optimal retrieval model remains a difficult long-standing challenge in information retrieval research. For example, it has been more than a decade since the Okapi (BM25) retrieval function was proposed [Robertson and Walker 1994; Robertson et al. 1995], but we still have not been able to find another retrieval function that is consistently more robust and effective than Okapi.

A retrieval function is typically evaluated using standard test collections and evaluation measures such as Mean Average Precision (MAP) and precision at 10 documents, which generally reflect the utility of a retrieval function. Unfortunately, such an evaluation methodology provides little *explanation* for the performance differences among retrieval functions. For example, comparing two retrieval functions based on MAP, we know which function gives an overall better ranking of documents on a particular data set, but it is hard to identify the underlying causes of such performance difference. The state of the art retrieval functions, when optimized, usually have similar MAP values even though their function forms are different and their retrieval results for the same query also tend to differ. This suggests that all the functions may have their own (potentially different) weaknesses and strengths. Clearly, in order to further improve the current generation of retrieval models, it is necessary to understand their weaknesses, and ideally, pinpoint specific components in a retrieval function that hinder its performance so that we can improve the function accordingly [Singhal et al. 1996; Singhal et al. 1998]. Thus, a very interesting and important research question is how to design a new evaluation methodology to help identify the strengths and weaknesses of retrieval functions.

In this article, we present a novel methodology to *analytically* and *experimentally* diagnose the weaknesses of a retrieval function, and pinpoint its components that need to be modified in order to further improve its retrieval performance. The methodology can also be used to compare multiple retrieval functions to identify relative strengths and weaknesses of each function so that we can gain insights about how to combine the strengths of different retrieval functions.

The motivation of our work comes from the empirical observation that good retrieval performance is closely related to the use of various retrieval heuristics, especially TF-IDF weighting and document length normalization. Virtually all the empirically effective retrieval formulas tend to boil down to an explicit or implicit implementation of these retrieval heuristics, even though they may be motivated quite differently (see, e.g., many experiment results reported in TREC (http://trec.nist.gov/)). It thus appears that these heuristics are somehow *necessary* for achieving good retrieval performance. However, different retrieval functions implement the heuristics differently, and it is unclear at all whether one implemen-

tation of a heuristic is better or worse than the others. For example, monotonic transformation of a component, such as different normalizations of TF, can easily lead to substantially inferior performance [Salton and Buckley 1988; Zobel and Moffat 1998]. Our main idea is, thus, to formalize these retrieval heuristics and further design tests to evaluate how well a retrieval function implements a retrieval heuristic, through both analytical analysis and empirical experiments.

Specifically, we first define a set of basic desirable constraints to capture formally what are exactly the necessary heuristics. We assume that any reasonable retrieval function should satisfy these constraints. We then check these constraints on a variety of retrieval formulas, which respectively represent the vector space model (pivoted normalization [Singhal et al. 1996]), the classic probabilistic retrieval model (Okapi [Robertson and Walker 1994]), the language modeling approach (Dirichlet prior [Zhai and Lafferty 2001b]), and the divergence from randomness approach (PL2 [Amati and Rijsbergen 2002]). We find that none of these retrieval formulas satisfies all the constraints unconditionally, though some formulas violate more constraints or violate some constraints more "seriously" than others. Empirical results show that when a constraint is not satisfied, it often indicates non-optimality of the method, and when a constraint is satisfied only for a certain range of parameter values, its performance tends to be poor when the parameter is out of the range. In general, we find that the empirical performance of a retrieval formula is tightly related to how well it satisfies these constraints. Thus the proposed constraints provide a good explanation of many empirical observations about retrieval methods, pinpoint the weaknesses of a retrieval function *analytically*, and suggest how we may further improve a retrieval function based on the constraints analysis.

Unfortunately, because we require the constraints to be necessary conditions that a retrieval function should satisfy, the number of constraints that can be defined in this way is inevitably limited. Thus if all the retrieval functions to be compared satisfy all the defined retrieval constraints, or they all satisfy the same set of constraints, constraint analysis alone would not be very helpful to understand the limitation of a function or the relative strengths and weaknesses of each function.

To address this limitation, we further propose an approach to diagnose the weaknesses of retrieval functions *experimentally*. Our main idea is to carefully design a set of diagnostic tests to amplify the differences of performance among retrieval functions under different conditions designed to capture various retrieval heuristics. Specifically, we first define a set of relevance-preserving collection perturbation operators as the basic tools for diagnostic tests. Such collection perturbations would create "extreme conditions" of data sets so as to amplify the differences among the retrieval functions in their effectiveness in handling the extreme conditions, which are designed based on specific retrieval heuristics such as document length normalization. We present a common procedure for designing diagnostic tests for retrieval models based on the defined perturbation operators. Following the proposed procedure, we design a group of diagnostic tests to examine different aspects of retrieval functions, including robustness in handling variations of document lengths, resistance to noisy terms, and appropriate balance of term frequency and length normalization.

Empirical results demonstrate several benefits of the proposed diagnosis method-

ology. First, it can reveal several clear differences among retrieval functions that cannot be revealed through constraint analysis or regular Cranfield-style evaluation. Second, as in the case of constraint analysis, empirical diagnosis also helps identify specific strengths and weaknesses of retrieval functions in implementing different retrieval heuristics and provides guidance on how to modify a retrieval function or combine different retrieval functions to achieve better performance. Based on such analysis of representative state-of-the-art retrieval functions, we propose some variants of the existing retrieval functions. Evaluation on eight representative data sets shows that the proposed variants outperform the corresponding original retrieval functions in most cases, indicating the effectiveness of the proposed diagnosis evaluation method in providing guidance for improving existing retrieval functions.

The rest of the paper is organized as follows. We first present seven formally defined retrieval constraints in Section 2. In Section 3, we apply these constraints to a variety of representative retrieval formulas and show that the satisfaction of these constraints is closely related to the empirical performance of a retrieval function. We then present diagnostic tests in Section 4, and show experiment results of diagnostic tests in Section 5. We discuss how to improve retrieval functions based on the results of diagnostic tests in Section 6, and summarize the overall diagnostic evaluation methodology in Section 7. Related work is discussed in Section 8. Finally, we conclude our work and discuss future research directions in Section 9.

2. FORMAL CONSTRAINTS ON RETRIEVAL FUNCTIONS

In this section, we formally define seven intuitive and desirable constraints that any reasonable retrieval formula should satisfy. They capture the commonly used retrieval heuristics, such as TF-IDF weighting, in a formal way, making it possible to apply them to any retrieval formula analytically.

These constraints are motivated by the following observations on some common characteristics of typical retrieval formulas. First, most retrieval methods assume a "bag of words" (more precisely, "bag of terms") representation of both documents and queries. Second, a highly effective retrieval function typically involves a TF part, an IDF part, and a document length normalization part [Salton and Buckley 1988; Zobel and Moffat 1998; Singhal et al. 1996; Hiemstra 2000]. The TF part intends to give a higher score to a document that has more occurrences of a query term, while the IDF part is to penalize words that are popular in the whole collection. The document length normalization is to avoid favoring long documents; long documents generally have more chances to match a query term simply because they contain more words. Finally, different retrieval formulas do differ in their ways of combining all these factors, even though their empirical performances may be similar.

These observations suggest that there are some "basic requirements" that all reasonable retrieval formulas should follow. For example, if a retrieval formula does not penalize common words, then it somehow violates the "IDF requirement," thus can be regarded as "unreasonable." However, some of these requirements may compromise each other. For example, while the TF heuristic intends to assign a higher score to a document that has more occurrences of a query term, the document length normalization component may cause a long document with a higher TF to

receive a lower score than a short document with a lower TF. Similarly, if two documents match precisely one single, but different query term, the IDF heuristic may allow a document with a lower TF to "beat" the one with a much higher TF. A critical question is thus how we can regulate such interactions so that they will all be "playing a fair game." Clearly, in order to answer this question, we must first define what is a "fair game," i.e., we must define what exactly is a *reasonable* retrieval function. To achieve this goal, we propose to characterize a reasonable retrieval formula by listing the desirable constraints that any reasonable retrieval formula must satisfy.

We now formally define seven such desirable constraints. Note that these constraints are *necessary*, but not necessarily sufficient, and should not be regarded as the *only* constraints that we want a retrieval function to satisfy; indeed, it is possible to come up with additional constraints that may also make sense. However, we focus on these seven basic constraints in this paper because they capture the major well-known IR heuristics, particularly TF-IDF weighting and length normalization.

Let us first introduce some notations. We use D to denote a document, Q to denote a query, and q or t to represent a term. $c(t, D)$ is the count of term t in document D. $|D|$ denotes the length of document D. S denotes a retrieval function, and $S(Q, D)$ gives the score of document D with respect to query Q. $td(t)$ denotes any reasonable measure of term discrimination value (usually based on term popularity in a collection). It gives higher weights to more discriminative terms. For example, $td(t)$ can be the Inverse Document Frequency (IDF) of term t.

2.1 Term Frequency Constraints (TFCs)

We define three constraints to capture the desired contribution of term frequency of a term to scoring.

TFC1: Let $Q = \{q\}$ be a query with only one term q. Assume $|D_1| = |D_2|$. If $c(q, D_1) > c(q, D_2)$, then $S(Q, D_1) > S(Q, D_2)$.

TFC2: Let $Q = \{q\}$ be a query with only one term q. Assume $|D_1| = |D_2| = |D_3|$ and $c(q, D_1) > 0$. If $c(q, D_2) - c(q, D_1) = 1$ and $c(q, D_3) - c(q, D_2) = 1$, then $S(Q, D_2) - S(Q, D_1) > S(Q, D_3) - S(Q, D_2)$.

TFC3: Let Q be a query and $q_1, q_2 \in Q$ be two query terms. Assume $|D_1| = |D_2|$ and $td(q_1) = td(q_2)$, where $td(t)$ can be any reasonable measure of term discrimination value. If $c(q_1, D_1) = c(q_1, D_2) + c(q_2, D_2)$ and $c(q_2, D_1) = 0$, $c(q_1, D_2) \neq 0$,$c(q_2, D_2) \neq 0$, then $S(Q, D_1) < S(Q, D_2)$.

The first constraint captures the basic TF heuristic, which gives a higher score to a document with more occurrences of a query term when the only difference between two documents is the occurrences of the query term. In other words, the score of a retrieval formula should increase with the increase in TF (i.e., the first partial derivative of the formula w.r.t. the TF variable should be positive). The second constraint ensures that the increase in the score due to an increase in TF is smaller for larger TFs (i.e., the second partial derivative w.r.t. the TF variable should be negative). Here, the intuition is that the change in the score caused by increasing TF from 1 to 2 should be larger than that caused by increasing TF from 100 to 101. The third constraint implies another desirable property - if two documents have the same total occurrences of all query terms and all the query

terms have the same term discrimination value, a higher score will be given to the document covering more distinct query terms.

2.2 Term Discrimination Constraint (TDC)

We define this constraint to capture the desired term discrimination scoring.
TDC: Let D be a document and $Q = \{q_1, q_2\}$ be a query. Assume there are two documents D_1 and D_2, where $|D_1| = |D_2|$, D_1 contains only q_1 and D_2 contains only q_2. If $td(q_1) > td(q_2)$, then $S(Q, D \cup D_1) > S(Q, D \cup D_2)$.

This constraint implies that we need to penalize the terms popular in the collection. It is essentially the basic constraint of the M-TDC (modified TDC) proposed in [Shi et al. 2005]. Based on TFC2 and TDC, the following constraint can be derived. Let D be a document and Q be a query. If $q_1 \in Q$, $q_2 \in Q$, $q_1 \in D$, $q_2 \in D$, $td(q_1) > td(q_2)$ and $c(q_1, D) \leq c(q_2, D)$, then $S(Q, D \cup \{q_1\}) > S(Q, D \cup \{q_2\})$.

This constraint is a relaxed-formulation of the original TDC defined in [Fang et al. 2004], which might be too strong and over-favor terms with higher term discrimination value. For example, according to the original constraint, given a query "SVM tutorial," a document with 99 occurrences of "SVM" and 1 occurrence of "tutorial" should receive a higher relevance score than another document with 50 occurrences of"SVM tutorial", which is somewhat counter-intuitive.

2.3 Length Normalization Constraints (LNCs)

We define two constraints to quantify the penalty on long documents.
LNC1: Let Q be a query and D_1, D_2 be two documents. If for some word $t \notin Q$, $c(t, D_2) = c(t, D_1) + 1$ but for any other term w, $c(w, D_2) = c(w, D_1)$, then $S(Q, D_1) \geq S(Q, D_2)$.
LNC2: Let Q be a query and $q \in Q$ be a query term. $\forall k > 1$, if D_1 and D_2 are two documents such that $|D_1| = k \cdot |D_2|$, $c(q, D_2) > 0$ and for all terms w, $c(w, D_1) = k \cdot c(w, D_2)$, then $S(Q, D_1) \geq S(Q, D_2)$.

The first constraint says that the score of a document should decrease if we add an extra occurrence of a "non-relevant word" (i.e., a word not in the query). The second constraint intends to avoid over-penalizing long relevant documents, as it says that if a document D has at least one query term, and we concatenate the document with itself k times to form a new document, then the relevance score of the new document should not be lower than the original one. Here, we make the assumption that the redundant issue is not considered.

2.4 TF-LENGTH Constraint (TF-LNC)

We define this constraint to balance term frequency heuristics and length normalization.
TF-LNC: Let $Q = \{q\}$ be a query with only one term q. If D_1 and D_2 are two documents such that $c(q, D_1) > c(q, D_2)$ and $|D_1| = |D_2| + c(q, D_1) - c(q, D_2)$, then $S(Q, D_1) > S(Q, D_2)$.

This constraint regulates the interaction between TF and document length. It ensures that the relevance score would not decrease after adding more query terms to a document. The intuition is that if D_1 is generated by adding more occurrences

of the query term to D_2, the score of D_1 should be higher than D_2.

Based on TF-LNC and LNC1, it is not hard to derive the following constraint: Let $Q = \{q\}$ be a query with only one term q. If D_2 and D_3 are two documents such that $c(q, D_3) > c(q, D_2)$ and $|D_3| < |D_2| + c(q, D_3) - c(q, D_2)$, then $S(Q, D_3) > S(Q, D_2)$.

To see why, assume we have a document D_1 such that $|D_1| = |D_2| + c(q, D_1) - c(q, D_2)$ and $c(q, D_3) = c(q, D_1)$. It is obvious that the only difference between D_1 and D_3 is that D_1 has more occurrences of the non-query terms. According to LNC1, we know that $S(Q, D_3) \geq S(Q, D_1)$. Since $S(Q, D_1) > S(Q, D_2)$ follows from TF-LNC, it is clear that $S(Q, D_3) > S(Q, D_2)$.

This constraint ensures that document D_1, which has a higher TF for the query term, should have a higher score than D_2, which has a lower TF, as long as D_1 is not too much longer than D_2.

Table I. Summary of intuitions for each formalized constraint

Constraints	**Intuitions**
TFC1	to favor a document with more occurrences of a query term
TFC2	to ensure that the amount of increase in score due to adding a query term repeatedly must decrease as more terms are added
TFC3	to favor a document matching more distinct query terms
TDC	to penalize the words popular in the collection and assign higher weights to discriminative terms
LNC1	to penalize a long document(assuming equal TF)
LNC2, TF-LNC	to avoid over-penalizing a long document
TF-LNC	to regulate the interaction of TF and document length

The first four constraints (i.e. TFCs and TDC) are intended to capture the desired scoring preferences when two documents have equal lengths. The other three constraints are applicable when we have variable document lengths.

Table I summarizes the intuitions behind each formalized constraint. In fact, TFC1 can be derived from LNC1 and TF-LNC. We still present TFC1 in the paper, because it is the most intuitive constraint. All the other constraints defined are basic and non-redundant in the sense that none of them can be derived from the others. Formally, suppose $\mathcal{C} = \{C_1, ..., C_n\}$ is a set of retrieval constraints, $\mathcal{S}_i$ is the set of all the retrieval functions satisfying constraint C_i. C_i and C_j are not redundant, if and only if the set differences $\mathcal{S}_i \backslash \mathcal{S}_j$ and $\mathcal{S}_j \backslash \mathcal{S}_i$ are both non-empty, i.e., $|\mathcal{S}_i \backslash \mathcal{S}_j| > 0$, and $|\mathcal{S}_j \backslash \mathcal{S}_i| > 0$.

We must emphasize once again that the constraints proposed in this section are necessary for a reasonable retrieval formula, but not necessarily sufficient, and should not be regarded as the *only* constraints that a reasonable retrieval formula has to satisfy. When any constraint is violated, we know the retrieval function may not perform well empirically, but satisfying all the constraints does not necessarily guarantee good performance.

3. CONSTRAINT ANALYSIS ON REPRESENTATIVE RETRIEVAL FUNCTIONS

In this section, we apply the seven constraints defined in the previous section to four state-of-the-art retrieval functions, which respectively represent the vector space

model (pivoted normalization [Singhal et al. 1996; Singhal 2001]), the classic probabilistic retrieval model (Okapi [Robertson et al. 1995]), the language modeling approach (Dirichlet prior smoothing [Zhai and Lafferty 2001b]), and the divergence from randomness model (PL2 [Amati and Rijsbergen 2002; He and Ounis 2005]). Our goal is to see how well each retrieval formula satisfies the proposed constraints and how closely the constraint analysis results are related to the empirical performance of a retrieval function. As will be shown, it turns out that none of these retrieval formulas satisfies all the constraints *unconditionally*, though some models violate more constraints or violate some constraints more "seriously" than others. The analysis thus suggests some hypotheses regarding the empirical behavior of these retrieval formulas. Furthermore, empirical results show that when a constraint is not satisfied, it often indicates non-optimality of the method, and when a constraint is satisfied only for a certain range of parameter values, its performance tends to be poor when the parameter is out of the range. In general, we find that the empirical performance of a retrieval formula is tightly related to how well it satisfies these constraints. Thus the proposed constraints provide a good explanation of many empirical observations about retrieval methods. More importantly, they make it possible to evaluate any existing or new retrieval formula *analytically* to obtain insights about how we may further improve a retrieval formula.

The following notations will be used in this subsection:

—$c(t, D)$ is the count of term t in the document D.
—$c(t, Q)$ is the count of term t in the query Q.
—N is the total number of documents in the collection.
—$df(t)$ is the number of documents that contain the term t.
—$|D|$ is the length of document D.
—$avdl$ is the average document length.
—$|Q|$ is the length of query Q.
—$c(t, C)$ is the count of term t in the collection C.
—$p(t|C)$ is the probability of a term t given by the collection language model [Zhai and Lafferty 2001b].

3.1 Constraint Analysis

3.1.1 *Pivoted Normalization Method.* In the vector space model, text is represented by a vector of terms. Documents are ranked by the similarity between a query vector and document vectors. The pivoted normalization retrieval formula [Singhal 2001] is one of the best performing vector space retrieval formulas:

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

Table II. Constraint analysis results (Pivoted)

TFCs	TDC	LNC1	LNC2	TF-LNC
Yes	Yes	Yes	Cond.	Cond.

The results of analyzing the pivoted normalization formula are summarized in Table II, where TFCs means TFC1, TFC2, and TFC3. It is easy to prove that TFCs, TDC and LNC1 can be satisfied unconditionally. We now examine some of the non-trivial constraints.

First, let us examine the TF-LNC constraint. Consider a common case when $|D_1| = avdl$. It can be shown that the TF-LNC constraint is equivalent to the following constraint on the parameter s:

$$s \leq \frac{l(c(t,D_1)) - l(c(t,D_2))}{(c(t,D_1) - c(t,D_2)) \times (1 + l(c(t,D_1)))} \times avdl$$

where $l(x) = ln(1 + ln(x))$.

This means that TF-LNC is satisfied only if s is below a certain upper bound. The TF-LNC constraint thus provides an upper bound for s, which is tighter for a larger $c(t, D_1)$. However, when $c(t, D_1)$ is small, TF-LNC constraint does not provide any effective bound for s, since $s \leq 1$.

Finally, we show that the LNC2 leads to an upper bound for parameter s as well. The LNC2 constraint is equivalent to

$$\frac{1 + ln(1 + ln(k \times c(t,D_2)))}{1 - s + s\frac{k \times |D_2|}{avdl}} \geq \frac{1 + ln(1 + ln(c(t,D_2)))}{1 - s + s\frac{|D_2|}{avdl}}$$

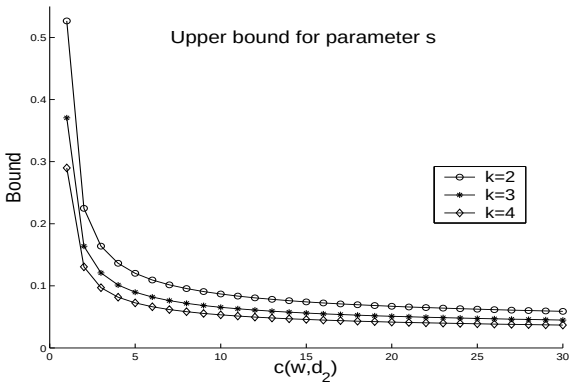

Fig. 1. Upper bound of parameter s.

Therefore, the upper bound of s can be derived as:

$$s \leq \frac{tf_1 - tf_2}{(k\frac{|D_2|}{avdl} - 1)tf_2 - (\frac{|D_2|}{avdl} - 1)tf_1}$$

where $tf_1 = 1 + ln(1 + ln(k \times c(t, D_2))), tf_2 = 1 + ln(1 + ln(c(t, D_2)))$. In order to get a sense of what the bound is exactly, consider a common case when $|D_2| = avdl$. We have

$$s \leq \frac{1}{k-1} \times (\frac{tf_1}{tf_2} - 1).$$

As shown in the Figure 1, the bound becomes tighter when k increases or when the term frequency is larger. This bound shows that in order to avoid over-penalizing a long document, a reasonable value for s should be generally small – it should be below 0.4 even in the case of a small k, and we know that for a larger k the bound would be even tighter. This analysis thus suggests that the performance can be poor for a large s, which is confirmed by our experiments.

3.1.2 *Okapi Method.* The Okapi formula is another highly effective retrieval formula that represents the classical probabilistic retrieval model [Robertson and Walker 1994]. The formula as presented in [Singhal 2001] is [1]

$$S(Q,D) = \sum_{t \in Q \cap D} \left(ln \frac{N - df(t) + 0.5}{df(t) + 0.5} \times \frac{(k_1+1) \times c(t,D)}{k_1((1-b) + b\frac{|D|}{avdl}) + c(t,D)} \times \frac{(k_3+1) \times c(t,Q)}{k_3 + c(t,Q)} \right) \quad (2)$$

where k_1 (between 1.0-2.0), b (usually 0.75), and k_3 (between 0-1000) are constants.

The major difference between Okapi and other retrieval formulas is the possibly negative value of the IDF part, which has been discussed in [Robertson and Walker 1997]. It is trivial to show that if $df(t) > N/2$, the IDF value would be negative.

When the IDF part is positive (which is mostly true for keyword queries), it is easy to see that Okapi method satisfies TFCs and LNCs. By considering a common case when $|D_2| = avdl$, the TF-LNC constraint is shown to be equivalent to $b \leq \frac{avdl}{c(t,D_2)}$. Since b is always smaller than 1, TF-LNC can be satisfied unconditionally. Moreover, we can show that TDC is unconditionally satisfied.

Although Okapi satisfies some constraints conditionally, unlike in the pivoted normalization method, the conditions do not provide any bound for the parameter b. Therefore, the performance of Okapi can be expected to be less sensitive to the length normalization parameter than the pivoted normalization method, which is confirmed by our experiments.

When the IDF part is negative, the Okapi formula would satisfy TDC but violate the TFCs, LNCs and TF-LNC, since matching an additional occurrence of a query term could mean decreasing the score. Since a negative IDF only happens when a query term has a very high document frequency (e.g., when the query is verbose), our analysis suggests that the performance of Okapi may be relatively worse for verbose queries than for keyword queries.

A simple way to solve the problem of negative IDF is to replace the original IDF in Okapi with the regular IDF in the pivoted normalization formula:

$$S(Q,D) = \sum_{t \in Q \cap D} \left(ln \frac{N+1}{df(t)} \times \frac{(k_1+1) \times c(t,D)}{k_1((1-b) + b\frac{|D|}{avdl}) + c(t,D)} \times \frac{(k_3+1) \times c(t,Q)}{k_3 + c(t,Q)} \right)$$

This modified Okapi satisfies all the defined constraints. We thus hypothesize that the modified Okapi would perform better than the original Okapi for verbose queries. As will be shown later, this is indeed true according to our experiment results.

The results of analyzing the Okapi formula are summarized in Table III. We distinguish two forms of the formula – the original formula and the one with a modified IDF part. The modification significantly affects the constraint analysis

[1] There is a typo in the formula in [Singhal 2001], which is corrected here.

Table III. Constraint analysis results (Okapi)

Formula	TFCs	TDC	LNC1	LNC2	TF-LNC
Original	Cond.	Yes	Cond.	Cond.	Cond.
Modified	Yes	Yes	Yes	Yes	Yes

results as discussed above.

3.1.3 *Dirichlet Prior Method.* The Dirichlet prior retrieval method is one of the best performing language modeling approaches [Zhai and Lafferty 2001b]. This method uses the Dirichlet prior smoothing method to smooth a document language model and then ranks documents according to the likelihood of the query according to the estimated language model of each document. With a notation consistent with formulas above, the Dirichlet prior retrieval function is

$$S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \cdot ln(1 + \frac{c(t,D)}{\mu \cdot p(t|C)}) + |Q| \cdot ln\frac{\mu}{|D| + \mu} \quad (3)$$

$p(t|C)$ is similar to the document frequency $df(t)$, and it indicates how popular the term t is in the whole collection.

Table IV. Constraint analysis results (Dirichlet)

TFCs	TDC	LNC1	LNC2	TF-LNC
Yes	Yes	Yes	Cond	Yes

The results of analyzing the Dirichlet prior formula are summarized in Table IV. TFCs, TDC, LNC1 and TF-LNC are easily seen to be satisfied. The LNC2 constraint can be shown to be equivalent to $c(t, D_2) \geq |D_2| \cdot p(t|C)$, which is usually satisfied for content-carrying words. If all the query terms are discriminative words, long documents will not be over-penalized. Thus, compared to pivoted normalization, Dirichlet prior appears to have a more robust length normalization mechanism, even though none of them satisfies the LNC2 constraint unconditionally.

3.1.4 *PL2 Method.* The PL2 method is a representative retrieval function of the divergence from randomness (DFR) retrieval model [Amati and Rijsbergen 2002; He and Ounis 2005]. The basic idea of PL2 is to measure the informative content of a term by computing how much the term frequency distribution departs from a distribution described by a random process. With the above notations, the PL2 method can be described as

$$S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \cdot \frac{tfn_t^D \cdot log_2(tfn_t^D \cdot \lambda_t) + log_2 e \cdot (\frac{1}{\lambda_t} - tfn_t^D) + 0.5 \cdot log_2(2\pi \cdot tfn_t^D)}{tfn_t^D + 1} \quad (4)$$

where $tfn_t^D = c(t,D) \times log_2(1 + c \cdot \frac{avdl}{|D|})$, $\lambda_t = \frac{N}{c(t,C)}$ and $c > 0$ is a retrieval parameter.

The constraint analysis results of the PL2 method are summarized in Table V. Unlike the previous three retrieval functions, the PL2 method cannot unconditionally satisfy any of the retrieval constraints. In particular, we can make the following two interesting observations.

Table V. Constraint analysis results (PL2)

Functions	TFCs	TDC	LNC1	LNC2	TF-LNC
PL2	Cond	Cond	Cond	Cond	Cond
Mod.-PL2	Yes	Cond	Yes	Cond	Cond

First, the analysis of the TDC, LNC2 and TF-LN constraints suggests a lower bound for the retrieval parameter c in the PL2 method. Specifically, assuming that $|D| = avdl$, we can show that the PL2 function can satisfy TDC if $c > 2^{\frac{log_2 e}{\lambda_t}} - 1$. Moreover, constraints LNC2 and TF-LN can be shown to be equivalent to $c \geq \frac{|D|}{avdl}$, which is $c \geq 1$ under the assumption that $|D| = avdl$. Clearly, the analysis results suggest that PL2 would violate the retrieval constraints for smaller values of c. Thus, we expect it to perform poorly for a smaller c, which is confirmed by our experiments.

Second, the term discrimination values (i.e., λ_t) affect whether the PL2 retrieval function satisfies a retrieval constraint. In particular, it can be shown that the three TF constraints are equivalent to the following constraint on the term discrimination value (λ_t) of term t:

$$\lambda_t * log(\lambda_t) + 0.18 * \lambda_t \geq log_2 e.$$

This means that the three TF constraints are satisfied only when query terms have term discrimination values that are larger than a threshold around 1. For example, Figure 3.1.4 shows two curves of relevance scores (i.e., $S(Q, D)$) with respect to the values of TF component (i.e., $c(t, D) \cdot log_2(1 + c\frac{avdl}{|D|})$) for a query with a relatively discriminative term ($\lambda_t = 2$, shown on left) and one with a common word ($\lambda_t = 0.5$, shown on right), respectively. We see clearly that PL2 violates the defined TF constraints in the right plot where the term discrimination value is small. Moreover, the analysis of LNC1 constraint also provides a similar lower bound for the term discrimination values, i.e., $\lambda_t \geq 1$.

Thus if a query contains a term with smaller term discrimination values, the PL2 function would violate the TF constraints and LNC1 constraint, which means that in such a case, more occurrences of a query term in a document may actually lead to lower relevance scores. This further suggests that the PL2 function would perform worse for verbose queries than for keyword queries, which again is confirmed in our experiments.

The analysis suggests that a simple way to potentially improve the performance of PL2 for verbose queries is to consider only the query terms whose values of λ_t are larger than 1 (the threshold obtained from constraint analysis). The modified PL2 is as follows:

$$S(Q, D) = \sum_{t \in Q \cap D, \lambda_t > 1} c(t, Q) \cdot \frac{tfn_t^D \cdot log_2(tfn_t^D \cdot \lambda_t) + log_2 e \cdot (\frac{1}{\lambda_t} - tfn\ _t^D) + 0.5 \cdot log_2(2\pi \cdot tfn_t^D)}{tfn_t^D + 1}.$$

The only difference from the standard PL2 is the extra condition $\lambda_t > 1$ derived from the analysis of the TF constraints and LNC1 constraint. The motivation of this modified function is to ignore query terms violating the retrieval constraints. This is reasonable since these terms have term discriminative values smaller than 1, i.e., each of these terms has, on average, at least one occurrence in every document

Table VI. Summary of constraint analysis results for different retrieval formulas

Formula	TFCs	TDC	LNC1	LNC2	TF-LNC
Pivoted	Yes	Yes	Yes	C_1^*	C_2^*
Dirichlet	Yes	Yes	Yes	C_3	Yes
Okapi(original)	C_4	Yes	C_4	C_4	C_4
Okapi(modified)	Yes	Yes	Yes	Yes	Yes
PL2(original)	C_5	C_6^*	C_7	C_8^*	C_8^*
PL2(modified)	Yes	C_6^*	Yes	C_8^*	C_8^*

of the collection. The effect of this modified retrieval function is similar to stopwords removal, but the modification is guided by the constraint analysis and the proposed method is more adaptive to the characteristics of different document collections.

The modified PL2 method satisfies the three TFCs and LNC1 constraints unconditionally and the other constraints conditionally. When the retrieval parameter c is set to a value larger than a threshold, the modified PL2 method would satisfy all the constraints. We thus hypothesize that the modified PL2 would perform better than the original PL2 for verbose queries when c is set appropriately based on the guidance provided by the constraint analysis. As will be shown later, this hypothesis is indeed confirmed by our experiment results.

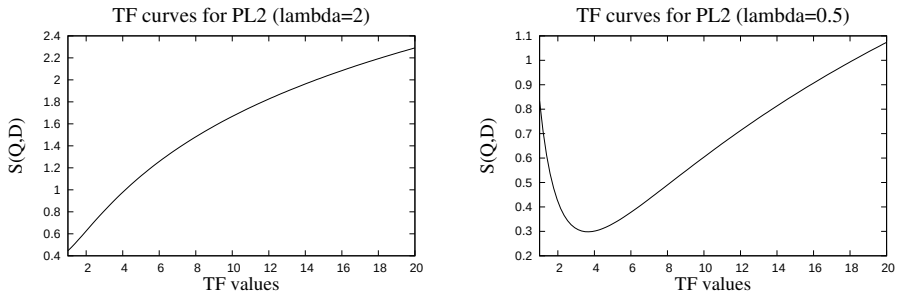

Fig. 2. TF curves are sensitive to term discrimination value: left plot ($\lambda_t = 2$) - TF constraints satisfied; right plot ($\lambda_t = 0.5$) - TF constraints violated

3.1.5 *Summary of Constraint Analysis Results.* We have applied the proposed seven constraints to four representative retrieval formulas. The results are summarized in Table VI, where a "Yes" means that the corresponding model satisfies the particular constraint and a "C_x" means that the corresponding model satisfies the constraint under some particular conditions *not* related to parameter setting, and a "C_x^*" means that the model satisfies the constraint only when the parameter is in some range. The specific conditions are

$$C_1^* \Leftrightarrow s \leq \frac{tf_1 - tf_2}{(k\frac{|D_2|}{avdl} - 1)tf_2 - (\frac{|D_2|}{avdl} - 1)tf_1}$$

$$C_2^* \Leftrightarrow s \leq \frac{(l(c(t,D_1)) - l(c(t,D_2))) \times avdl}{(c(t,D_1) - c(t,D_2)) \times (1 + l(c(t,D_1)))}$$

$$
\begin{aligned}
C_3 &\Leftrightarrow c(t, D_2) \geq |D_2| \cdot p(t|C) \\
C_4 &\Leftrightarrow idf(t) \geq 0 \Leftrightarrow df(t) \leq N/2 \\
C_5 &\Leftrightarrow \lambda_t * log(\lambda_t) + 0.18 * \lambda_t \geq log_2 e \\
C_6^* &\Leftrightarrow c > 2^{\frac{log_2 e}{\lambda_t}} - 1 \\
C_7 &\Leftrightarrow \lambda_t \geq 1 \Leftrightarrow N \geq c(t, C) \\
C_8^* &\Leftrightarrow c \geq \frac{|D|}{avdl}
\end{aligned}
$$

Based on the results, we can make three interesting observations: First, the original IDF part of the Okapi formula causes the formula to violate almost all constraints (see also [Robertson and Walker 1997] for a discussion about this weakness), thus we may expect Okapi to have worse performance for verbose queries. Second, the implementation of term discrimination part in PL2 (i.e., λ_t) causes it to violate several constraints for non-discriminative terms, thus we may also expect PL2 to perform worse for verbose queries. Third, C_1 and C_2 provide an approximate upper bound for the parameter s in the pivoted normalization method, while C_6 and C_8 provide a lower bound for the parameter c in the PL2 method. In contrast, however, by checking the constraints, we have not found any particular bound for the parameters in Dirichlet Prior and Okapi. Therefore, we may expect the pivoted normalization method and PL2 method to be more sensitive to the parameter setting than the other two methods. As we will further discuss in the next section, these predictions have been mostly confirmed in our experiments.

3.2 Benefits of Constraint Analysis

In the previous subsection, we have examined four representative retrieval formulas analytically. Based on the analysis, we propose some hypotheses about the performance for each retrieval formula. In this section, we test these hypotheses through carefully designed experiments. Our experiment results show that the proposed constraints can explain the performance difference in various retrieval models, provide an approximate bound for the parameters in a retrieval formula, and enable us to *analytically* diagnose the weaknesses of a retrieval function to obtain guidance on how to improve a retrieval function.

3.2.1 *Experiment Design.* Retrieval performance can vary significantly from one test collection to another. We thus construct several very different and representative test collections using the existing TREC test collections. To cover different types of queries, we follow [Zhai and Lafferty 2001b] , and vary two factors: query length and verbosity, which gives us four different combinations : short-keyword (SK, keyword title), short-verbose (SV, one sentence description), long-keyword (LK, keyword list), and long-verbose (LV, multiple sentences). The number of queries is usually larger than 50. To cover different types of documents, we construct our document collections by varying several factors, including (1) the type of documents; (2) document length; (3) collection size; and (4) collection homogeneity. Our choice of document collection has been decided to be news articles (AP), technical reports (DOE), government documents (FR), a combination of AP,

Table VII. Document set characteristic

	AP	DOE	FR	ADF	Web	Trec7	Trec8
#qry	142	35	42	144	50	50	50
#rel/q	103	57	33	126	46	93	95
size	491MB	184MB	469MB	1GB	2GB	2GB	2GB
#doc(k)	165K	226K	204K	437K	247K	528K	528K
#voc(k)	361K	163K	204K	700K	1968K	908K	908K
mean(dl)	454	117	1338	372	975	477	477
dev(dl)	239	58	5226	1739	2536	789	789
mean(rdl)	546	136	12466	1515	6596	1127	1325

Table VIII. Optimal s (for average precision) in the pivoted normalization method

	AP	DOE	FR	ADF	Web	Trec7	Trec8
lk	0.2	0.2	0.05	0.2	—	—	—
sk	0.01	0.2	0.01	0.05	0.01	0.05	0.05
lv	0.3	0.3	0.1	0.2	0.2	0.2	0.2
sv	0.2	0.3	0.1	0.2	0.1	0.1	0.2

DOE, and FR (ADF), the Web data used in TREC8 (Web), the ad hoc data used in TREC7 (Trec7) and the ad hoc data used in TREC8 (Trec8). Table VII shows some document set characteristics, including the number of queries used on the document set, the average number of relevant documents per query, the collection size, the number of documents, the vocabulary size, the mean document length, the standard deviation of document length, and the mean length of relevant documents.

The preprocessing of documents and queries is minimum, involving only Porter's stemming. We intentionally did not remove stop words for two reasons: (1) A truly robust model should be able to discount the stop words automatically; (2) Removing stop words would introduce at least one extra parameter (e.g. the number of stop words) into our experiments. On each test collection, for every retrieval method, we vary the retrieval parameter to cover a reasonably wide range of values. This allows us to see a complete picture of how sensitive each method is to its parameter. We use mean average precision as the main evaluation measure.

3.2.2 *Parameter Sensitivity.* Based on the analysis in the previous subsection, we formulate the following hypotheses: (1) The pivoted normalization method is sensitive to the value of parameter s, where $0 < s < 1$. The analysis of LNC2 suggests that the reasonable value for s should be generally smaller than 0.4 and the performance can be bad for a large s. (2) Okapi is more stable with the change of parameter b ($0 < b < 1$) compared with the pivoted normalization method. (3) The PL2 method is sensitive to the value of parameter c, where $c > 0$. The constraint analysis suggests that the reasonable values for c should be generally equal to or larger than 1.

We now discuss the experiment results from testing these hypotheses. First, let us consider the experiment result for the pivoted normalization method. As shown in Table VIII, the optimal value of s to maximize average precision is indeed very small in all cases. Moreover, Figure 3 shows how the average precision is influenced by the parameter value in the pivoted normalization method on the AP document

Table IX. Optimal c (for average precision) in the PL2 method

	AP	DOE	FR	ADF	Web	Trec7	Trec8
lk	2	2	10	2	—	—	—
sk	5	5	5	5	15	15	15
lv	5	2	10	2	2	5	5
sv	5	2	10	5	50	7	5

set and long-keyword queries; the curves are similar for all other data sets. Clearly when s is large, which causes the method not to satisfy the LNC2 constraint, the performance becomes significantly worse.

Next, we experiment with the Okapi method. Assume $k_1 = 1.2, k_3 = 1000$ [Singhal 2001] and b changes from 0.1 to 1.0. Okapi is indeed more stable than the pivoted normalization (shown in Figure 3). By checking the constraints, we have not found any particular bound for the parameters in Okapi, which may explain why Okapi is much less sensitive to the parameter setting than the pivoted normalization method where the LNC2 constraint implies a concrete bound on parameter s.

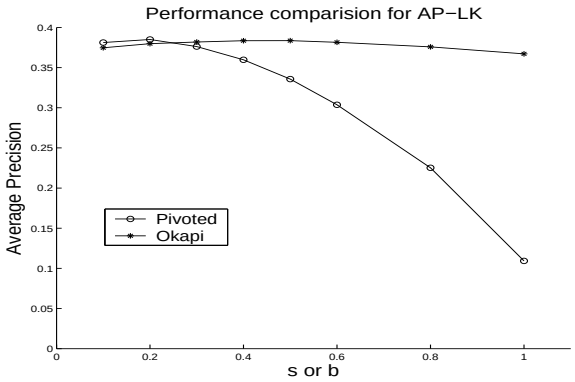

Fig. 3. Performance comparison between Okapi and Pivoted for AP-LK.

We now consider the experiment results for the PL2 method. Table IX shows that the optimal values of c are always larger than 1 in all cases. Moreover, as shown in Figure 4, when the values of c are smaller than 1, which would cause the retrieval function to violate TDC, LNC2 and TF-LNC constraints, the performance indeed becomes significantly worse.

In summary, the constraints generally can provide an empirical bound for the parameters in retrieval formulas and the performance would tend to be poor when the parameter is out of the bound.

3.2.3 *Performance Comparison.* We compare the performance of the four retrieval formulas through systematic experiments. Our goal is to see whether the experiment results are consistent with the analytical results based on formalized heuristics. We form the following hypotheses based on the constraint analysis: (1)

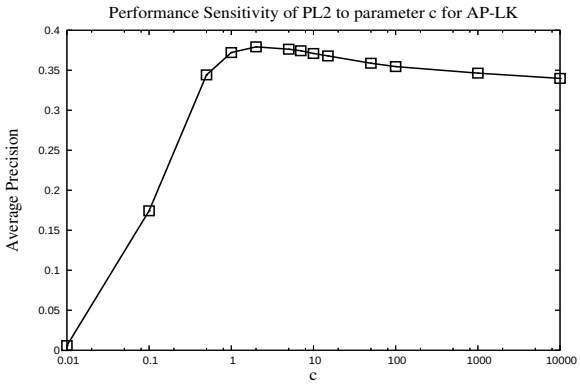

Fig. 4. Performance sensitivity of PL2 to parameter c for AP-LK.

For verbose queries, both Okapi and PL2 would perform worse than pivoted normalization and Dirichlet prior, due to violation of more constraints in these cases. (2) The modified Okapi and modified PL2, which are derived from heuristically modifying the original functions to make them satisfy more constraints in the case of verbose queries, would perform better than their respective original functions.

In order to test these hypotheses, we run experiments over seven collections and four query sets to test all these methods, including pivoted normalization, Dirichlet prior, Okapi, modified Okapi, PL2, and modified PL2. We summarize the optimal performance for each formula in Table X.

We see that, for keyword queries, the optimal performances of all the four retrieval formulas, representing four different types of retrieval models, are comparable, confirming that these state-of-the-art retrieval models are indeed all effective. However, for verbose queries (LV and SV), as we hypothesized, both Okapi and PL2 are substantially worse than pivoted normalization and Dirichlet prior. The fact that both modified Okapi (Mod-Okapi) and modified PL2 (Mod-PL2) are generally much better than their corresponding original functions further shows that the cause of the poor performance on verbose queries for these two functions is indeed due to the violation of constraints. The improvements of Mod-Okapi over Okapi and Mod-PL2 over PL2 are both significant (the p-values of the Wilcoxon signed rank test are all below 0.013).

Specifically, the original Okapi violates many constraints due to possible negative IDF scores in the case of verbose queries, which explains the poor performance of the original Okapi. Mod-Okapi was obtained by replacing the IDF in Okapi with the IDF from pivoted normalization so as to satisfy more constraints (indeed, after this modification, Mod-Okapi satisfies all our constraints), and it indeed improves over the original Okapi substantially for verbose queries. In Figure 5, we compare Mod-Okapi with the original Okapi (together with pivoted normalization) for a range of parameter values, where we see that Mod-Okapi is consistently better than the original Okapi for the entire range of parameter b, and gives performance comparable to that of pivoted normalization for verbose queries.

The case of PL2 is similar. The original PL2 violates many constraints in the

Table X. Comparison of optimal performance for four formulas.

		AP	DOE	FR	ADF	Web	Trec7	Trec8
lk	Piv	0.39	0.28	0.33	0.27	—	—	—
	Dir	0.38	0.28	0.32	0.25	—	—	—
	Okapi	0.38	0.27	0.28	0.33	—	—	—
	Mod-Okapi	0.39	0.28	0.28	0.33	—	—	—
	PL2	0.38	0.27	0.27	0.32	—	—	—
	Mod-PL2	0.38	0.27	0.27	0.32	—	—	—
sk	Piv	0.23	0.18	0.19	0.22	0.29	0.18	0.24
	Dir	0.22	0.18	0.18	0.21	0.30	0.19	0.26
	Okapi	0.23	0.19	0.23	0.19	0.31	0.19	0.25
	Mod-Okapi	0.23	0.19	0.23	0.19	0.31	0.19	0.25
	PL2	0.22	0.19	0.22	0.19	0.31	0.18	0.26
	Mod-PL2	0.22	0.19	0.22	0.19	0.31	0.18	0.26
lv	Piv	0.29	0.21	0.23	0.21	0.22	0.20	0.23
	Dir	0.29	0.23	0.24	0.24	0.28	0.22	0.26
	Okapi	0.03	0.07	0.09	0.06	0.23	0.08	0.11
	Mod-Okapi	0.30	0.24	0.25	0.23	0.28	0.26	0.25
	PL2	0.24	0.20	0.09	0.08	0.09	0.09	0.13
	Mod-PL2	0.29	0.22	0.25	0.20	0.27	0.21	0.25
sv	Piv	0.19	0.10	0.14	0.14	0.21	0.15	0.20
	Dir	0.20	0.13	0.16	0.16	0.27	0.18	0.23
	Okapi	0.08	0.08	0.08	0.09	0.21	0.09	0.10
	Mod-Okapi	0.19	0.12	0.16	0.14	0.25	0.16	0.22
	PL2	0.16	0.09	0.07	0.10	0.11	0.08	0.10
	Mod-PL2	0.19	0.10	0.18	0.15	0.25	0.15	0.21

case of very small term discrimination values (i.e., verbose queries), which explains its poor performance on verbose queries. After modifying the PL2 by excluding the terms whose term discrimination values are smaller than 1.0, the performance on verbose queries is improved consistently and substantially as can be seen by comparing PL2 and Mod-PL2. The performance remains the same for keyword queries since presumably no term in the keyword queries has a term discrimination value (i.e., λ_t) smaller than 1. Note that although removing the stopwords may achieve a similar effect and make the original PL2 perform well for both keyword and verbose queries, we believe that a robust retrieval function should be able to discount the stop words *automatically*, thus this result reveal a weakness of PL2, which can be corrected through a heuristic modification (i.e., Mod-PL2).

Overall, both Table X and Figure 5 show that satisfying more constraints appears to be correlated with better performance. Therefore, the proposed constraints can provide a plausible explanation for the performance difference in various retrieval models, allow us to diagnose the weaknesses of a retrieval function, and use the insights gained from the analysis to further improve a retrieval function.

4. DIAGNOSTIC EVALUATION WITH COLLECTION PERTURBATION

Constraints analysis described in the previous sections provides a principled way to examine the implementations of retrieval heuristics analytically. Unfortunately, if two analyzed retrieval functions satisfy the same set of constraints, constraint analysis would not be able to help us judge which is better. Moreover, analytical

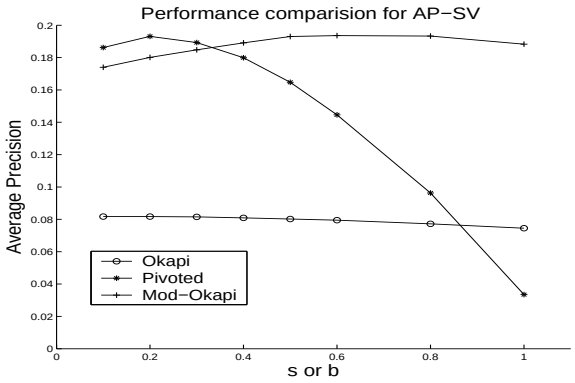

Fig. 5. Performance Comparison between modified Okapi, Okapi and Pivoted for AP-SV.

checking of a constraint can sometimes be mathematically challenging, thus we may not always be able to derive an analytical result. In all these cases, we will still need to rely on test collections to experimentally compare retrieval functions.

The traditional Cranfield evaluation methodology allows us to easily compare two retrieval functions based on their overall retrieval accuracy. However, when one retrieval function has a lower retrieval accuracy than another, such an evaluation method cannot help us understand why. An interesting question is, thus, whether we can design some tests to *experimentally* diagnose the strengths and weaknesses of retrieval functions. For example, can we design a test to examine whether the inferior performance of a retrieval function is due to an inferior length normalization component when compared with another retrieval function?

In this section, we propose a general methodology to compare retrieval functions experimentally to understand how well they implement specific retrieval heuristics. Our main idea is to perturb a standard retrieval test collection to gradually make it approach various "extreme conditions" (e.g., making all the documents have an equal length or amplifying the length differences of documents), and examine how a retrieval function responds to such perturbations. We can then compare the performance patterns exhibited by different retrieval functions under such perturbations. The perturbations will be designed in such a way that a certain aspect of difference in two retrieval functions (e.g., their effectiveness in handling length normalization) would be amplified under a particular extreme condition (e.g., when the variances of document lengths are made artificially very high).

Our idea is essentially similar to medical diagnosis as shown in Table XI. Specifically, we will propose a set of operators for perturbing existing evaluation collections while preserving the relevance status of all documents. Such perturbations make it possible to enlarge the empirical performance differences among retrieval functions and make it easier to observe the "symptoms" of existing retrieval functions (i.e., the problems of current heuristic implementations). We will further design various diagnostic tests targeting at testing specific aspects of a retrieval function. These diagnostic tests are thus similar to medical instruments (e.g., a medical thermometer) in that they allow us to measure specific symptoms of a retrieval function.

Table XI. Medical diagnosis analogy

Medical Domain	**IR Domain**
patients	retrieval functions
illness	non-optimal performance
diseases	problems of heuristic implementation, causes of non-optimal performance
medical records	existing findings in IR
symptoms	empirical results
medical instruments	relevance-preserved collection perturbations
medical tests	tests for retrieval models
treatments for disease	better implementations of heuristics, modification for performance improvement

Through analyzing the results from various tests, we can then diagnose the relative strengths and weaknesses of a retrieval function. We can then "treat" the retrieval function by fixing its weakness through better implementation of retrieval heuristics.

We now describe this diagnosis methodology in more detail.

4.1 Collection Perturbations

One reason why existing evaluation methodology is not informative enough is that a test collection usually has a mixed set of documents with various characteristics. Our idea to perform diagnosis evaluation is thus to perturb an existing evaluation collection to control or vary some characteristics. This would generate a series of perturbed collections, which can then be used to test a retrieval function. We hope the perturbed collections would allow us to see more meaningful differences between retrieval functions. With our medical domain analogy, these perturbations serve as our instruments to perform diagnostic tests for retrieval models.

We first introduce some new notations. $\mathcal{D}$ is the set of all the documents in the test collection, $\mathcal{D}_r$ is the set of relevant documents, and $\mathcal{D}_n$ is the set of non-relevant documents. $\mathcal{V}_n$ is a set of "noise terms," i.e., terms that are not relevant to any query; for example, they can be meaningless terms outside our vocabulary. In other words, $\forall t_n \in \mathcal{V}_n$, if we add t_n to D, the relevance status of D would not be changed. As in the previous sections, we assume that both queries and documents use "bag of terms" representation.

A standard evaluation collection includes a document collection, a set of queries, and a set of relevance judgments indicating which documents are relevant to which queries. To leverage the relevance judgments in the existing test collections, we keep the topics and perturb only the documents, which means to perturb term statistics in documents (e.g., term occurrences), document statistics (e.g., document length), and collection statistics (e.g., number of documents). We define five basic operators for collection perturbations, including term addition, term deletion, document addition, document deletion, and document concatenation; they are summarized in Table XII. Every operator has a parameter K to control the degree of perturbation. K can either be the same for all documents or vary according to term/document statistics, such as the occurrences of a term. These basic operators can be combined to perform more sophisticated perturbations.

Table XII. Basic perturbation operators

Name	Semantic	Operator
term addition	Add K occurrences of term t to document D	$aT(t, D, K)$
term deletion	Delete K occurrences of term t from document D	$dT(t, D, K)$
document addition	Add document D to the collection K times	$aD(D, K)$
document deletion	Delete document D form the collection	$dD(D)$
document concatenation	Concatenate document D_1 with D_2 K times,	$cD(D_1, D_2, K)$

Table XIII. Relevance-preserving perturbations

Name	Semantic	Perturbation
relevance addition	Add a query term to a a relevant document	$aT(t, D, K)$, where $D \in \mathcal{D}_r, t \in Q$
noise addition	Add a noisy term to a document	$aT(t, D, K)$, where $D \in \mathcal{D}, t \in \mathcal{V}_n$
internal term growth	Add a term to a document that originally contains the term	$aT(t, D, K)$, where $D \in \mathcal{D}, t \in D$
document scaling	Concatenate D with itself K times	$cD(D, D, K)$, where $D \in \mathcal{D}$
relevant doc. concatenation	Concatenate two relevant documents K times	$cD(D_1, D_2, K)$, where $D_1, D_2 \in \mathcal{D}_r$
non-relevant doc. concatenation	Concatenate two non-relevant documents K times	$cD(D_1, D_2, K)$, where $D_1, D_2 \in \mathcal{D}_n$
noise deletion	Delete a term from a non-relevant document	$dT(t, D, K)$, where $D \in \mathcal{D}_n, t \in D$
document addition	Add a document to the collection	$aD(D, K)$
document deletion	Delete a document from the collection	$dT(D)$, where $D \in \mathcal{D}$.

Since it is time consuming to recreate relevance judgments, we want to preserve the relevance status of every document after any perturbation. Following the definition of relevance used in TREC, we assume that any relevance evidence in a document makes the document relevant.

Note that not all the proposed basic operators are guaranteed to maintain the relevance status of a document. For example, delete query terms from a relevant document could change the document to non-relevant. Thus, what we need is *relevance-preserving* perturbations. A relevance-preserving perturbation is a perturbation where we have high confidence that the relevance status of each document after the perturbation remains the same as that of the corresponding original document.

We now define several *relevance-preserving* perturbations based on the proposed basic operators (summarized in Table XIII). All these perturbations are intuitive. For example, a relevant document remains relevant if we add more query terms. Also, under the assumption that a document is relevant as long as part of it is relevant, adding noisy terms to any document would not change its relevance status. Furthermore, the relevance status of a document remains the same if we concatenate it with itself several times. Similarly, concatenating two relevant documents or two non-relevant documents would not affect the relevance status either. Note that in document concatenation, the changes of term occurrences are proportional to the

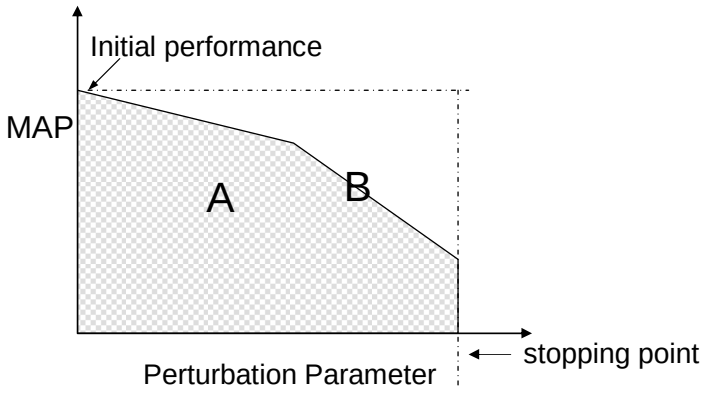

Fig. 6. Example curve for the results of a diagnostic test

document length.

4.2 Diagnostic Tests for IR Models

We now discuss how to use the proposed relevance-preserving perturbations to design diagnostic tests for retrieval models.

4.2.1 *Common Procedure.* In general, in order to design diagnostic tests, we would first identify the aspects of retrieval functions that we want to test, i.e., some specific desirable properties that we believe a reasonable retrieval function should have. This reasoning is similar to the definition of the formalized retrieval constraints given in the previous sections. However, instead of using those binary constraints to compare retrieval functions analytically, here we design diagnostic tests to examine these properties experimentally using test collections.

After we identify the desirable properties to be diagnosed, we need to further connect these properties with our relevance-preserving perturbations and select appropriate perturbation operators. Once the perturbations are chosen for a particular property, we could use the perturbation parameter of the operators to control the degree of perturbation. As we gradually increase the degree of perturbation, we would record the empirical performance of retrieval functions for each perturbation parameter value on the corresponding perturbed collections, and stop perturbing when we get enough information (e.g., when we can observe clear performance differences among retrieval functions). This procedure allows us to obtain a performance curve like the one shown in Figure 6; it gives us a picture of how the performance changes as we impose more dramatic perturbations. In such a figure, the x-axis is always the value of perturbation parameter, and y-axis is a standard retrieval performance measure, which is MAP in our experiments.

Note that the perturbation parameter K could be set in many different ways. Here we only consider two possibilities: (1) *constant growth*, where K is the same for all terms and documents; (2)*linear growth*, where K is proportional to some term statistics, such as $c(t, D)$, or document statistics, such as $|D|$. It is often hard to pre-define the range of perturbations. In this work, we increase K and stop when we observe clear performance differences among retrieval functions or when K

reaches a sufficiently large value. We leave the problem of finding a more principled way to set the parameter range as a future work. There are also different ways to select documents for perturbation: (1)*general collection perturbation*, where all the documents in the collection would be perturbed; (2)*sub-collection perturbation*, where only a subset of collection is perturbed. Note that we need to make sure that this choice be consistent with the requirements of the relevance-preserving perturbations.

Intuitively, the perturbation results can be very useful to understand the behavior of a retrieval function. For example, a flat curve would mean that the function being tested is completely insensitive to such perturbation, while a dropping curve would mean that the function suffered from the perturbation. To interpret the results, we need to design measures to quantitatively evaluate perturbation results.

In general, measures can be defined based on the area under the curve or some extreme performance values (e.g., initial and end values or maximum and minimum values). Naturally, which measure is the best would often depend on the property to be tested. In our study, we are most interested in how the performance degrades or increases as we increase the amount of perturbation. For this purpose, we define the following *performance ratio* (PR) measure:

$$PerformanceRatio = \frac{area\_under\_curve}{area\_under\_line\_through\_init\_point}$$

The PR value of the curve shown in Figure 6 can be computed by dividing the area of shaded part A by the area of rectangle B. Intuitively, the PR value tells us the average amount of degradation or gain in performance after a perturbation. A high PR value indicates more gain in performance while a low PR value indicates more loss. The desirable PR value would depend on the specific perturbation, though in most of our experiments, a high PR value is better and suggests a more robust retrieval function. Note that the PR value can be larger than 1, which means that the retrieval performance increases as we increase the amount of perturbation.

We now present three groups of diagnostic tests designed by following this common procedure.

4.2.2 *Length Variation Sensitivity Tests.* Document length normalization is an important component in virtually all effective retrieval functions. To help understand a function's length normalization mechanism, we design tests to examine the sensitivity of a retrieval function to the length variance in the document collection. We use document scaling perturbation, *i.e.*, $cD(D, D, K)$, to perform the tests, because it changes the document length of each document, yet maintains the relative ratio of term occurrences. We design the following three different tests to examine the different aspects of the length variation. These three tests differ in how to set the value of perturbation parameter i.e., K.

Length variance reduction test (LV1): This test is to use the document scaling perturbation to make all the documents have similar or identical length, and it would thus reduce the variance of document lengths. The test is defined as follows.

For every $D \in \mathcal{D}$, perform $cD(D, D, K)$

with $K = \frac{(1-\beta)\times|d|+\beta\times 1000000}{|d|}$ and $0 \leq \beta \leq 1$.

β is a parameter to control the degree of perturbation. When β is set to 0, all the documents have the original length. When β is set to 1, all the documents have the same length (i.e., number of terms in the documents is $1,000,000$).

Since more perturbation would deprive the retrieval function of any benefit of length normalization, the test result can indicate how effective the length normalization mechanism of a function is. A lower PR value indicates that the function could gain more through length normalization.

Length variance amplification test (LV2): This test is to amplify the differences in document lengths and make distribution of document lengths skewer. The test is defined as follows:

For every $D \in \mathcal{D}$, perform $cD(D, D, K)$,

where $K = |D| \times \beta$.

K is proportional to the original document length, which means that longer documents will grow much more rapidly compared with shorter ones. β is used to control the degree of perturbation. A larger β leads to skewer document length distribution. We expect a robust function to have a high PR value.

Length scaling test (LV3): This test is to concatenate all the documents with themselves K times, where K is same for all the documents. In this way, the length variance would change but the relative length ratio remains the same.

The test has the same intuition as the LNC2 constraint proposed in previous sections. Thus, if a retrieval function achieves a higher PR value, it means that the function does a better job to avoid over-penalizing long documents.

4.2.3 *Term Noise Resistance Tests.* A robust retrieval function should also be resistant to term noise, i.e., the terms that do not contribute to the relevance of a document ($\forall t_n \in \mathcal{V}_n$). We assume that a document is relevant if it contains some relevant evidence, so a reasonable retrieval function is expected to be resistant to the addition of term noise. We design the following test to examine the term noise resistance of a retrieval function.

Noise addition test (TN): This test is to add noise (i.e., non-relevant terms) to documents. We use the *noise addition perturbation* operator as follows:

For every $D \in \mathcal{D}$, perform $aT(t_n, D, K)$

where $t_n \in \mathcal{V}_n$ and K is a parameter.

There are two variations: (1) constant growth: K is a constant, i.e., we add the same number of noisy terms to all documents; and (2) linear growth: $K = \beta \times |D|$, $\beta > 1$, i.e., the length of a perturbed document is linear to the original document length. The test has the same intuition as the LNC1 constraint, because both of them consider the situations when more non-query terms are added to documents. Thus, a high PR value indicates that the function is reasonable in penalizing long documents.

An alternative way to examine noise resistance would be to design a test to remove noise from documents. Unfortunately, this is not easy since non-query terms may

still contribute to the relevance of a document, and removing many non-query terms from relevant documents may cause a relevant document to be non-relevant. We leave further exploration of this option as part of our future work.

4.2.4 *TF-LN Balance Tests.* TF (term frequency) and LN (length normalization) are two important heuristics that are often coupled together in retrieval models due to the need for TF normalization. We design three tests to examine the ability of a retrieval function to balance TF and LN. The main idea is to increase the occurrences of the query terms that are already in the documents. This has a mixed effect on the score of a document: the score may be increased due to the increase of TF values, but it may also be decreased due to the increased length of the document. Thus our tests would reveal the overall influence of such perturbation on the retrieval performance. The following three tests differ in how to pick the query terms whose occurrences are to be increased.

Single query term growth test (TG1): We increase the term occurrence for only one random query term, and use the internal term growth perturbation as follows:

t is a random query term, for every $D \in \mathcal{D}$

if $t \in D$, perform $aT(t, D, K)$.

This test is designed to increase term occurrence of one query term so that a query term will *dominate* in a document. A retrieval function with a higher PR value for this test is more robust against such dominance and favors documents containing more distinct query terms.

Majority query term growth test (TG2): We can increase the term occurrences for all but one random query term. This test can be defined as follows:

t is a random query term, for every $D \in \mathcal{D}$

for every $t' \in Q - \{t\}$, if $t' \in D$, perform $aT(t', D, K)$.

The test is designed to increase term occurrences for majority query terms so that only one query term will be less dominant in a document. Obviously this test is only meaningful for queries with at least two terms, and in the case when there are exactly two terms, it is the same as the previous test. A higher PR value indicates that the function is more robust against such majority dominance and favors documents containing more of the query terms (i.e., larger sum of all query term occurrences).

All query term growth test (TG3): We perform the internal term growth perturbation for all query terms :

For every $D \in \mathcal{D}$

for every $t \in Q$, if $t \in D$, perform $aT(t, D, K)$.

This test is to examine whether the increase of TF can compensate for the score loss caused by the length penalization. A retrieval function with higher PR value can balance the TF and LN parts better.

Table XIV. Tests and corresponding interpretations

Test	**Interpretation of higher value in the results**	**Label**
length variance reduction	have less gain on length normalization.	LV1
length variance amplification	be more robust to larger document variance.	LV2
length scaling	better at avoiding over-penalizing long documents.	LV3
term noise addition	penalize long documents more appropriately.	TN
single query term growth	favor documents with more distinct query terms.	TG1
majority query term growth	favor documents with more query terms.	TG2
all query term growth	balance TF and LN more appropriately.	TG3

Table XV. Optimal performance of representative functions (short keyword queries)

	Trec7	Trec8	Web	AP	DOE	FR
Pivoted	0.176	0.244	0.288	0.227	0.179	0.218
Okapi	0.186	0.251	0.310	0.226	0.185	0.230
Dirichlet	0.186	0.257	0.302	0.224	0.180	0.202
PL2	0.183	0.257	0.314	0.222	0.185	0.216

Table XVI. Length variation robustness test results

Test	Pivoted	Okapi	Dirichlet	PL2	Desirable Value and Interpretation
LV1	0.914	**0.845**	0.883	0.864	Low → better implementation of LN
LV2	**0.829**	0.822	0.811	0.739	High → more robust in a collection with higher length variance
LV3	0.850	**0.927**	0.826	**0.928**	High → better at avoiding over-penalizing long documents

The proposed tests and their interpretations are summarized in Table XIV.

5. DIAGNOSTIC TESTS ON REPRESENTATIVE RETRIEVAL FUNCTIONS

In this section, we perform the proposed diagnostic tests on the same four state-of-the-art retrieval functions defined earlier in Equations (1)-(4), i.e., pivoted normalization, Okapi, Dirichlet prior and PL2. The tests are conducted on the six data sets used in Section 3. With the traditional Cranfield evaluation methodology, the optimal performances of the four functions are similar on these data sets as shown in Table XV. Thus these MAP values alone cannot help us understand the relative strengths and weaknesses of these four functions, making it hard to gain insights to further improve any of them. We will show that the diagnostic tests are able to help us better understand their underlying differences, diagnose their weaknesses, and gain insights about how to improve their retrieval performance.

5.1 Length variation sensitivity tests

Table XVI shows the results of three length variation sensitivity tests. Every value is the average PR (i.e., performance ratio) on the six data sets. For the variance reduction test (i.e., LV1), Pivoted has the highest PR value, which means that it is least sensitive to this test. On the other hand, Okapi has the lowest PR value, which means that it loses the most when we "turned off" its length normalization part, indicating that the length normalization part of Okapi is implemented more reasonably than other functions.

Table XVII. Additional length scaling test results

	Pivoted	Okapi	Dirichlet	PL2	Desirable value and interpretation
LV3-nonrel	**0.11**	0.20	0.89	0.61	Low → better at avoiding over-penalizing long documents
LV3-rel	0.17	**2.09**	1.19	**1.99**	High → balance TF and LN better

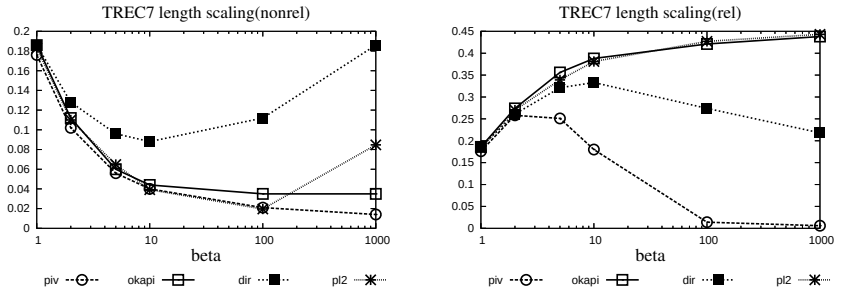

Fig. 7. Additional length scaling tests:LV3-nonrel(Left), LV3-rel(Right)

For the length variance amplification test (i.e., LV2), Pivoted has the highest PR value, which means that it is the most robust one if we increase the length variances in the collection. Thus, it means that Pivoted normalization function might be the best choice if the document lengths varies a lot in the collection.

For the length scaling test (i.e., LV3), both Okapi and PL2 have the highest PR values, indicating that they are the most robust retrieval functions for this test. Since this test has the same intuition as the LNC2 constraint, it can be regarded as a test to examine how well a retrieval function avoids over-penalizing long documents. Thus, the lower PR values of Dirichlet and pivoted indicate that these two functions might not do a good job at avoiding over-penalizing the long documents. Although this result is similar to what we have already obtained through analytical checking of the LNC2 constraint, these tests can be applicable to any retrieval function to perform such diagnostic analysis, whereas for some retrieval functions, we may not easily obtain analytical conclusions regarding whether they satisfy LNC2.

To further verify our results of LV3, we design and perform two additional length scaling tests. Instead of scaling all documents, we conduct two tests where we scale only non-relevant documents (i.e., LV3-nonrel) and only relevant documents (i.e., LV3-rel), respectively. The results are shown in Table XVII and Figure 7. In LV3-nonrel test, all the non-relevant documents become longer. We would expect that a retrieval function that penalizes long documents more harshly to have a higher PR value. Dirichlet has the highest PR value, followed by PL2, indicating that they both tend to penalize long documents more harshly than Okapi or pivoted normalization. In LV3-rel test, both term frequency and document length grow in all relevant documents. We expect that a retrieval function that balances TF and LN well would get a higher PR value. The lowest PR value of pivoted indicates that it does not balance the growth of TF and LN as well as the other functions, whereas Okapi and PL2 seem to be the best in balancing TF and LN.

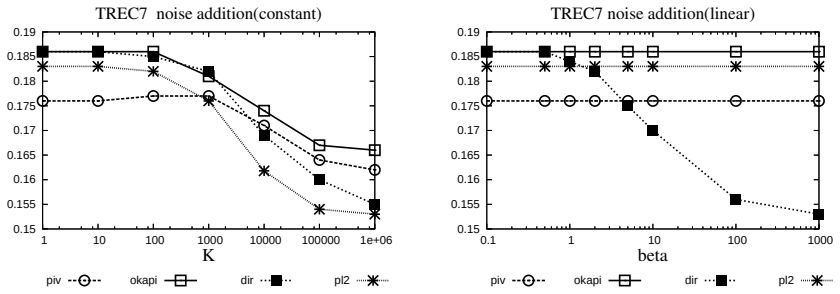

Fig. 8. Term noise addition tests (TN)

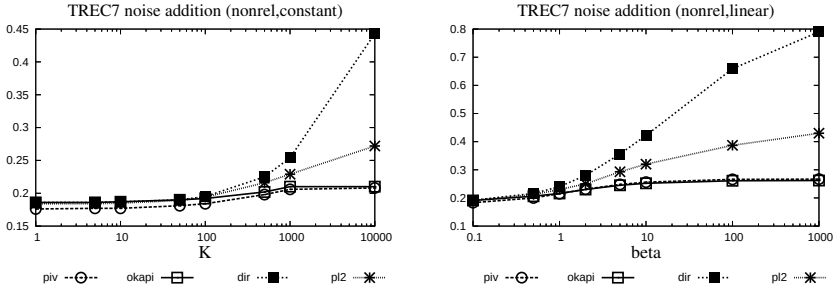

Fig. 9. Additional term noise tests (TN-nonrel)

5.2 Term noise resistance tests

Table XVIII and Figure 8 show the results of term noise resistance tests where the noisy terms are added to all the documents. The lowest PR value of Dirichlet indicates that Dirichlet does not penalize long documents as appropriately as others.

To further understand the results, we design one additional test (i.e., TN-nonrel). Instead of performing the test on all the documents, we perform it only on non-relevant documents. Thus, when we do more perturbation, the length of a non-relevant document would become longer, and we expect that a retrieval function penalizing long documents more harshly would perform much better when we do more perturbation. Figure 9 shows the results for TREC7. The curve for other data sets are similar. The performance of Dirichlet grows more quickly, indicating that it penalizes the long documents more harshly, which is consistent with our findings in non-relevant document length scaling test (i.e., LV3-nonrel).

5.3 TF-LN balance tests

The results of TF-LN balance tests are summarized in Table XIX. It is clear that PL2 has the highest scores for most of the tests, indicating that in general, PL2 can balance TF and LN better than the other three functions.

However, there appears to be no clear pattern among pivoted normalization, Okapi, and Dirichlet prior. After looking into the trends of performance (as we make more perturbations) for these tests, we found that Dirichlet prior behaves

Table XVIII. Noise resistant test results

	Pivoted	Okapi	Dirichlet	PL2	Desirable value and interpretation
TN (constant)	**0.940**	0.932	0.896	0.882	High → penalize long documents better
TN (linear)	**1**	**1**	0.826	**1**	High → penalize long documents better
TN-nonrel (constant)	**1.19**	1.26	2.94	1.79	Low → better at avoiding over-penalizing long documents
TN-nonrel (linear)	**1.49**	1.66	4.03	2.67	Low → better at avoiding over-penalizing long documents

differently from the other three functions. In Figure 10, we show the trend curves on TREC 7 for the single term growth and majority term growth tests. Here we see that while in general the performances of all the methods drop as we increase K (i.e., the amount of perturbation), the performance of Dirichlet prior drops much more quickly after K becomes larger than $1,000$.

There are two possible causes of this sudden drop: either some relevant documents ended up being penalized or some non-relevant documents ended up being rewarded as a result of the perturbation. Based on the analysis from previous length-related tests, we know that Dirichlet tends to penalize long documents more harshly, thus one possible explanation for this sudden drop is that when K is very large, the potential increase in the score of a document due to the increased occurrences of query terms cannot compensate for the decrease caused by the increase of document length. Since relevant documents tend to have more query terms, they get penalized more than non-relevant documents, leading to the quick degradation of performance.

To further look into this hypothesis, we perform another set of tests, i.e., after performing query term growth tests, we perform length variance reduction test again to make all the document lengths equal. In this way, we hope to factor out the effect of length normalization. The results are shown in Figure 11. We see that Dirichlet prior still has a quick drop for the single term growth test, which means that the drop was probably not caused by penalizing relevant documents (due to length normalization). Thus it is more likely that the reason was because some non-relevant documents were rewarded because of excessive occurrences of a single query term. This problem appears to be less serious in the case of majority term growth, which may be because when more query terms are repeated simultaneously in the documents, relevant documents will likely benefit more than non-relevant documents. Thus Dirichlet prior appears to differ from other methods in that it tends to rely more on the *overall* counts of terms, including repeated occurrences of the same term, when scoring a document, while others may put more emphasis on the matching of more *distinct* query terms. This distinct characteristic of Dirichlet prior is likely related to the independence assumption about multiple occurrences of the same term made in the Dirichlet prior method (which means that we would treat multiple occurrences of the same term as independent evidences).

For the all query term growth test, the results for constant growth and linear growth are not conclusive as shown in Figure 12 and Table XIX. In linear growth test, Pivoted has the smallest PR value, which means that it can not balance TF

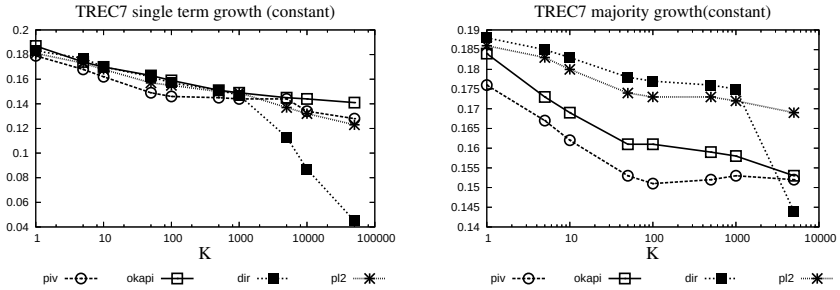

Fig. 10. Single and majority term growth tests

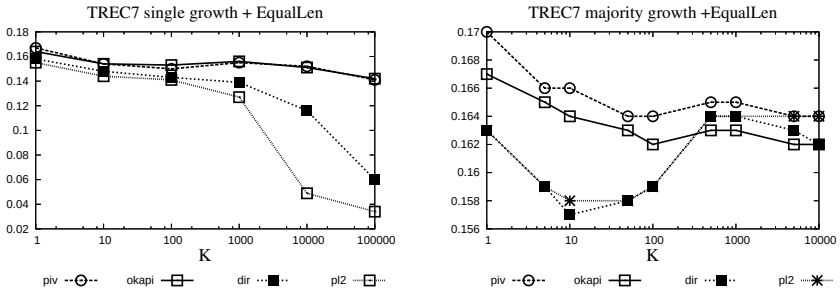

Fig. 11. Single and majority term growth + Equal Len

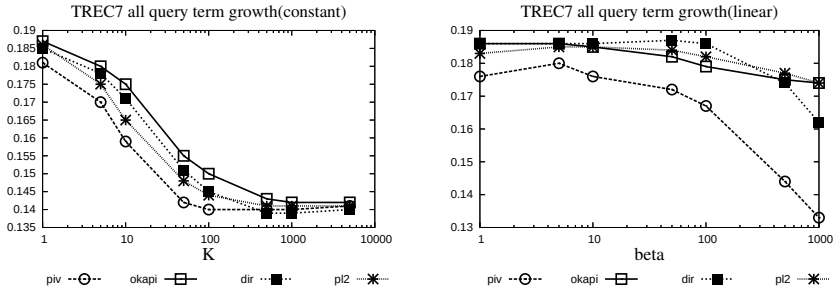

Fig. 12. All query term growth tests

and LN very well in these cases, because the increase of TF in Pivoted can not compensate for the penalty caused by the document length. This observation is also consistent with our analysis of the relevant document length scaling test.

5.4 Summary of diagnostic evaluation results

All the results presented in the previous subsection clearly demonstrate that the proposed diagnostic tests can help pinpoint the weaknesses and strengths of the four functions. Here we briefly summarize our findings in Table XXI. Instead of presenting results measured by PR, we give a confidence score to every pair-wise

Table XIX. TF-LN balance test results (Larger value is better)

		Pivoted	Okapi	Dirichlet	PL2
single	constant	0.840	0.854	0.85	**0.864**
(TG1)	linear	0.827	0.881	0.875	**0.932**
majority	constant	0.802	0.798	0.823	**0.838**
(TG2)	linear	0.775	0.773	0.817	**0.914**
all	constant	**0.730**	0.721	0.694	0.703
(TG3)	linear	0.793	0.897	0.870	**0.932**

Table XX. Additional TF-LN balance test results (perturb non-relevant docs only)

		Pivoted	Okapi	Dirichlet	PL2
single	constant	0.455	0.680	1.29	1.21
(TG1)	linear	0.383	0.624	1.41	1.19
majority	constant	0.417	0.637	1.58	1.36
(TG2)	linear	0.292	0.517	1.93	1.34
all	constant	0.222	0.425	1.93	0.839
(TG3)	linear	0.113	0.284	2.39	0.967

comparison. The confidence score is computed by the percentage of the data sets supporting the conclusion. For example, if 5 out of 6 data sets show that A performs better than B for test T, we have 83.3% confidence to say that A performs better than B for test T.

Comparing the findings from constraint analysis (as shown in Table VI) with those from diagnostic tests (as shown in Table XXI), we observe that many findings from these two strategies are consistent. First, constraint analysis shows that Okapi is the only retrieval function that satisfies both LN constraints, which is consistent with the results of LV1 test, i.e., the implementation of LN in Okapi is better. Second, Dirichlet is diagnosed to over-penalize long documents based on both constraint analysis (i.e., LNC2) and diagnostic tests (i.e., LV3, LV3-nonrel, TN and TN-nonrel). Finally, Pivoted does not balance the TF and LN well based on TF-LNC, LNC2, LV3-rel test, and TG3 test.

Although these findings from both constraint analysis and diagnostic tests are similar, the diagnostic tests are complementary with constraint analysis. Depending on the form of a retrieval function, it may be mathematically challenging to draw a clear conclusion from formal constraint analysis, thus constraint analysis alone may not be sufficient; in contrast, the diagnostic tests can be applied to any retrieval function to experimentally analyze how well it satisfies various properties. Moreover, , diagnostic tests can also provide additional information that can not be found using constraint analysis. For example, we could identify the unique advantage of Pivoted, i.e., it performs better when the document collection has larger length variance. Also, constraint analysis was unable to reveal the subtle differences in the implementations of TF in the analyzed retrieval functions, but diagnostic tests can reveal the unique strengths and weaknesses of TF implementation in these retrieval functions. In particular, the diagnostic tests show that PL2 can balance the TF and LN better than the other three functions while the constraints analysis can not show it.

Based on the results from constraint analysis and diagnostic tests, we summa-

Table XXI. Summary of diagnostic results I (>> means more desirable, ? represents uncertainty) P denotes pivoted, O denotes okapi, D denotes Dirichlet and L denotes PL2

Tests		**P vs. D**	**O vs. D**	**O vs. P**	**O vs. L**	**L vs. P**	**L vs. D**
LV1		$D >> P$ (66.7%)	$O >> D$ (83.3%)	$O >> P$ (100%)	$O >> L$ (66.7%)	$L >> P$ (100%)	$L >> D$ (66.7%)
		Okapi has better implementation of LN					
LV2		$D << P$ (83.3%)	$D << O$ (83.3%)	$O << P$ (66.7%)	$O >> L$ (66.7%)	$L?P$ (50%)	$L?D$ (50%)
		Piv. performs better in high-variance collections					
LV3		$P >> D$ (83.3%)	$O >> D$ (100%)	$O >> P$ (66.7%)	$O?L$ (50.0%)	$L >> P$ (83.3%)	$L >> D$ (83.3%)
		Dir.&PL2 over-penalize long documents					
LV3-nonrel		$P >> D$ (100%)	$O >> D$ (100%)	$P >> O$ (100%)	$O >> L$ (83.3%)	$L << P$ (100%)	$L >> D$ (83.3%)
		Dir.&PL2 over-penalize long documents					
LV3-rel		$D >> P$ (100%)	$O >> D$ (83.3%)	$O >> P$ (100%)	$O?L$ (50%)	$L >> P$ (100%)	$L >> D$ (100%)
		Piv. does not balance TF and LN well					
TN	constant	$P >> D$ (83.3%)	$O >> D$ (66.7%)	$P >> O$ (66.7%)	$O >> L$ (100%)	$L << P$ (100%)	$L << D$ (83.3%)
	linear	$P >> D$ (100%)	$O >> D$ (100%)	$P = O$ (100%)	$O = L$ (100%)	$L = P$ (100%)	$L >> D$ (100%)
	Dir. over-penalizes long documents						
TN -nonrel	constant	$P >> D$ (83.3%)	$O >> D$ (100%)	$P >> O$ (66.7%)	$O >> L$ (100%)	$L << P$ (100%)	$L >> D$ (100%)
	linear	$P >> D$ (100%)	$O >> D$ (100%)	$P >> O$ (66.7%)	$O >> L$ (100%)	$L << P$ (100%)	$L >> D$ (100%)
	Dir. over-penalizes long documents						
TG1	constant	$P >> D$ (100%)	$O >> D$ (100%)	$O >> P$ (66.7%)	$O << L$ (83.3%)	$L >> P$ (83.3%)	$L >> D$ (66.7%)
	linear	$P >> D$ (66.7%)	$O >> D$ (83.3%)	$O >> P$ (66.7%)	$O << L$ (66.7%)	$L >> P$ (100%)	$L >> D$ (83.3%)
	Okapi favors documents with more distinct query terms; PL2 balances TF-LN well						
TG2	constant	$D >> P$ (66.7%)	$D >> O$ (83.3%)	$P >> O$ (66.7%)	$O << L$ (100%)	$L >> P$ (83.3%)	$L?D$ (50%)
	linear	$D >> P$ (100%)	$D >> O$ (100%)	$P >> O$ (66.7%)	$O << L$ (100%)	$L >> P$ (100%)	$L >> D$ (100%)
	Dir. favors documents with more query terms; PL2 balances TF-LN well						
TG3	constant	$D?P$ (50%)	$O >> D$ (66.7%)	$O >> P$ (66.7%)	$O >> L$ (83.3%)	$L << P$ (83.3%)	$L >> D$ (83.3%)
	linear	$D >> P$ (100%)	$D >> O$ (66.7%)	$O >> P$ (66.7%)	$O << L$ (66.7%)	$L >> P$ (100%)	$L >> D$ (100%)
	Piv. does not balance TF and LN well; PL2 balances TF-LN well						

rize the weaknesses of every function in Table XXII. In the next section, we will show how we can leverage these findings to improve the state-of-the-art retrieval functions.

6. IMPROVING RETRIEVAL FUNCTIONS

We now present several ways to modify existing retrieval functions based on the results of diagnostic tests, and compare the performance (i.e., MAP) of the modified functions with the existing retrieval functions. In addition to evaluating the

Table XXII. Weaknesses of functions and supporting constraints/tests

	Weaknesses	Constraints	Diagnostic tests
Piv.	can not balance TF and LN well	TF-LNC	LV3 (rel), TG3
	penalizes long documents more harshly	LNC2	LV3
	does not favor documents with more query terms	—	TG2
Ok.	does not favor documents with more query terms	—	TG2
Dir.	penalizes long documents more harshly	LNC2	LV3, LV3-nonrel, TN, TN-nonrel
	under-rewards matching more distinct query terms	—	TG1
PL2	overly-penalizes noise-dominated long documents	—	TN, TN-nonrel

modified retrieval functions on the data sets that we used to diagnose the original retrieval functions (which can indicate whether our diagnostic evaluation method can indeed provide useful guidance for improving the performance of a retrieval function on the same data sets), we also evaluate the modified functions on the following two additional standard TREC collections to see whether our improvement can be generalized to other collections.

—**Robust04**: TREC disk45 (minus congressional record) with 249 official topics of Robust track in 2004.

—**Robust05**: AQUAINT data with 50 official topics used in Robust track 2005.

In all the result tables, ‡ and † indicate that the improvement is statistically significant according to Wilcoxon signed rank test at the level of 0.05 and 0.1 respectively.

6.1 Improving Length Normalization

Both constraint analysis and diagnostic tests indicate that Pivoted and Dirichlet suffer from the problem of penalizing long documents too harshly. Thus one way to improve them is to modify their length normalization components heuristically as follows:

$$MPln: S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times \frac{TF_{Piv}(t,D) \times IDF_{Piv}(t)}{LN_{Piv}(D)^{\lambda}}$$

$$MDln: S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times TFIDF_{Dir}(t,D) - |Q| \times LN_{Dir}(D)^{\lambda}$$

where $0 < \lambda \leq 1$ and

$$IDF_{Piv}(t) = ln(\frac{N+1}{df(t)})$$

$$TF_{Piv}(t,D) = 1 + ln(1 + ln(c(t,D)))$$

$$LN_{Piv}(D) = 1 - s + s\frac{|D|}{avdl}$$

$$TFIDF_{Dir}(t,D) = ln(1 + \frac{c(t,D)}{\mu \times p(t|C)})$$

$$LN_{Dir}(D) = ln(1 + \frac{|D|}{\mu})$$

Table XXIII. Improvement of LN modifications

	Trec7	**Trec8**	**Web**	**AP**	**DOE**	**FR**	**Robust04**	**Robust05**
Piv.	0.176	0.244	0.288	0.227	0.179	0.218	0.241	0.200
MPln	0.181	0.250‡	0.306‡	0.227†	0.180†	0.231	0.245‡	0.201†
Dir.	0.186	0.257	0.302	0.224	0.180	0.202	0.251	0.196
MDln	0.187	0.260†	0.318‡	0.225	0.182	0.205†	0.251	0.196

Table XXIV. LV test results for modified functions (over six data sets)

Tests	LV1	LV2	LV3
Desirable value	Low	High	High
Pivoted	0.914	0.829	0.850
MPln	**0.888**	**0.839**	**0.930**
Dirichlet	0.883	0.811	0.826
MDln	**0.869**	**0.816**	**0.906**

A comparison of the upper-bound performance (i.e., optimized performance) of the modified functions and the original functions is shown in Table XXIII. It shows that such length normalization modification indeed improves the performance in both cases. We also perform the diagnostic tests for these modified functions over the same six data sets that we used for such analysis earlier. Table XXIV shows that the PR values of these modified functions are now more desirable (i.e., lower for LV1 and higher for other tests) than the original ones in all the LV tests, which indicates that the modified functions have better implementation of length normalization part.

6.2 Improving TF Implementations

The diagnostic results show that the TF implementation of Dirichlet and that of Okapi/Pivoted represent two extreme cases: one is to favor documents with more query terms (i.e., larger sum of all query term occurrences), one is to favor documents with more distinct query terms. An ideal TF can be hypothesized to lie in somewhere between the two extremes. Based on this intuition, we heuristically modify the pivoted and Dirichlet as follows.

$$MPtf1 : S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times \frac{tfidf1(t,D)}{LN_{Piv}(D)}$$

$$MPtf2 : S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times \frac{tfidf2(t,D)}{LN_{Piv}(D)}$$

$$MDtf1 : S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times tfidf1(t,D) - |Q| \times LN_{Dir}(D)$$

$$MDtf2 : S(Q,D) = \sum_{t \in Q \cap D} c(t,Q) \times tfidf2(t,D) - |Q| \times LN_{Dir}(D)$$

where

$$tfidf1(t,D) = \alpha \times TF_{Piv}(t,D) \times IDF_{Piv}(t) + (1-\alpha) \times TFIDF_{Dir}(t,D)$$

$$tfidf2(t,D) = \alpha \times TF_{Ok}(t,D) \times IDF_{Piv}(t) + (1-\alpha) \times TFIDF_{Dir}(t,D)$$

Table XXV. Improvement of TF modifications

	Trec7	**Trec8**	**Web**	**AP**	**DOE**	**FR**	**Robust04**	**Robust05**
Piv.	0.176	0.244	0.288	0.227	0.179	0.218	0.241	0.200
MPtf1	0.182	0.248	0.293	0.227	0.185	0.222	0.245‡	0.201‡
MPtf2	0.182	0.250†	0.297†	0.227	0.187‡	0.216	0.246‡	0.200
Dir.	0.186	0.257	0.302	0.224	0.180	0.202	0.251	0.196
MDtf1	0.190†	0.261‡	0.313‡	0.229‡	0.183	0.227‡	0.254‡	0.204‡
MDtf2	0.188‡	0.257	0.313‡	0.229‡	0.185‡	0.218‡	0.252†	0.203†

Table XXVI. TG (constant) test results for modified functions (over six data sets)

Tests	TG1	TG2	TG3
Desirable value	High	High	High
Pivoted	0.840	0.802	0.730
MPtf1	**0.861**	**0.836**	0.723
MPtf2	**0.866**	**0.844**	0.721
Dirichlet	0.85	0.823	0.694
MDtf1	**0.857**	**0.828**	**0.697**
MDtf2	**0.862**	**0.835**	0.690

$$TF_{Ok}(t,d) \;=\; \frac{2.2 \times c(t,D)}{1.2 + c(t,D)}$$

and $\mu = 2000$ in $TFIDF_{Dir}(t,d)$, $0 \leq \alpha \leq 1$ and other notations are the same as explained at the beginning of the section.

The optimal performance of the modified retrieval functions and that of the original retrieval functions are compared in Table XXV. The results show that the modification can indeed improve the performance. In addition, Table XXVI shows that the PR values of these modified functions are higher than the original ones in TG1 and TG2 tests, which means that the modification corresponds to better implementation of the TF component.

6.3 Additivity of modified TF and LN implementations

Since both LN and TF modifications are effective, we can combine them in the following way.

$$MPtf2ln : S(Q,D) \;=\; \sum_{t \in Q \cap D} c(t,Q) \times \frac{tfidf2(t,D)}{LN_{Piv}(D)^{\lambda}}$$

$$MDtf2ln : S(Q,D) \;=\; \sum_{t \in Q \cap D} c(t,Q) \times tfidf2(t,D) - |Q| \times LN_{Dir}(D)^{\lambda}$$

The performance is reported in Table XXVII. Indeed, the combined modifications perform better than not only the original retrieval functions but also the individual modifications in almost all the cases based on comparisons of their optimal performances. As a reference, we also include the optimal performance of Okapi and PL2 in the last two rows of the table; we see that both of our new functions (**MPtf2ln** and **MDtf2ln**) also perform better than Okapi and PL2 in most cases. The additivity of performance improvement of LN and TF presumably comes from

Table XXVII. Additivity of TF and LN modifications

	Trec7	**Trec8**	**Web**	**AP**	**DOE**	**FR**	**Robust04**	**Robust05**
Piv.	0.176	0.244	0.288	0.227	0.179	0.218	0.241	0.200
MPln	0.181	0.250	0.306	0.227	0.180	0.231	0.245	0.201
MPtf2	0.182	0.250	0.297	0.227	0.187	0.216	0.246	0.200
MPtf2ln	**0.186**	**0.257**	**0.314**	**0.228**	**0.187**	**0.236**	**0.251**	**0.204**
Dir.	0.186	0.257	0.302	0.224	0.180	0.202	0.251	0.196
MDln	0.187	0.260	0.318	0.225	0.182	0.205	0.251	0.196
MDtf2	0.188	0.257	0.313	0.229	0.185	0.218	0.252	0.203
MDtf2ln	**0.190**	**0.263**	**0.322**	**0.230**	**0.185**	**0.228**	**0.255**	**0.203**
Okapi	0.186	0.251	0.310	0.226	0.185	0.225	0.248	0.201
PL2	0.183	0.257	0.314	0.212	0.188	0.216	0.252	0.196

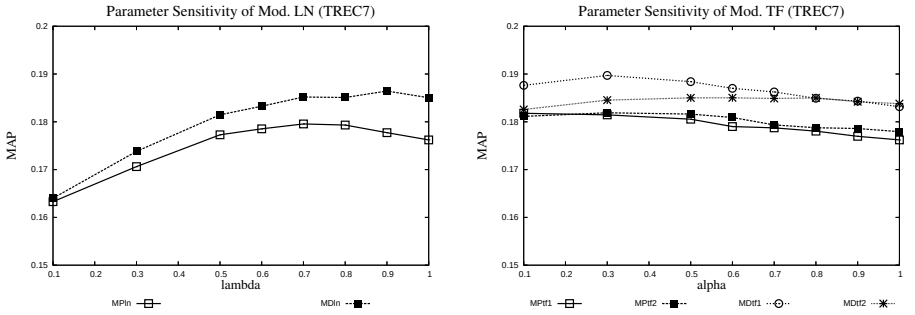

Fig. 13. Parameter Sensitivity: λ(Left), α(Right)

the fact that they capture different weaknesses in a retrieval function (i.e., length normalization and TF implementation), which once again confirms the usefulness of the diagnostic tests for obtaining insights to improve a retrieval function.

6.4 Further evaluation of **MPtf2ln** and **MDtf2ln**

The results in Table XXVII show that the two derived new functions, **MPtf2ln** and **MDtf2ln**, perform better than all the existing retrieval functions not only on the data sets used to derive them but also on the two new data sets (i.e., Robust04 and Robust05). However, these results are based on the optimal performances of all the involved retrieval functions. Since the two new functions both have two additional parameters, i.e., α and λ, we now look into the sensitivity of performance to these parameters. In both functions, α controls the degree of TF modification, while λ controls the degree of LN modification.

To examine the sensitivity to each parameter, we plot their sensitivity curves computed on the TREC7 data set in Figure 13. Since α only affects the TF implementation, we plot its sensitivity curves using functions **MPtf1**, **MPtf2**, **MDtf1**, and **MDtf2**; similarly, λ only affects length normalization, thus we plot its sensitivity curves using functions **MPln** and **MDln**.

From these curves, we see that the performance is less sensitive to parameter α than parameter λ, which may be because our modification of TF is more conservative than the modification of length normalization. But it clearly shows that the

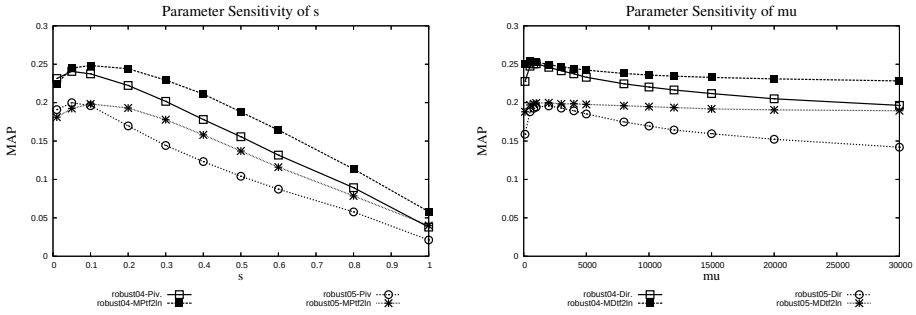

Fig. 14. Comparison of new functions with their corresponding original functions (left: **MPtf2ln**; right:**MDtf2ln**).

optimality of the length normalization component in a retrieval function can affect the retrieval performance significantly. It appears that the optimal value of α is around 0.3 while that of λ is around 0.7, and if they are set non-optimally, the modifications may not improve performance.

To compare our modified functions with the original functions more fairly, we set $\alpha = 0.3$ and $\lambda = 0.7$ in both **MPtf2ln** and **MDtf2ln**. With these two parameters fixed, these two functions now both have precisely one parameter just like their corresponding original functions (i.e., s for **MPtf2ln** and μ for **MDtf2ln**). We tune this single parameter for both the original functions and the new functions and compare their sensitivity curves on Robust04 and Robust05 in Figure 14. Since these two functions have been developed based on the other six data sets, comparing them with their original retrieval functions over these two new data sets would indicate well whether the new functions are generally more effective than the original functions. From this figure, it is clear that both new functions improve over their corresponding original existing functions for essentially the entire range of the parameter values.

We further show a detailed comparison of these two new functions with their corresponding original functions on all the eight data sets in Table XXVIII. We set $\alpha = 0.3$ and $\lambda = 0.7$ for both **MPtf2ln** and **MDtf2ln**, and set $s = 0.2$ and $\mu = 2000$ for all the functions, which are the default values suggested in the previous studies[Singhal 2001; Zhai and Lafferty 2001b].

Table XXVIII. Performance with fixed parameter values

	Trec7	Trec8	Web	AP	DOE	FR	Robust04	Robust05
Piv.	0.163	0.225	0.200	0.219	0.179	0.168	0.222	0.170
MPtf2ln	0.178‡ (+9%)	0.245‡ (+8%)	0.255‡ (+28%)	0.225† (+3%)	0.186 (+4%)	0.210‡ (+25%)	0.244‡ (+10%)	0.193‡ (+14%)
Dir.	0.186	0.251	0.301	0.224	0.180	0.189	0.246	0.196
MDtf2ln	0.187 (+0.5%)	0.258‡ (+3%)	0.320‡ (+6%)	0.229‡ (+2%)	0.183‡ (+2%)	0.214‡ (+13%)	0.250‡ (+2%)	0.200 (+2%)

The results show that both **MPtf2ln** and **MDtf2ln** consistently improve over

their original functions on all the data sets. The improvement of **MPtf2ln** over the pivoted normalization baseline is generally greater than that of **MDtf2ln** over the Dirichlet prior baseline, which is consistent with the upper-bound performance comparison shown in Table XXVII. Also, for both functions, the improvement on the two data sets known to have high variances in document lengths (i.e., Web and FR) appears to be more significant than on other data sets. This is likely because on those two data sets, the gain from improved document length normalization is more significant.

Thus, all these results show that **MPtf2ln** and **MDtf2ln** (with $\alpha = 0.3$ and $\lambda = 0.7$) can be recommended as improved versions of the original pivoted length normalization and Dirichlet prior retrieval functions.

7. DISCUSSION

A fundamental assumption made in our overall diagnostic methodology is that a good understanding of the weaknesses of a retrieval function is required in order to improve the function, and it is possible to discover and characterize the weaknesses of a retrieval function through examining how well it satisfies some desirable properties that we would expect a good retrieval function to satisfy. In this paper, we proposed two synergistic ways to achieve this.

First, we can formally define constraints that we want a retrieval function to satisfy based on major retrieval heuristics such as TF-IDF weighting and document length normalization, and check *analytically* whether a retrieval function satisfies each constraint. These constraints are meant to capture general "universal" properties that we want every reasonable retrieval function to satisfy, thus they are defined *independently* of specific relevance judgments for a query. A significant advantage of defining the constraints independently of relevance judgments is that we can study and compare retrieval functions analytically without needing experimentation. However, because of their generality, the defined constraints tend to be loose and far from sufficient. Thus, while violation of a constraint implies weakness of a function and poor empirical performance, satisfying all the constraints does not guarantee good performance.

In the second way, we design diagnostic tests to *experimentally* check whether a retrieval function exhibits desirable empirical behaviors on test collections. Specifically, we will perturb a test collection to artificially vary certain statistical characteristics of documents and queries and check how a retrieval function responds to such perturbation. While both aiming at revealing specific behavior of a retrieval function, the second way complements the first way in that (1) the diagnostic tests can reveal differences between retrieval functions even when they satisfy the same set of constraints, and (2) the diagnostic tests can be applied to any retrieval function while analytical analysis of constraints may be mathematically challenging for some retrieval functions. The diagnostic tests can also help researchers discover new constraints to enhance constraint analysis, while constraint analysis can provide guidance on what diagnostic tests may be useful.

Our study has demonstrated the usefulness of both ways of analyzing a retrieval function. Some of the state-of-the-art retrieval functions that we analyzed have been proposed for a long time. The fact that they still represent the state of the

art today suggests the difficulty in further improving them [2]. Thus the consistent improvement of our modified retrieval functions over the original functions on all the data sets is quite encouraging. Since our modification is *directly* guided by the results from the diagnostic tests, this indicates that the proposed diagnostic evaluation methodology can be used to effectively diagnose weaknesses and strengths of retrieval functions and generate insights about how to improve them; without such guidance, it would have been difficult to generate a new robust function that can consistently perform better than these state-of-the-art functions.

However, our study also has some limitations. First, although we have shown that the results of diagnostic evaluation of retrieval functions provide direct guidance for improving a retrieval function, our way of modifying the existing retrieval functions is somewhat ad hoc. It would be much more interesting to further explore more principled ways to address the weaknesses of these retrieval functions revealed through diagnostic evaluation. Second, the TF part and LN part in the Okapi, and the TF part and IDF part in the PL2 cannot be separated easily, which makes it harder to address the weaknesses of each component separately. Although the results in Table XXVII show that both **MPtf2ln** and **MDtf2ln** can potentially perform better than Okapi and PL2, it is still unclear how we can directly modify Okapi and PL2 to address its weakness. Another limitation is that the constraints and diagnostic tests defined in this paper are restricted to the bag-of-words representation. Although the proposed general diagnostic evaluation methodology is potentially applicable to analyzing retrieval functions based on other representations, how to further develop additional constraints and tests is, in general, challenging.

8. RELATED WORK

Many different retrieval models have been proposed, including vector space models [Salton et al. 1975; Salton and McGill 1983; Salton 1989], probabilistic models [Robertson and Sparck Jones 1976; van Rijbergen 1977; Turtle and Croft 1991; Fuhr 1992; Ponte and Croft 1998; Lafferty and Zhai 2003; Amati and Rijsbergen 2002], and logic-based models [van Rijsbergen 1986; Wong and Yao 1995; Fuhr 2001]. Our work provides a general methodology for diagnosing weaknesses and strengths of different retrieval models so as to gain insights about how to improve them. We have shown that the proposed diagnostic evaluation method can allow us to better understand four representative state-of-the-art models (i.e., the pivoted length normalization model [Singhal et al. 1996; Singhal et al. 1998], the Okapi/BM25 retrieval model [Robertson and Walker 1994], the Dirichlet prior language model [Zhai and Lafferty 2001b]), and the PL2 model [Amati and Rijsbergen 2002; He and Ounis 2005], and further derive improved versions of some of them.

Evaluation of an information retrieval system has been extensively studied (see, e.g., [Sparck Jones and Willett 1997; Voorhees 2007]). Some recent studies have focused on how to create a better pool and how to reduce the pool size while maintaining the confidence of the evaluation results [Cormack et al. 1998; Zobel 1998; Soboroff et al. 2001; Sanderson and Joho 2004; Carterette and Allan 2005;

[2]There are improved models (e.g., those based on pseudo feedback) that can outperform these basic models, but those models generally use additional information and computationally more expensive.

Carterette et al. 2006]. These studies mainly aim at establishing an evaluation methodology to evaluate a retrieval system from the perspective of the utility to users. As a result, the evaluation results are not always informative enough to directly explain performance differences among retrieval functions or provide guidance on how to improve the performance of a retrieval function. This is especially true when two retrieval functions perform similarly. The proposed diagnostic evaluation method in this paper extends the traditional evaluation methodology to enable diagnosis of weaknesses of retrieval functions through constraint analysis and diagnostic tests.

Studies of pivoted normalization function [Singhal et al. 1996; Singhal et al. 1998] demonstrate that a retrieval function can be improved if we could pinpoint its weakness. Singhal et. al. [Singhal et al. 1996] observed the deficiency of a particular length normalization method, and proposed pivoted normalization technique to modify the normalization function. They later noticed the poor performance of the logarithmic tf-factor on a TREC collection, found another deficiency of the implementation of term frequency part [Singhal et al. 1998], and modified the TF part accordingly. However, their methods are not general and can not be easily applied to identify weaknesses in other aspects; in contrast, the proposed diagnostic methodology in this paper is more general and can be potentially applied to many retrieval heuristics.

Previous work [Salton and Buckley 1988; Zobel and Moffat 1998] has also attempted to identify an effective retrieval formula through extensive empirical experiments, but the results are generally inconclusive with some formulas performing better under some conditions. Our diagnostic evaluation method offers more detailed and systematic understanding of how well a retrieval function implements various heuristics, directly providing insights about how to improve a retrieval function.

The proposed perturbation-based diagnostic methodology assumes that the retrieval performance is closely related to the robustness of a retrieval function to the noisy data, which is in spirit similar to some previous work on noisy data [Lopresti and Zhou 1996; Singhal et al. 1996]. Similar idea has also been used to predict query performance in [Zhou and Croft 2006]. The RIA workshop [Harman and Buckley 2004] has resulted in some interesting observations about the empirical behavior of retrieval functions through a group effort on manually analyzing performance patterns of different retrieval functions. Wong and co-authors [Wong et al. 2001] proposed an inductive theory for evaluation, but the proposed theory is too abstract to deal with term weighting. On the contrary, our proposed methodology provides an effective way of evaluating different term weighting strategies implemented in retrieval functions.

The formalization of retrieval constraints was initially reported in [Fang et al. 2004]. In this article, we extended and restructured the constraints slightly, and we further propose a supplementary diagnostic evaluation method based on collection perturbation. Retrieval constraints have been leveraged to derive new robust and effective retrieval functions (called the axiomatic approach) through searching in the "neighborhood" of existing retrieval functions for a better function [Fang and Zhai 2005] and accommodating semantic term matching [Fang and Zhai 2006;

Fang 2008]; these related studies can be regarded as successful applications of the proposed diagnostic evaluation methodology.

9. CONCLUSIONS AND FUTURE WORK

Understanding weaknesses of state-of-the-art retrieval functions is essential to further improve them. In this paper, we propose a novel general methodology to *diagnose* weaknesses and strengths of a retrieval function. The basic idea is to check how well a retrieval function implements necessary retrieval heuristics. We propose two complementary ways to achieve this through analytical constraint analysis and experimental diagnostic tests, respectively.

To analytically predict and compare the performance of a retrieval function, we formally define seven basic constraints that any reasonable retrieval function should satisfy, capturing several common heuristics, such as term frequency weighting, term discrimination weighting, and document length normalization. We check these seven constraints on four representative retrieval formulas analytically and derive specific conditions when a constraint is conditionally satisfied. The results of constraint analysis suggested several interesting hypotheses about the expected performance behavior of these retrieval functions. We design experiments to test these hypotheses using different types of queries and different document collections. We find that in many cases the empirical results indeed support these hypotheses. Specifically, when a constraint is not satisfied, it often indicates non-optimality of the method. This is most evident from the analysis of Okapi and PL2, based on which we successfully predicted the non-optimality of these two functions for verbose queries. In some other cases, when a method only satisfies a constraint for a certain range of parameter values, its performance tends to be poor when the parameter is out of this range, which is shown in the analysis of the pivoted normalization function and the PL2 function. In general, we find that the empirical performance of a retrieval formula is tightly related to how well they satisfy these constraints. Thus the proposed constraints can provide a good explanation of many empirical observations (e.g., the relatively stable performance of the Okapi formula) and make it possible to evaluate a retrieval function *analytically*, which is extremely valuable for testing new retrieval models. Moreover, when a constraint is not satisfied by a retrieval function, it often also suggests a way to improve the retrieval formula.

Since constraint analysis is insufficient if the analyzed retrieval function satisfies all the constraints or analytical analysis of constraints is mathematically difficult, we further propose a diagnostic evaluation methodology to evaluate *experimentally* how well a retrieval function satisfies various retrieval heuristics and diagnose weaknesses and strengths of the retrieval function. We formally define a set of relevance-preserving collection perturbation operators, which can change collection characteristics so as to reveal the weaknesses and strengths of retrieval functions in their implementation of a specific retrieval heuristic. These operators serve as basic tools for us to perform diagnostic tests. We present a common procedure to design the diagnostic tests for retrieval models. Following the procedure, we design three sets of diagnostics tests and perform the tests on six representative data sets. Experiments show that the proposed methodology can (1) identify the weaknesses

and strengths of a retrieval function, (2) explain the empirical differences among retrieval functions, and (3) give hints on how a retrieval function should be modified to further improve the performance. Based on the hints obtained from the diagnostic tests, we derived two new retrieval functions **MPtf2ln** and **MDtf2ln** as improved versions of the pivoted length normalization retrieval function and the Dirichlet prior retrieval function, respectively. Both have been shown to outperform their corresponding original functions not only on the six data sets used to derive them, but also on two new data sets.

Improving existing retrieval models is known to be a hard, yet very important, fundamental problem. The proposed diagnostic evaluation methodology offers a new way to help researchers diagnose weaknesses and strengths of a retrieval function and gain insights about how to improve the function. Our work opens up many interesting future research directions. First, we have only managed to formalize three basic retrieval heuristics (TF, IDF, and document length normalization) in this article; yet, as we have shown, they are already very useful for analytically evaluating a retrieval function. It should be very interesting to further explore, formalize, and test additional retrieval heuristics (e.g., proximity[Tao and Zhai 2007]). With more constraints, constraint analysis would be even more powerful. Second, as in the case of constraints, it is also very interesting to design more diagnostic tests to discover more interesting characteristics of different retrieval functions. It is also interesting to study the potential equivalence of a perturbation to a collection and a change to the functional form of a retrieval function. For example, smoothing of a document language model using a smoothing method such as Dirichlet prior may be regarded as perturbing a collection by adding pseudo counts of terms to each document. Third, in this paper, we have used the insights obtained from diagnostic evaluation of retrieval functions to heuristically improve some existing retrieval functions. Although the improved versions of these functions are shown to be more effective than the original functions, it would be more interesting to study how to systematically address the weaknesses of a retrieval function in a more principled way than simply introducing heuristic parameters. Fourth, the analysis in this paper is restricted to the four basic retrieval models with no pseudo feedback; it would be interesting to apply the proposed method to analyze and improve pseudo feedback methods and more advanced estimation methods for language models (e.g., [Lavrenko and Croft 2001; Zhai and Lafferty 2001a]). Finally, we plan to extend our diagnostic evaluation study by exploring more rigorous mathematical formulations.

10. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their useful comments, which have helped improving the quality of this paper. This paper is based upon work supported in part by the National Science Foundation under grants IIS-0347933 and IIS- 0713581, and by an Alfred P. Sloan Research Fellowship.

REFERENCES

Amati, G. and Rijsbergen, C. J. V. 2002. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Transactions on Information Systems 20,* 4, 357–389.

Carterette, B. and Allan, J. 2005. Incremental test collections. In *Fourteenth International Conference on Information and Knowledge Management (CIKM 2005)*.

Carterette, B., Allan, J., and Sitaraman, R. 2006. Minimal test collections for retrieval evaluation. In *Proceedings of the 2006 ACM SIGIR Conference on Research and Development in Information Retrieval*.

Cormack, G. V., Palmer, C. R., and Clarke, C. L. 1998. Efficient construction of large test collections. In *Proceedings of the 1998 ACM-SIGIR Conference on Research and Development in Information Retrieval*.

Fang, H. 2008. A re-examination of query expansion using lexical resources. In *Proceedings of the 46th Annual Meetings of the Association for Computational Linguistics*.

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

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

Fang, H. and Zhai, C. 2006. Semantic term matching in axiomatic approaches to information retrieval. In *Proceedings of the 2006 ACM SIGIR Conference on Research and Development in Information Retrieval*.

Fuhr, N. 1992. Probabilistic models in information retrieval. *The Computer Journal 35,* 3, 243–255.

Fuhr, N. 2001. Language models and uncertain inference in information retrieval. In *Proceedings of the Language Modeling and IR workshop*. 6–11.

Harman, D. and Buckley, C. 2004. Sigir 2004 workshop: Ria and where can ir go from here. *SIGIR Forum 38,* 2, 45–49.

He, B. and Ounis, I. 2005. A study of the dirichlet priors for term frequency normalisation. In *Proceedings of the 2005 ACM SIGIR Conference on Research and Development in Information Retrieval*.

Hiemstra, D. 2000. A probabilistic justification for using tf-idf term wieghting in information retrieval. In *International Journal on Digital Libraries*. 131–139.

Lafferty, J. and Zhai, C. 2003. Probabilistic relevance models based on document and query generation. In *Language Modeling and Information Retrieval*, W. B. Croft and J. Lafferty, Eds. Kluwer Academic Publishers.

Lavrenko, V. and Croft, B. 2001. Relevance-based language models. In *Proceedings of SIGIR'01*. 120–127.

Lopresti, D. and Zhou, J. 1996. Retrieval strategy for noisy text. In *Symposium on document analysis and information retrieval*.

Ponte, J. and Croft, W. B. 1998. A language modeling approach to information retrieval. In *Proceedings of the ACM SIGIR'98*. 275–281.

Robertson, S. and Sparck Jones, K. 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science 27*, 129–146.

Robertson, S. and Walker, S. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of SIGIR'94*. 232–241.

Robertson, S. and Walker, S. 1997. On relevance weights with little relevance information. In *Proceedings of SIGIR'97*. 16–24.

Robertson, S. E., Walker, S., Jones, S., M.Hancock-Beaulieu, M., and Gatford, M. 1995. Okapi at TREC-3. In *The Third Text REtrieval Conference (TREC-3)*, D. K. Harman, Ed. 109–126.

Salton, G. 1989. *Automatic Text Processing: The Transformation, Analysis and Retrieval of Information by Computer*. Addison-Wesley.

Salton, G. and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing and Management 24*, 513–523.

Salton, G. and McGill, M. 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill.

Salton, G., Yang, C. S., and Yu, C. T. 1975. A theory of term importance in automatic text analysis. *Journal of the American Society for Information Science 26,* 1 (Jan-Feb), 33–44.

Sanderson, M. and Joho, H. 2004. Forming test collections with no system pooling. In *Proceedings of the 2004 ACM-SIGIR Conference on Research and Development in Information Retrieval*.

Shi, S., Wen, J.-R., Yu, Q., Song, R., and Ma, W.-Y. 2005. Gravitation-based model for information retrieval. In *Proceedings of the 2005 ACM SIGIR Conference on Research and Development in Information Retrieval*. 488–495.

Singhal, A. 2001. Modern information retrieval: A brief overview. *Bulletin of the IEEE Computer Society Technical Committee on Data Engineering 24,* 4, 35–43.

Singhal, A., Buckley, C., and Mitra, M. 1996. Pivoted document length normalization. In *Proceedings of the 1996 ACM SIGIR Conference on Research and Development in Information Retrieval*. 21–29.

Singhal, A., Choi, J., Hindle, D., Lewis, D. D., and Pereira, F. C. N. 1998. ATT at TREC-7. In *Text REtrieval Conference*. 186–198.

Singhal, A., Salton, G., and Buckley, C. 1996. Length normalization in degraded text collections. In *symposium on document analysis and information retrieval*. 149–162.

Soboroff, I., Nicholas, C., and Cahan, P. 2001. Ranking retrieval systems without relevance judgements. In *Proceedings of the 2001 ACM-SIGIR Conference on Research and Development in Information Retrieval*.

Sparck Jones, K. and Willett, P., Eds. 1997. *Readings in Information Retrieval*. Morgan Kaufmann Publishers.

Tao, T. and Zhai, C. 2007. An exploration of proximity measures in information retrieval. In *Proceedings of the 2007 ACM SIGIR Conference on Research and Development in Information Retrieval*.

Turtle, H. and Croft, W. B. 1991. Evaluation of an inference network-based retrieval model. *ACM Transactions on Information Systems 9,* 3, 187–222.

van Rijbergen, C. J. 1977. A theoretical basis for theuse of co-occurrence data in information retrieval. *Journal of Documentation*, 106–119.

van Rijsbergen, C. J. 1986. A non-classical logic for information retrieval. *The Computer Journal 29,* 6.

Voorhees, E. M. 2007. Trec: Continuing information retrieval's tradition of experimentation. *Communications of ACM 50,* 11, 51–54.

Wong, K.-F., Song, D., Bruza, P., and Cheng, C.-H. 2001. Application of aboutness to functional benchmarking in information retrieval. *ACM Transactions on Information Systems 19,* 4, 337–370.

Wong, S. K. M. and Yao, Y. Y. 1995. On modeling information retrieval with probabilistic inference. *ACM Transactions on Information Systems 13,* 1, 69–99.

Zhai, C. and Lafferty, J. 2001a. Model-based feedback in the language modeling approach to information retrieval. In *Tenth International Conference on Information and Knowledge Management (CIKM 2001)*. 403–410.

Zhai, C. and Lafferty, J. 2001b. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of SIGIR'01*. 334–342.

Zhou, Y. and Croft, W. B. 2006. Ranking robustness: a novel framework to predict query performance. In *15th International Conference on Information and Knowledge Management (CIKM 2006)*. 567.

Zobel, J. 1998. How reliable are the results of large-scale information retrieval experiments? In *Proceedings of the 1994 ACM SIGIR Conference on Research and Development in Information Retrieval*.

Zobel, J. and Moffat, A. 1998. Exploring the similarity space. *SIGIR Forum 31,* 1, 18–34.

Appendix: Constraint Analysis

We now provide more details for the constraint analysis results described in Section sec:analysis. Instead of going through all constraints, we will focus on only a few as the representatives ones.

The constraints analysis for the three TFCs are similar. We will use TFC1 as an example to show how to get the constraint analysis results for TFC constraints. TFC1 requires that the first partial derivative of a retrieval function with respect to term count, i.e., $c(t, D)$ should be positive when $c(t, D)$ is larger than 0. With the notation $x = c(t, D)$, we can rewrite the first partial derivative of a retrieval function S as a function of x, i.e., $f_S(x)$.

$$f_{piv}(x) = \frac{c(t,Q)log\frac{N+1}{df(t)}}{1-s+s\frac{|D|}{avdl}} \times \frac{1}{(1+ln(x))\times x}$$

$$f_{okapi}(x) = \frac{k_1\cdot(k_1+1)\cdot(1-b+b\frac{|D|}{avdl})}{(k_1\cdot(1-b+b\frac{|D|}{avdl})+x)^2} \times ln\frac{N-df(t)+0.5}{df(t)+0.5} \times \frac{(k_1+1)\times c(t,D)}{k_1((1-b)+b\frac{|D|}{avdl})+c(t,D)}$$

$$f_{dir}(x) = \frac{c(t,Q)}{(1+\frac{x}{\mu\cdot p(t|C)})\times\frac{1}{\mu\cdot p(t|C)}}$$

It is clearly that f_{piv} and f_{dir} are both larger than 0 when x is larger than 0. Thus, it means that Pivoted normalization and Dirichlet Prior can satisfy the TFC1 unconditionally. Moreover, it is clear that whether the value of $f_{okapi}(x)$ is larger than 0 depends on whether the value of $ln\frac{N-df(t)+0.5}{df(t)+0.5}$ is larger 0, so Okapi satisfies TFC1 conditionally. For PL2 method, for the simplicity, we can take the derivative of tfn_t^D instead of $c(t, D)$. Assuming $x = tfn_t^D$, we get

$$f_{pl2}(x) = \frac{1}{(1+x)^2}\times(x+log(\lambda_t\cdot x)+\frac{1}{2x}-\frac{log_2(2\pi\cdot x)}{2}-\frac{log_2e}{a}).$$

When $x = 1$, we have $f_{pl2}(x) = 1 + log(\lambda_t) + 0.5 - log(2\pi) * 0.5 - \frac{1.44}{lambda}$. Thus, $f_{pl2}(x) > 0$ requires that $\lambda_t * log(\lambda_t) + 0.18 * \lambda_t \geq log_2e$. When x is larger, the lower bound of the λ_t becomes smaller. Thus, in order for PL2 to satisfy TFC1, we need to set the value satisfying $\lambda_t * log(\lambda_t) + 0.18 * \lambda_t \geq log_2e$.

We now discuss the constraint analysis results for TDC constraint. TDC requires that the first partial derivative of a retrieval function with respect to TD component should be positive when the term distriminative value, such as $log\frac{N+1}{df(t)}$ in pivoted normalization method. Let us denote $y = TD(t)$, i.e., term discrimination value of term t, we can rewrite get the following first partial derivative of a retrieval function S as a function of y, i.e., $g_S(y)$.

$$g_{piv}(y) = \frac{1+ln(1+ln(c(t,D)))}{(1-s)+s\frac{|D|}{avdl}}\cdot c(t,Q)$$

$$g_{okapi}(y) = \frac{(k_1+1)\times c(t,D)}{k_1((1-b)+b\frac{|D|}{avdl})+c(t,D)} \times \frac{(k_3+1)\times c(t,Q)}{k_3+c(t,Q)}$$

$$g_{dir}(y) = \frac{c(t,D)\cdot c(t,Q)}{1+y\cdot c(t,D)}$$

$$g_{pl2}(y) = \frac{1}{tfn_t^D + 1} \times (\frac{tfn_t^D}{y} - \frac{log_2 e}{y^2})$$

Note that in pivoted, $y = ln\frac{N+1}{df(t)}$; in okapi, $y = ln\frac{N-df(t)+0.5}{df(t)+0.5}$; in dirichlet, $y = \frac{1}{\mu \cdot p(t|C)}$; and in PL2, $y = \lambda_t$. It is clear that pivoted, dirichlet and Okapi can satisfy TDC unconditionally because the corresponding g functions are always larger than 0. For PL2 method, with the notation $tfn_t^D = c(t,D) \times log_2(1 + c \cdot \frac{avdl}{|D|})$, and the assumption that $|D| = avdl$, $g_{pl2}(y) > 0$ requires that $c > 2^{\frac{log_2 e}{\lambda_t}} - 1$.

Finally, we briefly explain how to analyze PL2 method with the LNC2 constraint. The results for the other three methods have been described in Section 3. Given the analysis results of TFCs for PL2, it can be easily shown that LNC2 is equivalent to say that tfn_t^D increases when the values of $c(t,D)$ and $|D|$ increase. Assuming $x = c(t,D)$ and $y = |D|$, we can rewrite tfn_t^D as a function of x and y:

$$tfn_t^D = h(x,y) = x \cdot log_2(1 + c\frac{avdl}{y}).$$

Thus, LNC2 is equivalent to $\frac{dh(x)}{dx} > 0$ and $\frac{dh(y)}{dy} > 0$, which leads to $c \leq \frac{|D|}{avdl}$.

Received May 2007; revised April 2009 and September 2009; accepted March 2010