

# A Study of Parameter Tuning for Term Frequency Normalization

Ben HE  
Department of Computing Science  
University of Glasgow  
Glasgow, UK  
ben@dcs.gla.ac.uk

Iadh Ounis  
Department of Computing Science  
University of Glasgow  
Glasgow, UK  
ounis@dcs.gla.ac.uk

## ABSTRACT

Most current term frequency normalization approaches for information retrieval involve the use of parameters. The tuning of these parameters has an important impact on the overall performance of the information retrieval system. Indeed, a small variation in the involved parameter(s) could lead to an important variation in the precision/recall values. Most current tuning approaches are dependent on the document collections. As a consequence, the effective parameter value cannot be obtained for a given new collection without extensive training data. In this paper, we propose a novel and robust method for the tuning of term frequency normalization parameter(s), by measuring the normalization effect on the within document frequency of the query terms. As an illustration, we apply our method on Amati & Van Rijsbergen's so-called normalization 2. The experiments for the ad-hoc TREC-6,7,8 tasks and TREC-8,9,10 Web tracks show that the new method is independent of the collections and able to provide reliable and good performance.

## Categories and Subject Descriptors

H.3 [INFORMATION STORAGE AND RETRIEVAL]:  
Miscellaneous

## General Terms

Experimentation, Performance

## Keywords

Information retrieval, term frequency normalization, parameter tuning, document length

## 1. INTRODUCTION

The term frequency normalization has been one of the main issues in information retrieval (IR) for many years.

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'03, November 3–8, 2003, New Orleans, Louisiana, USA.  
Copyright 2003 ACM 1-58113-723-0/03/0011 ...\$5.00.

Robertson and Walker iterated the effect of document length in [5]:

“Some documents may simply cover more material than others (scope), . . . , a long document covers a similar scope to a short document, but simply use more words (verbosity)”

Indeed, the term frequency is dependent on the document length. Therefore, the need to smooth the document length effect by a technique called *term frequency normalization*.

In general, approaches for term frequency normalization are *parametric*. They assume a fixed form of density function with parameters. The estimation of these parameters is a crucial issue, which also tackles the collection dependency problem.

A classical method for tuning the term frequency normalization parameters is the pivoted normalization [8]. In 1996, Singhal et. al. studied a set of collections and proposed a heuristic normalization method by “pivoting” the normalization factor to fit the relevance judgement information:

$$(1.0 - slope) + slope \cdot \frac{factor}{factor_{avg}} \quad (1)$$

where *factor* and *factor<sub>avg</sub>* are the normalization factor and the average normalization factor respectively. The normalization factor is obtained by using a specific normalization method, e.g. the Cosine normalization [7]. *slope* is a parameter. They claimed that *slope* = 0.2 is effective across collections.

As one of the most well established IR systems, Okapi's normalization method is similar to the pivoted normalization. In the so-called Okapi BM25 document weighting function, the *idf* factor  $w^{(1)}$  is normalized as follows [6]:

$$w = w^{(1)} \frac{(k_1 + 1)tf}{K + tf} \frac{(k_3 + 1)qtf}{k_3 + qtf} \quad (2)$$

where  $w$  is the final weight;  $K$  is:

$$k_1((1 - b) + b \frac{l}{avgJ})$$

$l$  and  $avgJ$  are the document length and the average document length in the collection respectively;  $k_1$ ,  $k_3$  and  $b$  are parameters;  $qtf$  is the number of occurrences of a given term in the query;  $tf$  is the within document frequency of the given term.

Let both the numerator and the denominator of  $\frac{(k_1+1)tf}{k+tf}$  be divided by  $tf$ , the equation (2) becomes:

$$w = w^{(1)} \frac{k_1 + 1}{\frac{k_1((1-b)+b \cdot \frac{l}{avg_l})}{tf} + 1} \frac{(k_3 + 1)qtf}{k_3 + qtf}$$

$$\text{Let } tfn = \frac{tf}{k_1((1-b)+b \cdot \frac{l}{avg_l})},$$

$$\begin{aligned} w &= w^{(1)} \frac{k_1 + 1}{\frac{1}{tfn} + 1} \frac{(k_3 + 1)qtf}{k_3 + qtf} \\ &= w^{(1)} \frac{(k_1 + 1)tfn}{tfn + 1} \frac{(k_3 + 1)qtf}{k_3 + qtf} \end{aligned} \quad (3)$$

Hence the term frequency normalization component of the BM25 formula can be seen as:

$$tfn = \frac{tf}{k_1((1-b)+b \cdot \frac{l}{avg_l})} \quad (4)$$

This is quite similar to the pivoted normalization. Both formulas assume an interpolated form of function for the factor. The default setting of the Okapi system is  $k_1 = 1.2$ ,  $k_3 = 1000$  and  $b = 0.75$  [9]. The setting was obtained by taking into account the relevance judgement in experiments on a merged data set of TREC collections [9]. However, in [4], Chowdhury et. al. argue that it is necessary to recalibrate the parameters *slope* and  $b$  for different collections. In their experiments for the TREC-9 and TREC-10 Web tracks, even within the supposed “safe” range of  $b$  values, the mean average precision could decrease by up to 36%.

Recently, Amati & Van Rijsbergen proposed four normalization methods in [1], [2], and [3]. In their experiments, the most robust and effective one is the so-called “normalization 2”. The normalization 2 assumes that term frequency density is a decreasing function of the document length. Its normalization formula is:

$$tfn = tf \cdot \log_2(1 + c \cdot \frac{avg_l}{l}), (c > 0) \quad (5)$$

The default setting of the parameter  $c$  was obtained by taking into account the relevance judgement on the TREC collections. However, similarly to the *slope* of the pivoted normalization and the parameter  $b$  of the BM25 formula, the setting needs to be tuned for different collections. For example, using the PL2 weighting scheme in which the normalization2 is applied [2], the optimum parameter value for the TREC-1 ad-hoc task is  $c = 1$ , whilst for the TREC-9 Web Track, the optimum parameter value is  $c = 2$ , which can achieve 8.52% higher mean average precision than  $c = 1$ .

In this paper, we propose a novel method for tuning the term frequency normalization parameters, which is independent of the document collections. To illustrate the method, we apply it on Amati & Van Rijsbergen’s normalization 2.

According to the work of Amati & Van Rijsbergen, the normalization component of the BM25 formula in equation (4) can be derived from the normalization 2 [2]. Therefore, we believe that our method could cope with the approaches mentioned above.

The remainder of the paper is organized as follows. In section 2 we describe our method in details. In section 3 we illustrate our method on Amati & Van Rijsbergen’s so-called normalization 2. The evaluation of the method and its associated experiments and results are given in section 4. Conclusions and future work are summarized in section 5.

## 2. A METHOD FOR PARAMETER TUNING OF TERM FREQUENCY NORMALIZATION

In this section, we propose a collection independent method for automatically tuning the parameters of a term frequency normalization approach. Our approach is based on the notion of *normalization effect*. The basic assumption behind our approach is the following:

*Assumption 1*

The optimum parameter values on different collections give similar normalization effect.

Although the notion of normalization effect has appeared in several publications, there is no detailed study and clear definition proposed so far. In [9], Sparck-Jones, Walker and Robertson mentioned the notion of normalization effect for the BM25’s normalization method (equation (4)):

“If the new tuning constant  $b$  is set to 1, the simple normalization factor is used. Smaller values reduce the normalization effect.”

In their description, the normalization effect is the change of the within document frequency. In this paper, we define the change by the quotient of the normalized term frequency divided by the original term frequency:

$$NE_{d_i}(\alpha) = \frac{tfn_{d_i}(\alpha)}{tfd_i} \quad (6)$$

where  $\alpha$  is a vector of the involved parameters;  $NE_{d_i}(\alpha)$  is the normalization effect on the document  $d_i$  by setting the parameter vector to  $\alpha$ .

The equation (6) illustrates the normalization effect on a single document. For a weighting scheme applying a term frequency normalization method, the normalization effect on a single document can lead to the change of the ranking of the documents. So the term frequency normalization affects not only a single document, but also the set  $D$  of documents containing (at least one of) the query terms<sup>1</sup>. We define the normalization effect on the set  $D$  as follows:

$$NE_D(\alpha) = \frac{Var(NE_{d_i}(\alpha))}{\mu}, d_i \in D \quad (7)$$

where  $NE_D(\alpha)$  is the normalization effect on  $D$  by setting the parameter vector to  $\alpha$ ;  $Var$  denotes the variance;  $\mu$  is used as a calibrating factor, which is the mean value of  $\{NE_{d_i}(\alpha)\}$  in  $D$ .

When the normalization component is disabled, which leads to  $NE_{d_i} = 1$  for all the documents in  $D$ ,  $NE_D(\alpha)$  is zero.

*Example*

Let us give an example to show how the equation (7) works. Assume that the set  $D$  has three documents. Let  $\alpha_1$  and  $\alpha_2$  be two parameter vectors. Assume that using equation (6), we get  $\{NE_{d_i}(\alpha_1)\} = \{1, 2, 3\}$  and  $\{NE_{d_i}(\alpha_2)\} = \{3, 6, 9\}$ . We can see that the use of vector  $\alpha_2$  gives three times larger normalization effect on each document, than

<sup>1</sup>In Amati & Van Rijsbergen’s framework, each model only associates a relevance degree to the documents in the set  $D$ . [2].

using vector  $\alpha_1$ . Using the equation (7), we will obtain  $NE_D(\alpha_1) = \frac{1}{3}$  and  $NE_D(\alpha_2) = 1$ . So  $NE_D(\alpha_2)$  is also three times larger than  $NE_D(\alpha_1)$ .

The reader might have noticed that in equation (7), we have chosen a distribution of normalization effect on the set of documents with respect to (wrt.) a query rather than wrt. a single query term. Indeed, in this paper, we focus on the parameter tuning with respect to a query. However, if we can have a definition for  $NE_D(\alpha)$  wrt. a query term, it is possible to estimate the parameters of term frequency normalization for each query term.

We have already defined the normalization effect on a single document (i.e.  $NE_{d_i}$ ) and on the document set  $D$  (i.e.  $NE_D$ ). However, according to our definition of normalization effect in equations (6) and (7),  $NE_D(\alpha)$  increases with the variance of the document length in  $D$ . Since  $D$  is a sample of documents in the collection, the variance of document length in  $D$  is collection dependent, so  $NE_D(\alpha)$  is also collection dependent. Hence the need to normalize  $NE_D(\alpha)$  to make it independent of the document collections. In this paper, we normalize  $NE_D(\alpha)$  by dividing it by  $NE_{D,max}(\alpha)$ , the maximum  $NE_D(\alpha)$  value wrt. different settings of  $\alpha$  (assuming that the maximum value exists). According to our definition of the normalization effect,  $NE_{D,max}(\alpha)$  is dependent on the used term frequency normalization method. Later we will prove that for the normalization 2, the maximum value of the normalization effect on  $D$  does exist.

We denote the normalized  $NE_D(\alpha)$  as  $NE n_D(\alpha)$ . According to the basic assumption of our method, i.e. Assumption 1, for a given collection, when the assigned parameter values give the highest mean average precision value,  $NE n_D(\alpha)$  should be a constant. We denote this constant as  $NE n_D(\alpha_{optimum})$ . Once we obtain  $NE n_D(\alpha_{optimum})$  from the training data, for a given new collection, we will assign the parameter vector such that it gives  $NE n_D(\alpha_{optimum})$ .

### 3. APPLICATION ON THE NORMALIZATION 2

Our approach, described above, could be applied to all term frequency normalization methods mentioned in the introduction section. As an illustration, we apply our method on the normalization 2 in equation (5).

For the normalization 2, the parameter vector  $\alpha$  is  $\{c\}$  where  $c > 0$ . In order to normalize  $NE_D(c)$ , we need to obtain the maximum  $NE_{D,max}(c)$ . To do so, let us study the monotonic behavior of the function  $NE_D(c)$ .

Let us assume a continuous and increasing distribution of  $NE_{d_i}$  from 0 to  $NE_{d,max}$ , where  $NE_{d,max}$  is the maximum  $NE_{d_i}$  in the document set  $D$ . Since most normalization methods assume higher  $tf$  density for shorter documents,  $NE_{d,max}$  is the  $NE_{d_i}$  of the shortest document in  $D$ . Then we approximate  $NE_D$  by:

$$\begin{aligned} NE_D &= \frac{Var(NE_{d_i})}{\mu} \\ &= \frac{n \sum_D NE_{d_i}^2}{\sum_D NE_{d_i}} - \sum_D NE_{d_i} \\ &\approx \frac{n \int_0^{NE_{d,max}} NE_d^2 d(NE_d)}{\int_0^{NE_{d,max}} NE_d d(NE_d)} - \int_0^{NE_{d,max}} NE_d d(NE_d) \\ &= \frac{2n NE_{d,max}}{3} - \frac{NE_{d,max}^2}{2} \end{aligned} \quad (8)$$

where  $n$  is the number of documents within  $D$ .

Its derivative is:

$$\begin{aligned} NE'_D &\approx \left( \frac{2n NE_{d,max}}{3} - \frac{NE_{d,max}^2}{2} \right)' \\ &= \frac{2n}{3} \cdot NE'_{d,max} - NE_{d,max} \cdot NE'_{d,max} \end{aligned} \quad (9)$$

Thus  $NE_D$  has its maximum value when  $NE_{d,max} = \frac{2n}{3}$  (i.e. the solution of the function  $NE'_D = 0$ ). For the normalization 2,  $NE_{d,max}$  is an increasing function of its parameter. So  $NE'_D$  is a monotonic decreasing function of the parameter  $c$ , the curve of the function  $NE_D(\alpha)$  should have a bell shape. When  $c = \xi$ , which is the solution of function  $NE'_D(c) = 0$ ,  $NE_D$  is maximum. Also,  $NE_{d,max} = \frac{2n}{3}$ , the maximum normalization effect in set  $D$ , seems to be unreasonably large. We believe that it is because of the error of the approximation in equation (8). However, the error couldn't affect the monotonic behavior of the function  $NE_D(c)$ . In our experiments,  $\xi$  ranges from 2.00 to 4.70 in all involved TREC collections.

Moreover, since the curve of the function  $NE_D(c)$  should have a bell shape, there might exist two  $c$  values that are on the different sides of the "bell" and give the same  $NE_D(c)$  value. Since for a given  $NE n_D(c)$  value, we want only one  $c$  value that corresponds to it, we should take the derivative of the function  $NE_D(c)$  into consideration. Thus we normalize  $NE_D(c)$  by the following equation:

$$NE n_D(c) = \frac{NE_D(c)}{NE_{D,max}(c)} \cdot \tau \quad (10)$$

where  $\tau$  indicates if  $NE_D(c)'$  is positive or negative. When  $NE_D(c)' > 0$ ,  $\tau = 1$  and when  $NE_D(c)' < 0$ ,  $\tau = -1$ .

For the normalization 2, the Assumption 1 can be refined as:

For the collections  $Coll_1, Coll_2, \dots, Coll_n$ , if the optimum parameter value in the collections are  $c_1, c_2, \dots, c_n$ , then  $NE n_{D, Coll_i}(c_i)$ , the normalized normalization effect for each collection by assigning the optimum parameter value, is a constant.

We denote the optimum  $NE n_D(c)$  as  $NE n_D(c_{optimum})$ . It can be obtained from relevance judgement on a training data set. Once we obtain  $NE n_D(c_{optimum})$ , for a given new collection, we will assign the  $c$  value such that it gives  $NE n_D(c_{optimum})$ .

## 4. EXPERIMENTS

### 4.1 Experimental Setup

The main goal of the experiments is to show that our method is independent of the collections. As described in the last section, our method needs a training data set to obtain  $NE n_D(c_{optimum})$ . Therefore, our experiments include two parts: first, we run extensive experiments on the training data set and obtain  $NE n_D(c_{optimum})$  by relevance judgement; second, we evaluate our method on other collections to test if it has collection independence.

In our experiments, we use the disk1 and disk2 of the TREC collections as the training data set. On this data set, we test  $c$  values from 0.1 to 32 with an interval of

0.1 for the TREC-1,2 and 3 ad-hoc tasks where each task has 50 topics. Each topic consists of 3 fields: title, description and narrative. In our experiments we use two types of queries: short query (title only) and long query (title+description+narrative). For each task, we measure the mean average precision for each  $c$  value, and consider the  $c$  value that gives the highest mean average precision as the optimum value. Then for each task, we obtain the  $NE_D(c)$  value for the optimum  $c$  value. At last, we compute the mean value of the optimum  $NE_D(c)$  for the three tasks and consider it as  $NE_D(c_{optimum})$ .

To evaluate the  $NE_D(c_{optimum})$  obtained from the training data set, we run experiments for the TREC-6,7,8 ad-hoc tasks on the disk4 and disk5 (no CR database in the TREC-7,8 ad-hoc tasks), the TREC-8 Web track on the WT2G collection and the TREC-9,10 Web tracks on the WT10G collection. The collections involved in our experiments are heterogeneous, and their document length distributions are quite different. Table 1 lists the standard deviation of the document length in the collections.

We consider the default setting of the normalization 2 as our baseline. The default setting is  $c = 7$  for short query and  $c = 1$  for long query. It was empirically obtained by extensive experiments on the TREC collections and has been considered as the optimum value for the TREC collections [1]. For each task, we assign the parameter value such that we obtain  $NE_D(c_{optimum})$  and compare it to the baseline.

In our experiments, both documents and queries are stemmed, tokens from a standard stop-words list are removed and no query expansion is applied. In each run we assign the same parameter value for all queries and documents, and compute the mean  $NE_D(c)$  value of all queries for the assigned parameter value.

In all our experiments, we use Amati & Van Rijsbergen’s PL2 model as the weighting scheme [2]. Using the PL2 model, the weight of a query term within a document is given by:

$$w(t, d) = Inf_1 \cdot Inf_2 \quad (11)$$

where

$$Inf_1 = tf \cdot \log_2 \frac{tf}{\lambda} + (\lambda + \frac{1}{12 \cdot tf} - tf) \cdot \log_2 e + 0.5 \cdot \log_2(2 \cdot tf)$$

and

$$Inf_2 = \frac{1}{tf + 1}$$

Replacing the raw term frequency  $tf$  by the normalized term frequency  $tfn$  by using the normalization 2 in equation (5), we obtain the final weight.

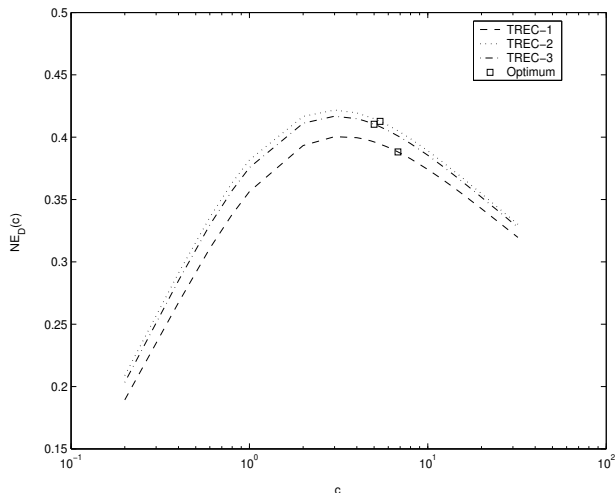
## 4.2 Experimental Results

First, we run experiments on the training collection. As shown by the results, for the short queries, the optimum  $NE_D(c)$  values for the TREC-1,2,3 ad-hoc tasks are on the decreasing side of the curves of the function  $NE_D(c)$  (see Figure 1). The system performance is effective and robust in a large interval of optimum parameter values (about  $3 < c < 20$ , see Figure 2).

For the long queries, the parameter tuning becomes more risky. There is only a small segment of parameter values that can be considered as “safe” (see Figure 4). The optimum

**Table 1: The standard deviation ( $\sigma$ ) of document length in the TREC collections.**

TREC collections	TREC Tasks	$\sigma$
disk1 & disk2 (Training set)	1,2,3 ad-hoc	862.50
disk4 & disk5	6 ad-hoc	1200.72
disk4 & disk5 (No CR)	7,8 ad-hoc	558.12
WT2G	8 Web	2009.22
WT10G	9, 10 Web	2303.32



**Figure 1: The mean  $NE_D(c)$  and the parameter  $c$  on the TREC-1,2,3 ad-hoc tasks for short queries. The optimum  $NE_D(c)$  values are on the decreasing side of the curves in all three tasks.**

$NE_D(c)$  values are on the increasing side of the curves of the function  $NE_D(c)$  (see Figure 3).

In the three training tasks,  $NE_D(c)$  is a decreasing function of  $c$  when  $c > 0$ . The curves of the function  $NE_D(c)$  form a bell shape in all three tasks, which supports our prediction of the monotonic behavior of the function  $NE_D(c)$  in equation (8), (9). Table 3 lists the maximum  $NE_D(c)$  of all tasks involved in our experiments.

From the relevance judgement of the training data set, we obtain the optimum  $NE_D(c)$  value for each task. The values are listed in Table 2. Using the equation (10), we get the optimum  $NE_D(c)$  values for the three tasks. Computing the mean value of the optimum  $NE_D(c)$  values for the three tasks, we obtain  $NE_D(c_{optimum})$ , i.e.  $NE_D(c_{optimum}) = -0.9773$  for short query and  $NE_D(c_{optimum}) = +0.9793$  for long query respectively (see Table 4).

Then, we run experiments for the TREC-6,7,8 ad-hoc tasks and the TREC-8,9,10 Web tracks to evaluate our method. For each run, we assign the  $c$  value that gives  $NE_D(c_{optimum})$ , which was obtained from the training set as mentioned above. Here we denote the  $c$  value that gives  $NE_D(c_{optimum})$  as  $c_{tuning}$ . For each task, we measure the average precision for  $c_{tuning}$  and the baseline respectively.

The results for short queries are listed in Table 5. Table 5 shows that our method achieves similar performance to the baseline in the experiments. The effective parameter

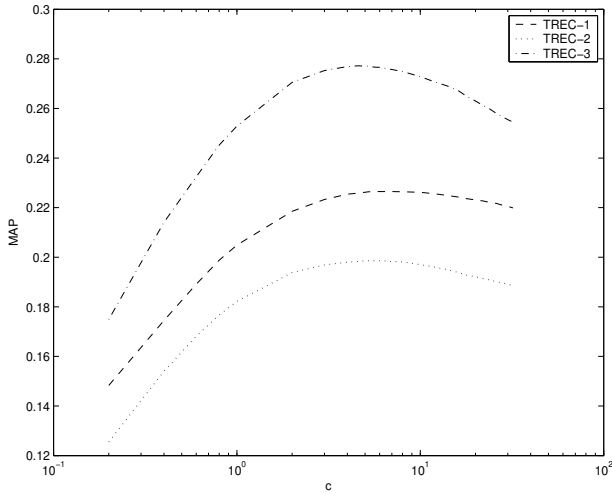


Figure 2: The mean average precision(MAP) and the parameter  $c$  on the TREC-1,2,3 ad-hoc tasks for short queries. The normalization 2 is effective in a large interval of parameter values.

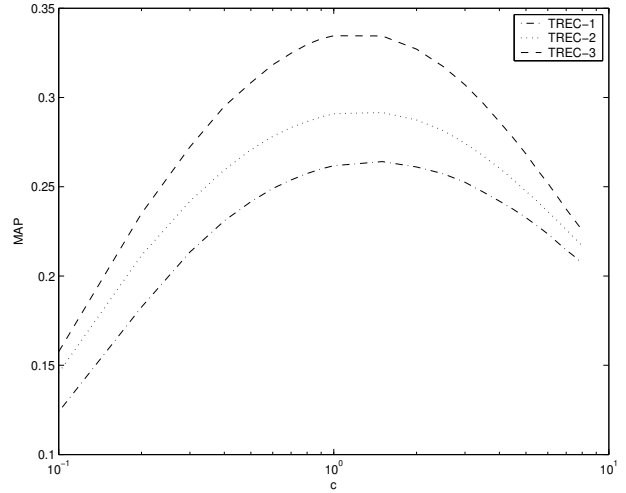


Figure 4: The mean average precision(MAP) and the parameter  $c$  on the TREC-1,2,3 ad-hoc tasks for long queries. The normalization 2 is only effective in a small range of parameter values.

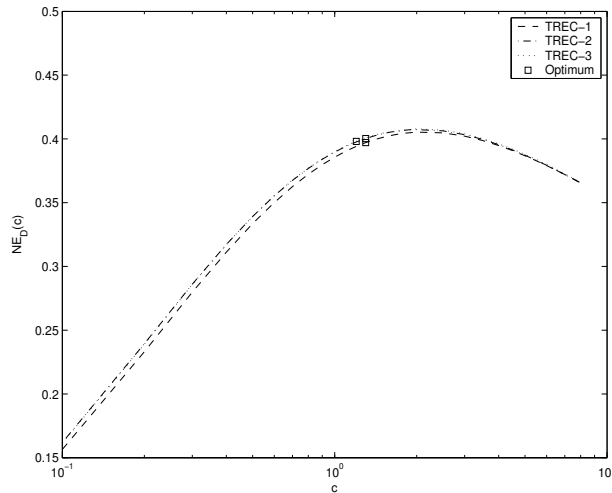


Figure 3: The mean  $NE_D(c)$  and the parameter  $c$  on the TREC-1,2,3 ad-hoc tasks for long queries. The curves are almost indistinguishable. The optimum  $NE_D(c)$  values are on the increasing side of the curves in all three tasks.

Table 2: The optimum average  $NE_D(c)$  value in TREC-1,2,3 tasks for two types of queries. (+) and (-) indicate the derivative of  $NE_D(c)$  is positive or negative respectively.

TREC task	Short Query		Long Query	
	$NE_D(c)$	$c$	$NE_D(c)$	$c$
1 ad-hoc	0.3882(-)	6.80	0.3971(+)	1.30
2 ad-hoc	0.4127(-)	5.40	0.4003(+)	1.30
3 ad-hoc	0.4104(-)	5.00	0.3980(+)	1.20

Table 3: The maximum  $NE_D(c)$  in different TREC tasks for two types of queries.

TREC task	Short Query		Long Query	
	$NE_D(c)$	$c = \xi$	$NE_D(c)$	$c = \xi$
1 ad-hoc	0.4006	3.29	0.4054	2.15
2 ad-hoc	0.4218	3.00	0.4074	2.07
3 ad-hoc	0.4168	3.06	0.4080	2.07
6 ad-hoc	0.3373	3.36	0.3334	2.06
7 ad-hoc	0.3162	2.93	0.2855	2.22
8 ad-hoc	0.3164	3.03	0.2818	2.29
8 web	0.7478	3.83	0.6307	2.07
9 web	0.7320	4.54	0.5736	2.02
10 web	0.7187	4.69	0.6015	2.22

values are indeed accurately located. Also, it seems that the normalization 2 is effective for short query across collections.

Results for long queries are also encouraging (see Table 6). Compared to the baseline, our method achieves similar mean average precision in the TREC-6 ad-hoc task and improves the mean average precision in the TREC-7,8 ad-hoc tasks and the TREC-8,9,10 Web tracks.

### 4.3 Discussion

In our experiments, it seems that the normalization 2 is robust for short queries. Indeed, using the default setting, we can achieve effective performance across the TREC collections. However, for the long queries, our approach outperforms the baseline in 5 TREC tasks out of 6.

In the experiments on the training set, for long queries, the function  $NE_D(c)$  seems to be independent of the query sets (see Figure 3). The curves of the three tasks are almost indistinguishable. Whilst for the short queries, there is a clear difference among the curves (see Figure 1). The difference between the curves of the two types of queries might result from the effect of the query length. When the query is

**Table 4: The optimum average  $NE_{n_D}(c)$  value in TREC-1,2,3 tasks for two types of queries.  $NE_{n_D}(c)$  is computed according to equation (10).**

	Short Query	Long Query
TREC task	$NE_{n_D}(c)$	$NE_{n_D}(c)$
1 ad-hoc	-0.9690	+0.9795
2 ad-hoc	-0.9784	+0.9826
3 ad-hoc	-0.9846	+0.9755
Average $NE_{n_D}(c)$	-0.9773	+0.9793

**Table 5: MAP(mean average precision) for short queries by using the parameter value obtained from our tuning method compared to the baseline ( $c=7$ ).  $c_{tuning}$  is the parameter value that is obtained by our tuning method.**

TREC task	$c_{tuning}$	MAP	Baseline	$\Delta$ (%)
6 ad-hoc	6.30	0.2349	0.2350	$\approx 0$
7 ad-hoc	5.15	0.1887	0.1894	$\approx 0$
8 ad-hoc	5.36	0.2537	0.2571	-1.32
8 web	7.31	0.3102	0.3103	$\approx 0$
9 web	9.20	0.2052	0.2042	+0.490
10 web	9.50	0.2086	0.2091	-0.239

**Table 6: MAP(mean average precision) for long queries by using the parameter value obtained from our tuning method compared to the baseline ( $c=1$ ).  $c_{tuning}$  is the parameter value that is obtained by our tuning method.**

TREC task	$c_{tuning}$	MAP	Baseline	$\Delta$ (%)
6 ad-hoc	1.27	0.2241	0.2250	-0.004
7 ad-hoc	1.39	0.2199	0.2149	+2.33
8 ad-hoc	1.43	0.2450	0.2397	+2.21
8 web	1.15	0.2545	0.2481	+2.58
9 web	1.13	0.2218	0.2182	+1.65
10 web	1.24	0.2297	0.2221	+3.42

longer, the document set  $D$  should be larger, which implies that a larger number of documents are sampled, so the document length distribution in  $D$  is more similar to the document length distribution of the whole collection. According to our definition of the normalization effect,  $NE_D(c)$  is dependent on the document length distribution in  $D$ . So we might obtain similar curves of the function  $NE_D(c)$  for different (long) query sets on the same collection. Also, since for the long queries,  $NE_D(c)$  is more dependent on the collections than for the short queries, the parameter tuning for the long queries should be more risky than for the short queries.

Moreover, the optimum normalization effect for short query and long query are significantly different. This could link to the study by Zhai & Lafferty [10], where three smoothing methods were studied. Although their work is from a language modelling perspective, it shares some relative features of term frequency normalization. In their experiments, the performance of the smoothing methods is heavily correlated to the query length. For this unexpected result, they suggest that perhaps the purpose of the smoothing methods is not only to improve the document model, but also to “explain” the common and non-informative words within the query terms. Indeed, the term frequency normalization works like “explaining” each query term by discounting or increasing the within term frequency, which can be represented by the normalization effect. When the query length is different, the needed “explaining power” (the normalization effect) for each query term is different.

## 5. CONCLUSIONS AND FUTURE WORK

In this paper, we have studied the collection dependency problem of the term frequency normalization parameters tuning. By measuring the normalization effect, we propose a novel method, which is collection independent, for tuning the term frequency normalization parameter. Our method assumes that the optimum parameter values should give a constant normalization effect across collections. As an illustration, we have successfully applied our method on Amati & Van Rijsbergen’s normalization 2.

Our experiments show that the proposed method is able to select term frequency normalization parameters that are as effective as good hand-tuned parameters. Indeed, in most tasks, our method achieves similar results to the hand-tuned baseline for the short queries and outperforms the baseline for the long queries. Hence, the effectiveness and robustness of our method is proved.

In this paper, the proposed method has been illustrated on a specific term frequency normalization, namely the normalization 2. However, the method could also be applied on BM25’s normalization method. The main difficulty of the application of our method on BM25 is that the  $b$  value, giving the maximum  $NE_D$ , could be out of the range of  $[0, 1]$ . To solve this problem, we have proposed another version of definition for  $NE_D(\alpha)$ , the normalization effect on the set of documents containing (at least one of) the query terms, instead of the definition provided in equation (7). The details of the application of our method on BM25 will be submitted for publication in the near future.

Finally, according to our experimental results, the query length has strong effect on the tuning of the term frequency normalization parameters. As a consequence, there is currently a gap between the  $NE_{n_D}(c_{optimum})$  values of the long

queries and the short queries. In the future, we will study the effect of query length on term frequency normalization and integrate it with our study in this paper, so that our method can be independent of not only the document collections, but also the queries.

## 6. ACKNOWLEDGMENTS

This work is funded by a UK Engineering and Physical Sciences Research Council (EPSRC) project grant, number GR/R90543/01. The project funds the development of the Terrier Information Retrieval framework (URL: <http://ir.dcs.gla.ac.uk/terrier>). We would like to thank Gianni Amati and Vassilis Plachouras for their useful comments and support. We would also like to thank Ryen White for his comments on an initial version of this paper.

## 7. REFERENCES

- [1] G. Amati. *Probabilistic Models for Information Retrieval based on Divergence from Randomness*. PhD thesis, Department of Computing Science, University of Glasgow, 2003.
- [2] G. Amati and C. J. V. Rijsbergen. Probabilistic models of information retrieval based on measuring the divergence from randomness. In *ACM Transactions on Information Systems (TOIS)*, volume 20(4), pages 357 – 389, October 2002.
- [3] G. Amati and C. J. V. Rijsbergen. Term frequency normalization via pareto distributions. In *Advances in Information Retrieval, 24th BCS-IRSG European Colloquium on IR Research Glasgow, UK, March 25-27, 2002 Proceedings.*, volume 2291 of *Lecture Notes in Computer Science*, pages 183 – 192. Springer, 2002.
- [4] A. Chowdhury, M. C. McCabe, D. Grossman, and O. Frieder. Document normalization revisited. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 381–382, 2002.
- [5] S. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232–241, 1994.
- [6] S. Robertson, S. Walker, M. M. Beaulieu, M. Gatford, and A. Payne. Okapi at trec-4. In *NIST Special Publication 500-236: The Fourth Text REtrieval Conference (TREC-4)*, pages 73 – 96, 1995.
- [7] G. Salton, A. Wong, and C. Yang. A vector space model for information retrieval. *Journal of American Society for Information Retrieval*, 18(11):613 – 620, November 1975.
- [8] A. Singhal, C. Buckley, and M. Mitra. Pivoted document length normalization. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 21–29, 1996.
- [9] K. Sparck-Jones, S. Walker, and S. E. Robertson. A probabilistic model of information retrieval: Development and comparative experiments. *Information Processing and Management*, 36(2000):779 – 840, 2000.
- [10] C. Zhai and J. Lafferty. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 334 – 342, 2001.