Adaptive Term Frequency Normalization for BM25

Yuanhua Lv
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
ylv2@uiuc.edu

ChengXiang Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801
czhai@cs.uiuc.edu

ABSTRACT

A key component of BM25 contributing to its success is its sub-linear term frequency (TF) normalization formula. The scale and shape of this TF normalization component is controlled by a parameter k_1, which is generally set to a *term-independent* constant. We hypothesize and show empirically that in order to optimize retrieval performance, this parameter should be set in a *term-specific* way. Following this intuition, we propose an information gain measure to directly estimate the contributions of repeated term occurrences, which is then exploited to fit the BM25 function to predict a *term-specific* k_1. Our experiment results show that the proposed approach, *without needing any training data*, can efficiently and automatically estimate a term-specific k_1, and is more effective and robust than the standard BM25.

Categories and Subject Descriptors

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

General Terms

Algorithms

Keywords

BM25, Term frequency, adaptation, information gain

1. INTRODUCTION

The Okapi BM25 retrieval function [9, 11] has been the state-of-the-art for nearly two decades. The BM25 formula, as presented in [3], scores a document D with respect to query Q as follows:

$$\sum_{q \in Q \cap D} \frac{(k_3+1)\cdot c(q,Q)}{k_3 + c(q,Q)} \cdot dtf(q,D) \cdot \log \frac{N+1}{df(q)+0.5} \quad (1)$$

where $c(q,Q)$ is the count of term q in Q, N is the total number of documents, $df(q)$ is the document frequency of q, and k_3 is a parameter.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
CIKM'11, October 24–28, 2011, Glasgow, Scotland, UK.
Copyright 2011 ACM 978-1-4503-0717-8/11/10 ...$10.00.

A key component of BM25 contributing to its success is its sub-linear term frequency (TF) normalization formula:

$$dtf(q,D) = \frac{(k_1+1)\cdot c(q,D)}{k_1\left(1-b+b\frac{|D|}{avdl}\right)+c(q,D)} = \frac{(k_1+1)\cdot c'(q,D)}{k_1+c'(q,D)} \quad (2)$$

where

$$c'(q,D) = \frac{c(q,D)}{1-b+b\frac{|D|}{avdl}} \quad (3)$$

is the pivoted normalization method [12] for document length normalization. Here $b \in [0,1]$ is the slope parameter, $|D|$ represents document length, $avdl$ stands for average document length, and $c(q,D)$ is the raw TF of q in D.

The scale and shape of this TF normalization component is controlled by a parameter k_1, which is generally set to a *term-independent* constant [9, 6, 3]. Although many studies have attempted to improve BM25 from various perspectives, e.g., [10, 4, 8, 7], in all these studies, the sub-linear TF normalization formula with a fixed and term-independent parameter setting remained the same as the original BM25 [11], and no work has attempted to improve the TF normalization of BM25 through adaptively normalizing TF for individual terms. In this paper, we propose to automatically set k_1 in a per-term basis.

Our work is motivated by the observation that k_1 is intuitively related to the rate of increase in score contribution from matching an additional occurrence of a term. We notice that the optimal rate of score increase for a term generally should depend on the global tendency of the term to be repeated. Specifically, the increase of score from occurrences t to occurrences $t+1$ is related to how many of those documents with at least t occurrences of the term in the collection actually have at least $t+1$ occurrences of the term. Intuitively, if almost all documents containing at least t occurrences of the term actually contain at least $t+1$ occurrences, the score increase should be smaller, whereas if very few of those documents actually contain at least $t+1$ occurrences (i.e., they mostly contain exactly t occurrences), the score increase from t to $t+1$ should be larger. Our intuition that k_1 should be set in a per-term basis is also confirmed in the empirical plots in Figure 1.

Following this intuition, we develop an unsupervised approach to *adaptively* predict the optimal k_1 value for each term based on the counts of documents containing different levels of TF for a query term. Specifically, we propose to measure the contribution of the occurrences of a query term using the information gain of "observing one more token of the same query term", which can be estimated directly based

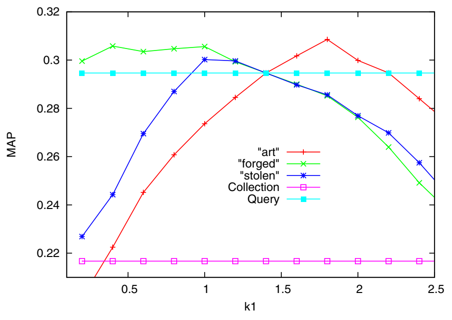


Figure 1: Comparison of different strategies for optimizing k_1 for TREC query topic 422, where we optimize k_1 for the whole collection ("Collection"), for this particular query ("Query"), and for each individual term (while we use the setting of "Query" for the other two terms), respectively.

on term statistics, and then we can fit the BM25 function with these contribution scores to solve a k_1 for each term.

Our experiment results on multiple test collections show that the proposed approach can effectively and efficiently estimate a term-specific k_1, and that it not only achieves better retrieval performance than a well-tuned BM25, but also is more robust than the standard BM25.

2. ADAPTIVE TF NORMALIZATION

We propose to measure the contribution of the observed t occurrences of a query term q in document D using the information gain (denoted as IG_q^t) of "encountering one more occurrence of q" in the current document D. Following the divergence from randomness (DFR) framework [1], we also assume that the probability that the observed term contributions to select a relevant document is high, if the probability of encountering one more token of the same term in a relevant document is similarly high, i.e., if D really talks about q, q would be likely to appear again. The probability of a further success in encountering a term is thus a conditional probability that gradually increases as t increases. However, on the other hand, if successes were brought about by pure chance, then the conditional probability would be constant no matter how t changes, but this constant could still be very large. Considering these intuitions and analysis, we hypothesize that the *information gain* of "encountering one more token of the same term" should be an appropriate measure of the contribution of the observed t term occurrences:

$$\frac{(k_1+1)\cdot t}{k_1+t}\cdot idf(q) \propto IG_q^t \tag{4}$$

Next, we introduce how to compute the information gain IG_q^t and how to estimate k_1.

2.1 Information Gain Measuring Term Contributions

The probability of a further success in encountering a term q conditioned on the current observed t occurrences is denoted as $p(t+1|t,q)$ in our paper. The use of this probability is not entirely novel: the Laplace's law of succession was used to estimate a similar probability in [1]. However, their methods do not consider the characteristics of q and are *term-independent*. In order to estimate a term-specific probability, we exploit term statistics of q. Specifically, we approximate $p(t+1|t,q)$ with the probability of randomly picking a document that will contain at least $t+1$ occurrences of q from a document set of which each document contains at least t occurrences of q, formally,

$$p(t+1|t,q) = \frac{df_{t+1}(q)+0.5}{df_t(q)+1} \tag{5}$$

where 0.5 and 1 are used to smooth the probability to avoid zero probabilities (can be interpreted as hyperparameters of a beta prior distribution). Similarly, we have an "initial occurrence" probability $p(1|0,q)$, which is the probability of randomly picking a document that contains q:

$$p(1|0,q) = \frac{df(q)+0.5}{N+1} \tag{6}$$

Hence, the information gain of "encountering one more token" is the change of *information content* from an initial state to the current state of observing t occurrences:

$$IG_q^t = -\log_2\left(\frac{df(q)+0.5}{N+1}\right) + \log_2\left(\frac{df_{t+1}(q)+0.5}{df_t(q)+1}\right) \tag{7}$$

where the definition of informative content as $-\log_2 P$ was given in semantic information theory [5].

However, it is well-known that longer documents tend to have higher TF. Following BM25, we also choose the pivoted normalization method [12], as shown in Equation 3, to obtain the normalized version of TF as $t = c'(q,D)$. Yet there is another problem after document length normalization: $t = c'(q,D)$ is usually no longer an integer, and thus it may not make sense to talk about the probability of encountering "one" more token conditioned on $c'(q,D)$. Our solution is to simply calculate a list of IG_q^t values only for *integers* $t \in \{0,1,2,\cdots\}$, and then use a few sample IG_q^t values to fit BM25. To estimate the information gain for an integer t, we define $df_t(q)$ in a more formal way.

$$df_t(q) = \begin{cases} |\{D|c'(q,D) \geq t-0.5\}| & \text{if } t \in \{2,3,\cdots\} \\ df(q) & \text{if } t=1 \\ N & \text{if } t=0 \end{cases} \tag{8}$$

Here we use the rounding technique, which intuitively makes sense. For example, it is not reasonable to exclude a document D from $df_2(q)$ if $c'(q,D)=1.99$. Empirical analysis also shows that this rounding technique works very well.

It is observed that, when t becomes very large, the estimated information gain values are often noisy due to document sparseness (i.e., $df_t(q)$ may be too small). To address this problem, we only calculate IG_q^t for $t \in \{0,1,\cdots,T\}$, where T is chosen heuristically as the smallest integer that breaks up the word burstiness rule [2], i.e., $IG_q^T > IG_q^{T+1}$.

It is interesting to see that, besides document length normalization, the information gain measure also achieves both an IDF effect and a term-specific TF normalization effect. Note that $\left(-\log_2\frac{df(q)+0.5}{N+1}\right)$ in Formula 7 is essentially the traditional IDF formula. And according to the word burstiness phenomenon [2], as t increases and becomes large, the second component of IG_q^t generally would also increase, and the increasing rate should be *term-dependent*.

2.2 Fitting BM25 with Information Gains

Empirical evidence shows that it is beneficial to set k_1 in a per-term basis. However, how can we achieve this goal?

Intuitively, k_1 is related to the "score gain" of matching a term t times, so we can try to measure "information gain" and connect k_1 with the proposed IG_q^t, i.e., we can now estimate k_1 by relating the BM25 TF component to IG_q^t.

Specifically, in Formula 4, we assume that the score distribution of the information gain measure is proportional to the score distribution of BM25. Now we align the two distributions at a special point, $t = 1$. We know that when $t = 1$, the left side and the right side of the Formula 4 appear to be $idf(q)$ and IG_q^1 respectively. So we can align two distributions by normalizing both sides of Formula 4 to make their score be 1.0 when $t = 1$, formally,

$$\frac{IG_q^t}{IG_q^1} = \frac{(k_1 + 1) \cdot t}{k_1 + t} \tag{9}$$

Because we have already obtained values $\{IG_q^0, IG_q^1, \cdots, IG_q^T\}$, we can estimate the optimal value of $k_1(q)$, i.e., the q-specific k_1, using a curve-fitting technique and the least square method:

$$\hat{k_1}(q) = \arg\min_{k_1} \sum_{i=0}^{T} \left(\frac{IG_q^t}{IG_q^1} - \frac{(k_1 + 1) \cdot t}{k_1 + t} \right)^2 \tag{10}$$

The estimation of $\hat{k_1}(q)$ only relies on a few IG_q^t values which can be calculated very quickly given the inverted index. So the estimation process is very efficient. Moreover, because $\hat{k_1}$ and IG_q^1 are generally not affected by online factors, they can also be pre-computed offline.

Now we can solve IG_q^t by substituting Equation 10 into 9, and then use IG_q^t to replace the left side of Formula 4, which leads to a BM25-like retrieval function, called BM25-adpt,

$$\sum_{q \in Q \cap D} qtf(q, Q) \cdot \frac{c'(q, D)(\hat{k_1}(q) + 1)}{\hat{k_1}(q) + c'(q, D)} \cdot IG_q^1 \tag{11}$$

It is clear that BM25-adpt has fewer parameters than the standard BM25, since a term specific parameter $\hat{k_1}(q)$ can be estimated automatically using Formula 10.

Moreover, comparing BM25-adpt with the standard BM25, there is also another interesting difference in the implementation of IDF: BM25-adpt uses a novel IDF formula, IG_q^1. From the definition of IG_q^t in Formula 7, we can see that IG_q^1 is essentially a *second-order* IDF, which not only considers the discrimination power of a term (i.e., $-\log_2 \frac{df(q)+0.5}{N+1}$), but also captures *randomness* of the repeated occurrences (i.e., $\log_2 \frac{df_2(q)+0.5}{df_1(q)+1}$). Our experiments have also shown that IG_q^1 works as effectively as the traditional IDF.

3. EXPERIMENTS

We used several standard TREC collections in our study: TREC8, Robust04, WT2G and WT10G. Queries were taken from the title field of the TREC topics. The preprocessing of documents and queries included Porter stemming and stopword removing using a total of 418 standard stopwords.

We used two variations of the standard BM25 function as our baselines, "BM25-basic" and "BM25". They differ from each other only in the parameter setting. Specifically, in "BM25-basic", k_1 was fixed to 1.2 as suggested in [6], while b was well tuned; this tuning strategy has been frequently used in many studies, e.g., [3], in order to reduce the effort of parameter tuning. In "BM25", both b and k_1 were optimized; "BM25" represents the upper bound of the standard BM25 function, which, however, requires significantly more tuning effort than "BM25-basic". In the proposed method, which is labeled as "BM25-adpt", we only tuned parameter b since k_1 was eliminated, so it requires the same amout of effort as "BM25-basic" for parameter tuning. In all runs, $k_3 = 1000$.

Collection	BM25-basic	BM25	BM25-adpt
TREC8	0.2459	0.2557	**0.2595**$^+$
Robust04	0.2507	0.2543	**0.2571**$^+$
WT2G	0.3191	0.3198	**0.3215**$^+$
WT10G	0.2065	0.2099	**0.2122**

Table 1: Optimal performance comparison.

Collection	BM25-basic	BM25	BM25-adpt
Robust04	0.2507	0.2536	**0.2571**$^+$
WT10G	0.2034	0.2056	**0.2091**$^+$

Table 2: Performance comparison using cross validation.

In our experiments, the top-ranked 1000 documents for each run were compared in terms of their mean average precisions (MAP). Note that the superscripts $+$ and $*$ indicate that the MAP improvements of BM25-adpt over BM25-basic and BM25 are statistically significant respectively.

3.1 Performance Comparison

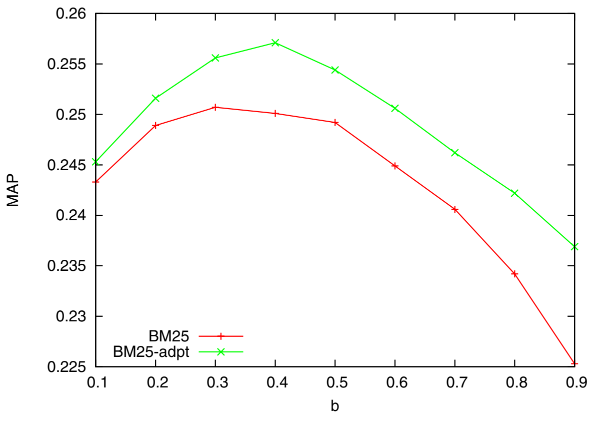
We compare the optimal performance of different methods in Table 1. It is observed that "BM25-adpt" works effectively: its improvements over "BM25-basic" are statistically significant in most cases, and it even outperforms "BM25" consistently. It suggests that our way of automatically setting a term-specific $k_1(q)$, needing significantly less effort for parameter tuning, outperforms the standard way of manually tuning a term-independent parameter.

Comparison of optimal performance alone may raise some concern of over-fitting. So we further use a 5-fold cross-validation method to verify our observations on Robust04 and WT10G. The results are reported in Table 2. Overall, the comparison results are consistent with our previous observations.

In practice, we may not have appropriate training data in the same domain for all queries. So we are also interested in the robustness of a retrieval algorithm when the training and test sets are from different domains. In this experiment, we simulate such a scenario: TREC, WT10G, and WT2G collections are used as the training sets respectively to train the retrieval functions, which are then evaluated on Robust04. The results are presented in Table 3. It shows clearly that the proposed method "BM25-adpt" is more robust, and it significantly outperforms both "BM25-basic" and "BM25". This suggests that our algorithms, although with less parameters, perform more robustly.

3.2 Parameter Analysis

Parameter b in BM25 controls the influence of document length in TF normalization. So we are interested in how the setting of b affects the retrieval performance of our method. We set $k_1 = 1.2$ [6] in the standard BM25 algorithm so as to focus on the sensitivity of parameter b. Note that b is the only parameter that we need to tune in the proposed algorithm "BM25-adpt". The sensitivity curves are drawn in Figure 2 (left). We can see that the setting of b affects the retrieval performance of both methods significantly. However, our method appears consistently more effective and robust



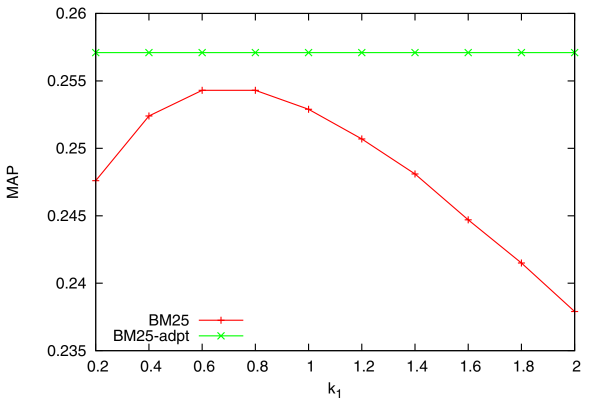


Figure 2: Sensitivity of MAP to parameter b (left) and k_1 (right) on collection Robust04.

Training	BM25-basic	BM25	BM25-adpt
TREC8	0.2507	0.2524	**0.2571**$^{+*}$
WT10G	0.2501	0.2538	**0.2571**$^{+*}$
WT2G	0.2489	0.2527	**0.2556**$^{+*}$

Table 3: Performance comparison on Robust04 using other collections for parameter training.

than the standard BM25 when we vary b. We also see that when b deviates from its optimal setting, the MAP of the standard BM25 often drops more quickly than our method; one possible explanation is that our method can be able to adjust k_1 adaptively for each b.

In the above analysis, we set $k_1 = 1.2$ in the standard BM25. Since a better-tuned k_1 can often lead to better performance as we have shown in Figure 1, so we want to know how critical this k_1 is and how likely a standard BM25 with manually tuned k_1 can achieve performance similar to our method. We thus set the parameter b in each method to its corresponding optimal value, and then vary $k1$ from 0.2 to 2.0 to examine the sensitivity of k_1, which is shown in Figure 2 (right). We can see that the standard BM25 is quite sensitive to k_1, and even with the optimal setting for k_1, its performance is still lower than "BM25-adpt".

4. CONCLUSIONS AND FUTURE WORK

BM25 has been a state-of-the-art retrieval function for a long time. The important parameter k_1 has so far been manually set to a term-independent constant based on a training data set. In this paper, we proposed a novel unsupervised approaches to *automatically* and *adaptively* set parameter k_1 in BM25 for each term and each collection. Specifically, we developed an information gain measure to directly estimate the contributions of repeated term occurrences, which was explored to fit the BM25 function to solve a term-specific k_1 without needing any training data. Our experiment results on multiple test collections show that the proposed approach works more effectively and robustly than the standard BM25 with a manually tuned k_1. As a "by product" of the derived formulas, we effectively eliminated the k_1 parameter from BM25. Thus with the proposed adaptive BM25 formula, we only need to tune parameter b, which can reduce significantly the effort for parameter tuning.

There are many interesting problems for future research. For example, how to also adaptively set another important parameter in BM25, i.e., b? how to adaptively set k_1 at the query level instead of term-level, since terms in the same query should interact with each other in TF normalization?

5. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their useful comments. This material is based upon work supported by the National Science Foundation under Grant Numbers IIS-0713581, CNS-0834709, and CNS-1028381, a Sloan Research Fellowship, and a Yahoo! Key Scientific Challenge Award.

6. REFERENCES

[1] Gianni Amati and Cornelis Joost Van Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. *ACM Trans. Inf. Syst.*, 20:357–389, October 2002.
[2] Kenneth W. Church and William A. Gale. Poisson mixtures. *Natural Language Engineering*, 1:163–190, 1995.
[3] Hui Fang, Tao Tao, and ChengXiang Zhai. A formal study of information retrieval heuristics. In *SIGIR '04*, pages 49–56, 2004.
[4] Ben He and Iadh Ounis. On setting the hyper-parameters of term frequency normalization for information retrieval. *ACM Trans. Inf. Syst.*, 25, July 2007.
[5] Jaakko Hintikka. On Semantic Information. In J. Hintikka and P. Suppes, editors, *Information and Inference*, pages 3–27. D. Reidel Pub., 1970.
[6] K. Sparck Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: development and comparative experiments. In *Information Processing and Management*, pages 779–840, 2000.
[7] Yuanhua Lv and ChengXiang Zhai. Lower-bounding term frequency normalization. In *CIKM '11*, 2011.
[8] Yuanhua Lv and ChengXiang Zhai. When documents are very long, bm25 fails! In *SIGIR '11*, pages 1103–1104, 2011.
[9] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *SIGIR '94*, pages 232–241, 1994.
[10] Stephen Robertson, Hugo Zaragoza, and Michael Taylor. Simple bm25 extension to multiple weighted fields. In *CIKM '04*, pages 42–49, 2004.
[11] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *TREC '94*, pages 109–126, 1994.
[12] Amit Singhal, Chris Buckley, and Mandar Mitra. Pivoted document length normalization. In *SIGIR '96*, pages 21–29, 1996.